Universitat Politècnica De Catalunya

MASTER THESIS

A comparative study of demand forecasting models for an online retailer business

Author
Raul Lorenzo Villagrasa

Supervisor
Argimiro Arratia Quesada

June 2021

A thesis submitted in fulfillment of the requirements for the Master in Innovation and Research in Informatics, Facultat d'Informàtica de Barcelona (FIB)

UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

Abstract

Nowadays, with the current situation caused by **COVID-19**, understanding the customer behaviour, being able to predict the outcome given a date or a product, knowing which items are the most sold and understanding what features of the products are the most influential is essential for the success of the retailer. This study evaluates and compares different demand forecasting models for an online retailer business.
We designed and implemented a system which is able to prepare the data recollected by the online retailer, extract the necessary features for a hybrid model, show to the user relevant business information using an interpretability layer and predict the future sales. In addition, the system is able to predict what kind of customer is more susceptible to buy a product and it categorises new products which are not in the system.
The results of the study aim at increasing the knowledge of the business and improving its profitability.

Contents

List of Figures

List of Tables

1 Introduction

The COVID-19 pandemic that started at the end of 2019 and beginning of 2020 lead to a critical situation for retails companies whose market was focused on physical stores. Due to the mandatory quarantine, only those companies which were offering online services continued business through this rough situation.
Clearly, the retail market has evolved to an online service where the consumer can get everything they want from home. Because of that, the retailers have to think and adopt new ways to proceed immediately. The only way to get through this situation is to analyse all the data available and to anticipate the future landscape of the company. A new business reality will shape after this pandemic.

Now more than ever, machine learning techniques can help to exploit data to get insights to understand the business model in order to be able to make changes. For instance, these techniques allow to predict the customer demand and inventory, and to optimize supply decisions in real-time. They can provide information in how to become more efficient providing to the customers what they want and understanding their behaviour. Also, machine learning techniques can be helpful when doing pricing strategies, avoiding overrated/underrated prices for different products.

Basically, machine learning techniques can show us how to become more efficient in the decision-making of any business, in this particular case, in retail companies.

The process of making estimations about the future based on historical sales' data is called **Demand Forecasting**. This field has been studied for years and we can find a vast literature applying demand forecasting methods to different scenarios, not only in retail companies, but in general.
The main goal of these studies on demand forecasting models is to determine those that best suit for the task of forecasting, as a stock and inventory demand, earnings prediction and future products trends.
After reviewing the literature, we have found that depending on the origin of the source, there are three general methodologies:

- Judgmental

  Based on subjective information. In this type of methodology, a person with a considerable knowledge about the company and the market makes the forecast.

- Statistical

  In this kind of approach, we consider all the classical approaches from causal methods like regression analysis to rule-based-forecasting models

- Extreme machine learning approaches

  In this type of method, we consider all the neural networks (NN) approaches

In this work, we will be studying Statistical and Extreme machine learning approaches to construct Demand forecasting models.

1.1 What is Camper?

Camper [24] is a contemporary footwear brand and online retail company with headquarters in the island of Mallorca, Balearic Islands, Spain. Founded in 1975, Camper is descendant from a shoe-making family business which was created to respond to the demand for a new and fresh style of footgear.
The origins of the company began in 1877, when Antonio Fluxa, who was a skilled cobbler, travelled from Mallorca to England and returned with the first sewing machines on the island. From that moment, the family business started to develop until the company that it is now.

The name of Camper came from the word *campesino*, which means peasant in Spanish, the austerity and simplicity of the rural world which combined with the history and the culture from the Mediterranean landscape lead to the creation of the brand's basis.

Nowadays, Camper's shoes are still designed and developed in Inca, in the rural heart of Mallorca. Now, in its fourth generation, the brand's footprint has stretched around the globe with stores in more than 40 countries.

1.2 Main goals of the thesis

The aim of this master thesis is to produce a software capable of analysing the online sales to extract insights from the data, and predicting the future sales to generate proper reports that Camper can take advantage of. In addition, this software has an interpretability layer in order to help the user to understand the reasons of the decisions made by the system.

To accomplish this goal, we faced several challenges:

- To understand the current business model from Camper and how the data insights can help to improve it.
- To study in depth the current state of the art to know what techniques are the best approaches for our work.
- To create a methodology to compare the models that will be built.
- To study and design an explainability layer that would help understand how our models use the information to make their decisions.

2 State of the art

The aim of this section is to provide a concise overview of the currently demand forecasting methods and approaches used in the actual industry.

J. Scott Amstrong, a well known forecasting and marketing expert and K. C. Green, also a demand forecasting expert promote simpler models over complicated ones in order to perform predictive analysis of the historical data to forecast the future demand. In the paper [1], they reviewed the literature related to the relative accuracy of demand forecasting methods. They concluded that there are different ways to proceed depending on the problem faced.
For instance, in those cases where there is not enough data to fit models, they show that in order to improve the judgment, impose a structure with survey of intention or expectations, judgmental bootstrapping, structured analogies and simulation interactions.
When there is a lack of input features, causal methods or econometric methods are a good fit.
Finally, J. Scott Amstrong and K. C. Green do not recommend to use complex econometric methods and they recommend to avoid quantitative methods that have not been validated and not use those models that do not use domain knowledge [1]. Ultimately, they urge to improve forecasting and decision-making.

In the paper [11], the authors compare statistical based models, extreme learning machines and grey models, in a real world scenario. The authors used different indexes to make the comparison, accuracy, speed, DSR [2], stability and ease of use. They showed how the statistical based models are the ones with the best results, outperforming all the other methods.

Even though in most of the literature reviewed it is proved that neural networks are not recommendable to use, there is one kind of neural network which is not very used in the literature in the scope of online retail companies, Recurrent Neural Network, concretely Long Short-Term Memory (LSTM).
In the paper [18] a comparison is performed among different sorts of neural networks, and defines an approach using LSTM networks which outperforms all the other models. Proving that it is possible to optimize LSTM networks to the point of getting good performances for demand forecasting.

Machine learning models tend to be more and more complex to the point that they are becoming black boxes [3]. It is becoming harder to trace the relationship between the input variables and the model outcomes and this result in a lack of transparency and understanding. Developers are creating unverified models based on their accuracy but without the knowledge of how variables are combined to make the predictions so this will

[1] Domain knowledge is knowledge of the field related to the data to which it belongs. I.e. neural networks, stepwise regression, ...

[2] Data Sufficient Requirements

[3] Black box is a model which the designer cannot understand how variables are combined to make the predictions.

lead to a path of failure.

In this study, we will be focused on knowing how input variables are used in the models and what impact they have on the final prediction.

There are currently several packages which can be used to interpret models.

DALEX [19] is a package able to understand any supported model [4], and helps to understand how complex models work. It differentiates between two levels, instance and dataset.
In the instance level, it creates a wrapper around the predictive model, which is able to create a predictive diagnosis, a profile of the prediction, creates predictions by parts and finally predicts the outcome of the model. Otherwise, in the dataset level, with the same wrapper, it is able to create a model diagnosis, a profile of the model, to calculate the variable importance and eventually to extract the performance of the model.
With this package it is easy to see how the model changes depending on a particular observation only if one or a few variables change. Also, it allows to decompose the response model into additive attributions.

LIME [9] is a package able to explain any black box classifier. Mainly support for scikit-learn classifiers.
The main idea of this method is that locally, every model's behaviour could be explained as a linear approximation. Although a model can be very complex, with this assumption is easier to approximate it around the neighbourhood of a particular instance. This method is based on perturbing the input and seeing how it affects to the predictions.

Captum [21] is an open source package for model interpretability built in PyTorch [5]. Captum provides a lot of methods to implement interpretability algorithms that allow the ML researches improve and troubleshoot models by facilitating the identification of different features that contribute to the response model variable. The target is to design better models and fix unexpected outputs.

Shapley values. In the paper by Erik Strumbelj and Igor Kononenko [4], they show a general method for explaining individual predictions of different classification models based on fundamental concepts from coalitional game theory. This algorithm avoids the exponential time complexity using sampling-based approximation. The main idea of this approximation is calculating the contributions of the i-th feature, comparing their contributions with and without it. It can be applied to any machine learning algorithm and the explanations are very intuitive.

[4]Supported models are scikit-learn, keras, H2O, tidymodels, xgboost, mlr or mlr3.
[5]PyTorch is an open source machine learning framework.

3 Methods and Data

3.1 General Methodology

In this research project, methodology is defined by the general strategy which we are going to follow to accomplish our purposes. We will provide a structured approach to plan our processes in the project, which will include all the steps and the tasks involved in each phase, and their relationships.
In this case, we are going to apply the Cross Industry Standard process for Data Mining, known as CRISP - DM [6], a robust and well-proven methodology.

Figure 1: CRISP-DM process diagram [8]

Business & Data understanding

After the first step (Business Understanding) done in the early stages of the project, in which we understood the context and the goals of the project, we proceeded to our second and crucial step called Data understanding. Its main objective is to know what kind of data we have, its quality, completeness, distributions, etc. So, we will perform an exploratory data analysis in order to extract all the valuable information. In this stage, we will be able to determine new objectives to develop, apart from the main ones, which could be interesting to go in depth.

[6] CRISP - DM is an open standard process which describes common approaches used in Data Mining projects. This methodology covers the phases of the project. It consists of 6 steps, although it is not mandatory to follow all of them. These steps are Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment

Data Preparation and Modelling

In this stage, we are going to prepare all the data to fit the models we will perform. Moreover, the optimizations of the models are settled in this step, so our final result will be the final models to evaluate.

Evaluation

Finally, we will verify that the results are correct and valid, to do that, we are going to use various quality measures, that we detail below, although we will be focused in **RMSE**:

Mean Absolute Error (MAE),
This is the average absolute value of the difference between the predicted value and the true value.

Mean Squared Error (MSE)
This is the average of the squared difference between the predicted value and the observed value. It is a non-negative measure, so the main point is to minimize it. It assesses the quality of the predictor.

Root Mean Squared Error (RMSE)
Standard deviation of the prediction errors (residuals). RMSE measure how spread out are these residuals. To make it easy, it tells you how the data is concentrated in the best fit line.

R-Squared (R^2)
It is the proportion of variance in the dependent variable, which is predictable from the independent variable.

In addition, an interpretability layer will be implemented to understand in depth the models created. This will lead to knowing better why the business is going as it goes, and will allow redefining the route path to follow in order to improve their results.

3.2 Data understanding

The aim of this section is to explore and understand the different data given by Camper to work with. The actual data can be described in terms of a table where the rows contain the records (information) and the columns represent the features.
The main dataset contains approximately 1 million rows and 45 columns of sales information, collected during a period of three years (2018, 2019 and 2020) from different inter-database sources. The attributes are divided into categorical and quantitative data. Looking into the categorical attributes, it can be seen different levels of granularity, each features not having the same number of categories.
In order to check the quality of the dataset, we have to consider the following factors:

- Degree of usefulness of the features
- Completeness of the dataset

In **Appendix A** [6], we will find the information regarding to the features of the dataset, where we explain them briefly. Taking them into consideration, we have removed the descriptive columns, which do not add valuable information, and also the redundant information as there were several columns which contained the same informnation. So we have ended it up with the following features (details of the meaning of each features can be seen in Appendix A subsection [6.1]):

- ANO_FACTURA
- CUSTOMER_ID
- MES_FACTURA
- FECHA_FACTURA
- IMP_VENTA_NETO_EUR
- TEMPORADA_COMERCIAL_ID
- PRODUCTO_ID
- TALLA
- ESFUERZO_VENTA_ID
- NUMERO_DEUDOR_PAIS_ID
- JERARQUIA_PROD_ID
- GRUPO_ARTICULO_PRODUCTO_ID
- GENERO_PRODUCTO
- CATEGORIA
- TIPOLOGIA
- CONSUMER_COLOR
- CREMALLERA
- CORDONES
- OUTSOLE_SUELA_TIPO
- OUTSOLE_SUELA_SUBTIPO
- PLANTILLA_EXTRAIBLE
- CONTACTO_SN
- EDAD_SN
- GENERO_CONTACTO
- EDAD_COMPRA
- EDAD_RANGO_COMPRA
- CIUDAD_CONTACTO
- IDIOMA_CONTACTO

3.3 Exploratory data analysis

In this section, we are going to perform an exploratory data analysis of the dataset. The analysis is going to be separated in four stages:

- Data preparation
- Data insights
- Customer segmentation
- Principal component analysis

Since the dataset is very extensive and Camper has not specified what kind of information they want to document, it is possible that we could not investigate all the insights from the data, but we have tried to reflect all the significant information.

3.3.1 Missing data

This missing data analysis process performs these three primary demands:

- Pattern of missing data
- Missing values located
- Amount of missing values

Using the variables explained in the previous section, we are going to analyse if there are features which do not have information. Once we get them, we are going to calculate the percentage of amount of data that these missing values represent in our system, so we will be able to see if we can apply any imputation technique.

Camper has informed us that the missing values which they have localized are filled with the symbol **NV**. Apart from that, we will look for Nan and Null values, and other kind of symbols which can be used to represent a missing value.

Feature	Number of missing values	Percentage (%)
CUSTOMER_ID	1939	0.15
TALLA	27983	2.11
ESFUERZO_VENTA_ID	951	0.07
JERARQUIA_PROD_ID	21423	1.61
GRUPO_ARTICULO_PRODUCTO_ID	21423	1.61
GENERO_PRODUCTO	48370	3.64
CATEGORIA	49115	3.7
TIPOLOGIA	60648	4.57
CONSUMER_COLOR	46823	3.53
CREMALLERA	67440	5.08
CORDONES	60057	4.52
OUTSOLE_SUELA_TIPO	61259	4.61
OUTSOLE_SUELA_SUBTIPO	60816	4.58
PLANTILLA_EXTRAIBLE	74242	5.59
GENERO_CONTACTO	1939	0.15
CIUDAD_CONTACTO	11297	0.85
IDIOMA_CONTACTO	3561	0.27

Table 1: Percentage missing values in features

There was not any correlation among the features with missing information and the ones without. We could not find a pattern. Also, the amount of data which was not represented was always less than the 6%, so, this will not be a problem for our analysis and modelling as we do not have a huge amount of missing values in these columns.
In addition, the column called EDAD_SN specify if the value of age is given or not.

EDAD_SN	Number of rows
Informed	788261
Non-informed	307038
Final Percentage (%)	28.03

Table 2: Missing values Age

Filtering these data by the non-informed parameter, we have realised that all ages that were encoded with 0, in fact they were missing values.

There are several methods which we can use in order to impute the missing data, in this case, we are going to explore these two different methods:

- $\mathbf{K}_{nn}$: The strategy for imputing missing values is based on the assumption that a point value can be approximated by the values of the closer points to it.
- **Multivariate imputer**: The strategy for imputing missing values is based on modelling each feature with missing value as a function of other features in a round-robin fashion, estimating each feature from all the others.

The problem found exploring the viability of using these methods of imputation is that, for instance, there was a product with three different values for the feature color (black, brown and missing value) and the same characteristics which were related with the product itself. So, when we tried to impute this value, we realized that it was not possible as the correctness of the imputation could not be guaranteed since sometimes it imputed as black and others as brown.
This occurs in several features, it actually adds noise and falsify our data so instead of impute the data, that it can lead to errors, we will keep the non-values but we will encode them with the same code.

3.3.2 Outlier detection

The observations in the dataset that do not fit in some way can be found in the following columns:

- IMP_VENTA_NETO_EUR
- EDAD_COMPRA

Representing the information in quartiles, we have found several outliers that were out of the ranges in both cases.

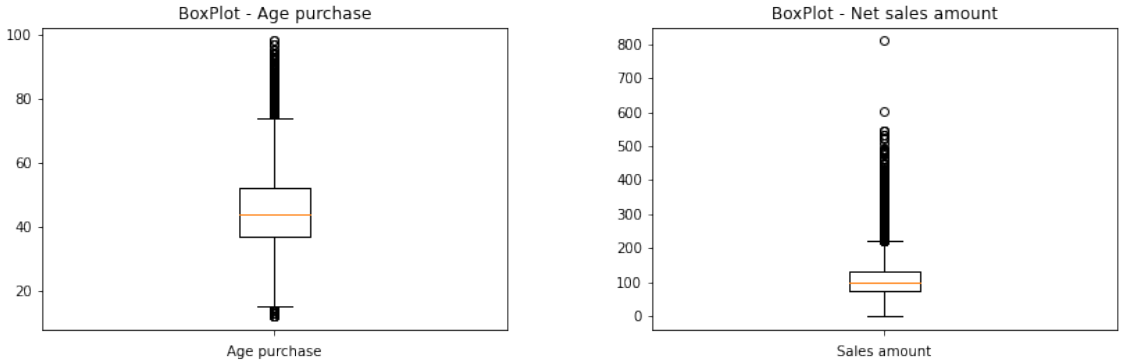

Figure 2: Outlier detection

Net sales amount analysis

As we could see in the quartile representation, the minimum amount of euros earned was 0 and the highest was more than 800€. The number of products cheaper than 10€ was 9918.
After requesting information to Camper regarding to the prices of products, they said that the products where the amount of money earned was less than 10€ were incorrect and the categories we have to focus on are:

- 01 $->$ Adult shoes
- 03 $->$ Factories shoes

- 04 − > Bags
- 08 − > Kids
- 10 − > Firsts Walkers

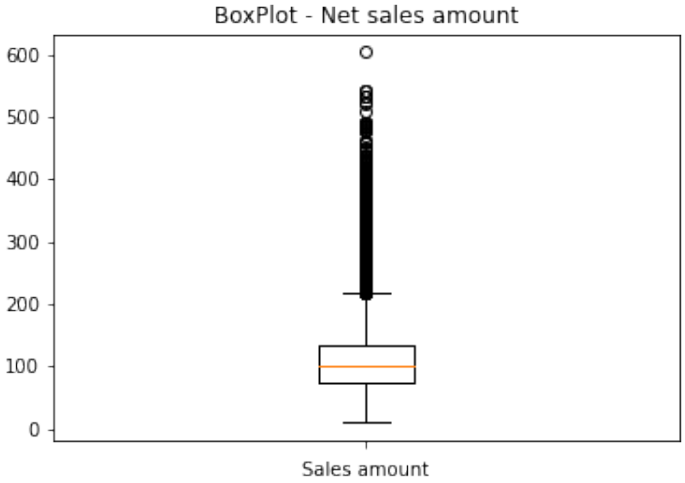

Figure 3: Outliers removed

Finally, all the possible outliers were removed and we could not consider the remaining values as outliers such that all of them are possible.

Age analysis of customers

The minimum age of a customer who had bought products was 12 years old, and the maximum age was 98. Although these results may seem unlikely, they could be possible, so we could not remove them.
In figure [4], it can be seen the distribution of the ages. Looking it closer, in the red squares we can see the amount of products bought by customers who are less than 20 and more than 75, there are very few of them. As we can see, it follows a Gaussian distribution.

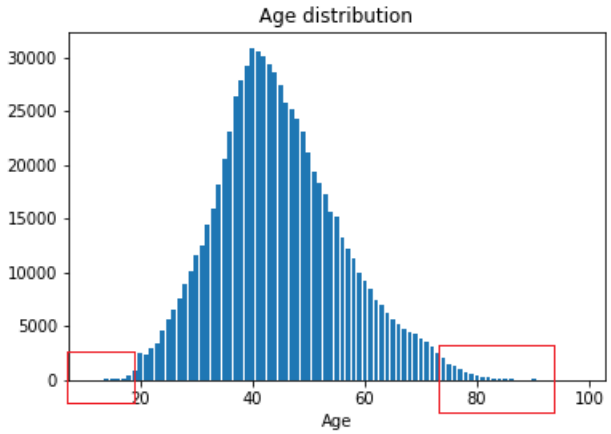

Figure 4: Age distribution of customers

3.3.3 Feature engineering

In this section, we are going to extract some features from the dataset that will help us to understand and create the future models. Also, our main target is to remove data redundancy and prepare it for the analysis.

Date related

Firstly, we have one feature with the full date saved, and variables like **month** and **year** that already have this information, so instead of the full date, we will remove it and save only the day of the sale.
Also, we are going to extract the day of the week, for future analysis during the data insights.

Product features related

In many columns related to the characteristics of the products we have the same information in different languages, in order to have homogeneity in the data and to avoid errors, such that the meaning of them will be the same and they can be coded differently, we are going to unify them into a single language.

3.3.4 Data insights

Our main target is to extract all the knowledge from analyzing the information from the datasets. The analysis of this information will provide insights that will help Camper business make decisions and reduces risks besides to understand the current reality of the company.

Profit by year

In table [3], we will find the money earned per year from 2018 until 2020.

Year	Total profit
2018	31108951.09€
2019	36233669.68€
2020	48325415.06€

Table 3: Money earned in 2018, 2019 and 2020

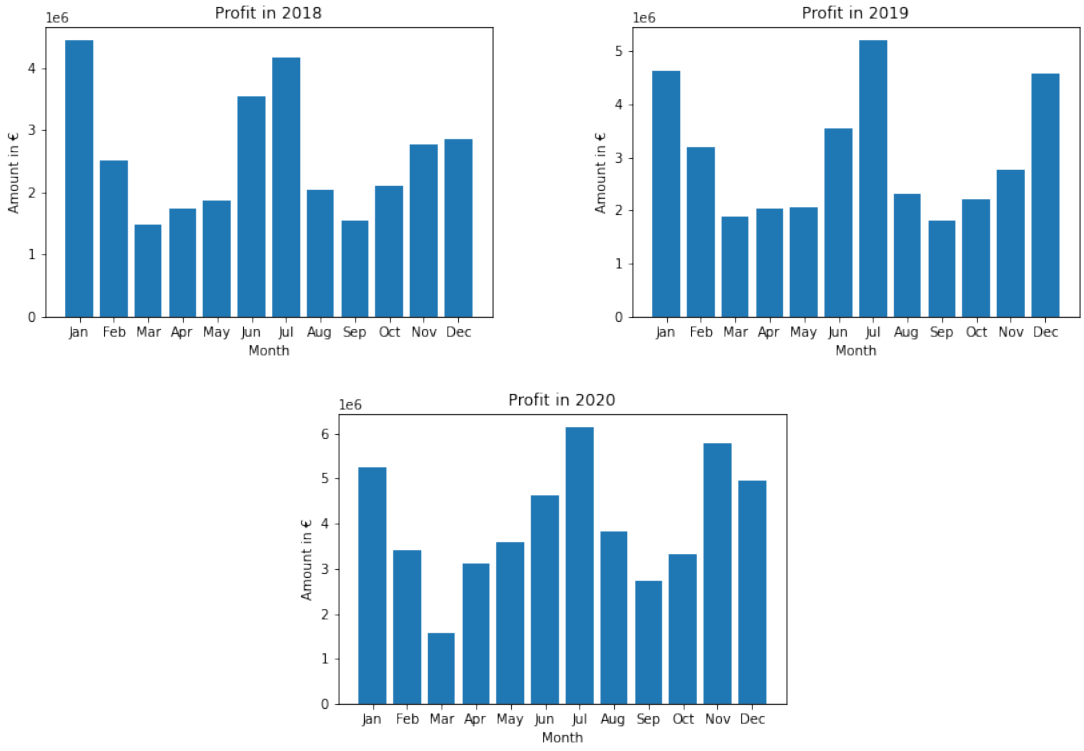

Figure 5: Profit each year

In figure [5], we can see the total profit each month by years. Checking the results, we observe some peaks during the months of January, June and July, which suggest a seasonal pattern in sales.
In 2018, it could be seen the months where the profit was higher were January, June and July. Probably this was because of the discounts they did (Sales). We will see it later.
In 2019, the highest profit was in January, July and December. Notice that on average all the sales have been increasing every month over the previous year.
In 2020, we had a strange phenomenon, when we mentioned before that the sales were increasing in comparison between 2018 and 2019 month by month, in this case the sales have dropped in the March and they started to increase again.
The reason is simple, **COVID-19**. A lot of people lost their jobs because the factories and

companies where they were working closed as a result of the pandemic. In fact, almost all countries stopped their productions and everything was locked down. That's why March was the lowest income in those years.
After this month, the income started to increase and it hit its maximum in July.
But what was the cause of this increase? In order to survive as a company, Camper changed their business model and commenced a new very assertive e-commerce campaign. This new strategy led to a sales' increase, although some of their competitors went bankrupt. Also, this change affected the consumer's pattern that we will see later, in the density figure [6].
What happens in November? In order to reactivate the economy before Christmas, in several countries the sales were advanced (discount done during some months of the year).
In order to see the distribution of the data, we have calculated the density of the previous results. They look like a bi-modal representation, where this reveals that there were two different types, when there were sales and when they were not.
2018 and 2019 follow a similar distribution, having slightly differences; on the contrary, 2020 has a totally different shape, a new customer behaviour. The reason being as aforementioned.

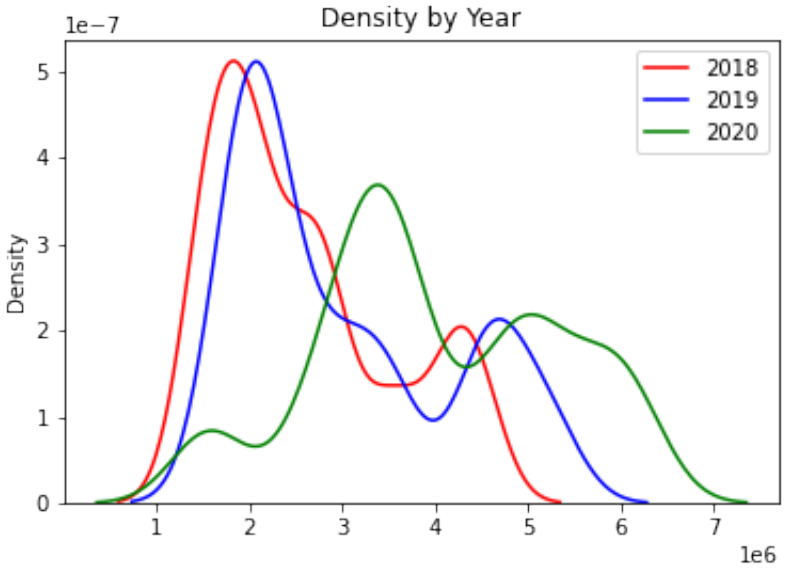

Figure 6: Profit density by year

Benefit from each month (%)			
	2018	2019	2020
Jan	14.3	12.75	10.86
Feb	8.06	8.83	7.05
Mar	4.8	5.19	3.26
Apr	5.57	5.64	6.44
May	6.03	5.68	7.42
Jun	11.39	9.78	9.57
Jul	13.42	14.35	12.71
Aug	6.59	6.38	7.92
Sep	5.0	5.02	5.62
Oct	6.78	6.07	6.88
Nov	8.91	7.66	12.0
Dec	9.17	12.64	10.27

Table 4: Monthly benefit

Most profit products by year

Most products sold					
2018		2019		2020	
Product	€	Product	€	Product	€
21595-018	122690.29	21595-018	122260.82	18811-033	181077.75
20848-017	109714.58	K200387-004	106950.33	K400325-004	178853.90
K100243-001	103069.51	K200564-001	103231.71	K100249-013	163948.64
17665-014	101830.23	18869-059	99211.39	21595-018	160819.22
K200387-004	100184.83	K200508-007	98511.62	K201037-004	157686.25
K200564-001	99468.04	17665-014	95579.81	K400325-009	134007.88
K200157-002	99186.78	46104-094	94436.47	K200733-004	127189.72
K400295-001	97003.57	20848-017	91429.12	17665-199	121292.06
18648-003	86631.07	K100243-001	88438.27	17665-014	117662.31
K200491-001	86342.54	K200491-001	87054.04	K100243-001	116260.13

Table 5: Most products sold each year

These were the common products sold:

- 21595-018 − > Women Moccasin/Ballerina
- 20848-017 − > Women Shoe
- 17665-014 − > Men Shoe
- K200387-004 − > Women Moccasin/Ballerina
- K200564-001 − > Women Sandal
- 17665-014 − > Men Shoe
- K100243-001 − > Men Shoe
- K200491-001 − > Women - Semi-open Shoe

All elements from the list above belong to the category group 01, Adult shoes.
Also, it was interesting to see which were the best selling products each month. In this way, we were able to see if there was any month where the product hit a maximum. In this case, we were focused on the top ten products.

2018		2019		2020	
Month	Product	Month	Product	Month	Product
4	21595-018	4	21595-018	12	18811-033
4	20848-017	6	K200387-004	11	K400325-004
7	K100243-001	7	K200564-001	7	K100249-013
7	17665-014	7	18869-059	4	21595-018
6	K200387-004	4	K200508-007	7	K201037-004
7	K200564-001	7	17665-014	10	K400325-009
7	K200157-002	1	46104-094	7	K200733-004
1	K400295-001	4	20848-017	7	17665-199
1	18648-003	7	K100243-001	7	17665-014
7	K200491-001	7	K200491-001	7	K100243-001

Table 6: Monthly top products

Looking at the results, in 2018, 8/10 products were best-sellers in January/June/July (Sales) and finally, in 2019 and in 2020 7/10 products were best-sellers during sales season. We could conclude that sales had a huge influence on demand.

Season profit

The commercial season in Camper has two different categories, Autumn - Winter (odd numbers) and Spring - Summer (even numbers). So every year, we will find three different codifications for each season.

- 2018: 85, 86 and 87
- 2019: 87, 88 and 89
- 2020: 89, 90 and 91

Knowing this, we are going to analyse which season is more successful for each year.

- 2018
Season Autumn - Winter: 6953948.71€
Season Spring - Summer: 14867118.1€
Season Autumn - Winter: 9287884.28€

- 2019:
Season Autumn - Winter: 7819328.66€
Season Spring - Summer: 17037854.9€

Season Autumn - Winter: 11376486.12€

- 2020:
Season Autumn - Winter: 8656207.62€
Season Spring - Summer: 22866646.96€
Season Autumn - Winter: 16802560.48€

The most successful season was Spring - Summer in 2020, with approximately €22.87M.

Representing the season information above as:

- 85 − > red, 86 − > blue, 87 − > green (2018)
- 87 − > red, 88 − > blue, 89 − > green (2019)
- 89 − > red, 90 − > blue, 91 − > green (2020)

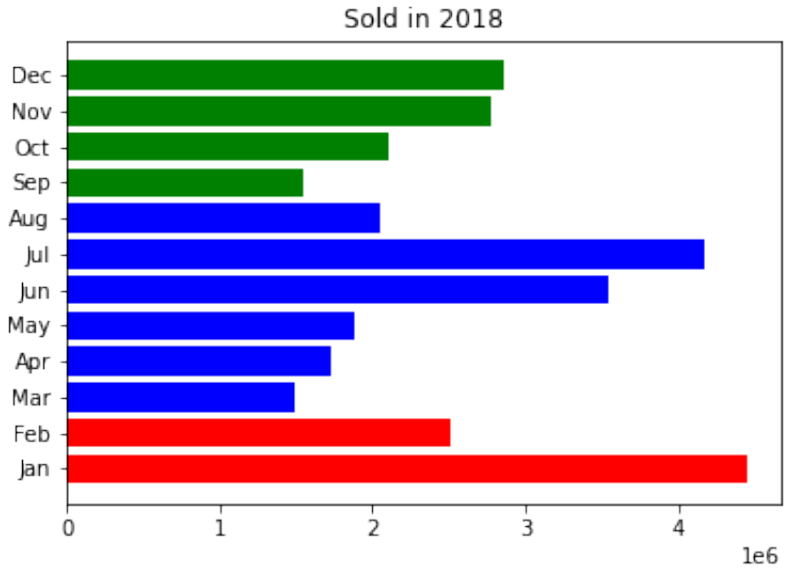

Figure 7: Season money earned in 2018

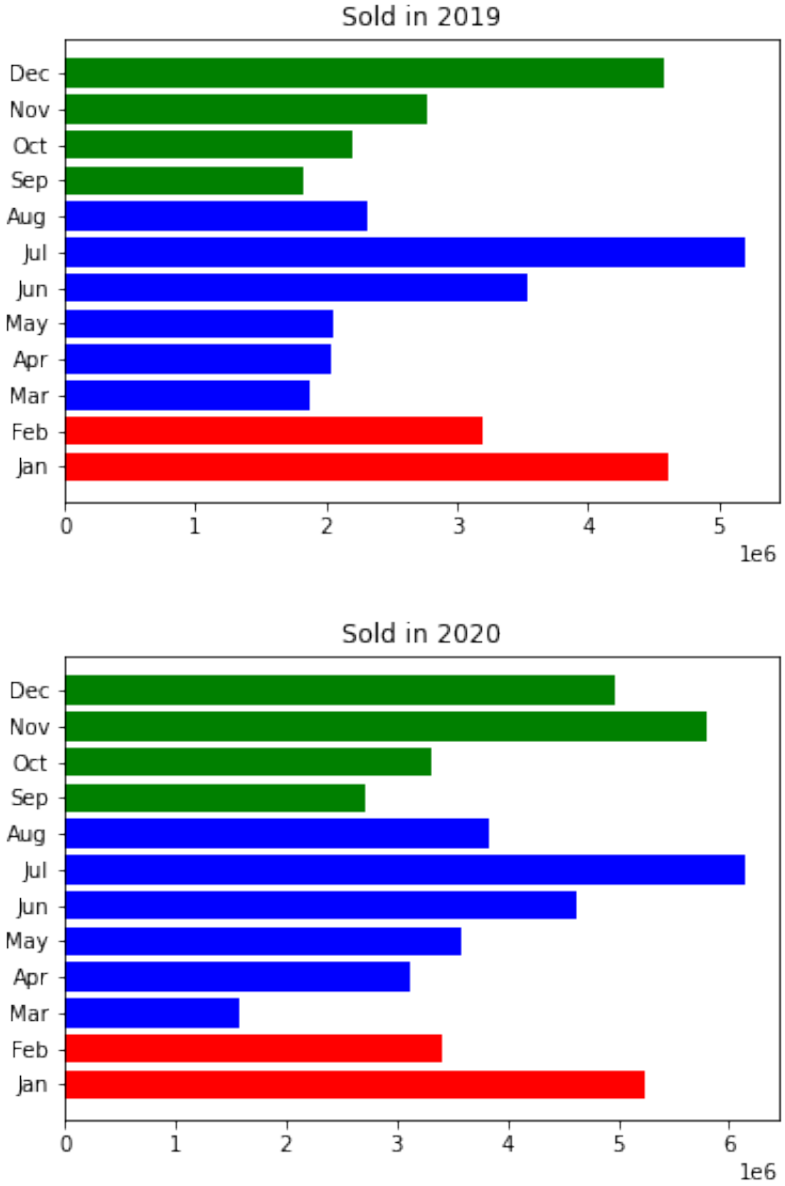

Figure 8: Season money earned in 2019 and 2020

Product size

Looking at the figures [9] and [10], it could be seen that for each year we had similar distributions, being a Gaussian Distribution. The size range from 38-44 was the center of the distribution in all cases.

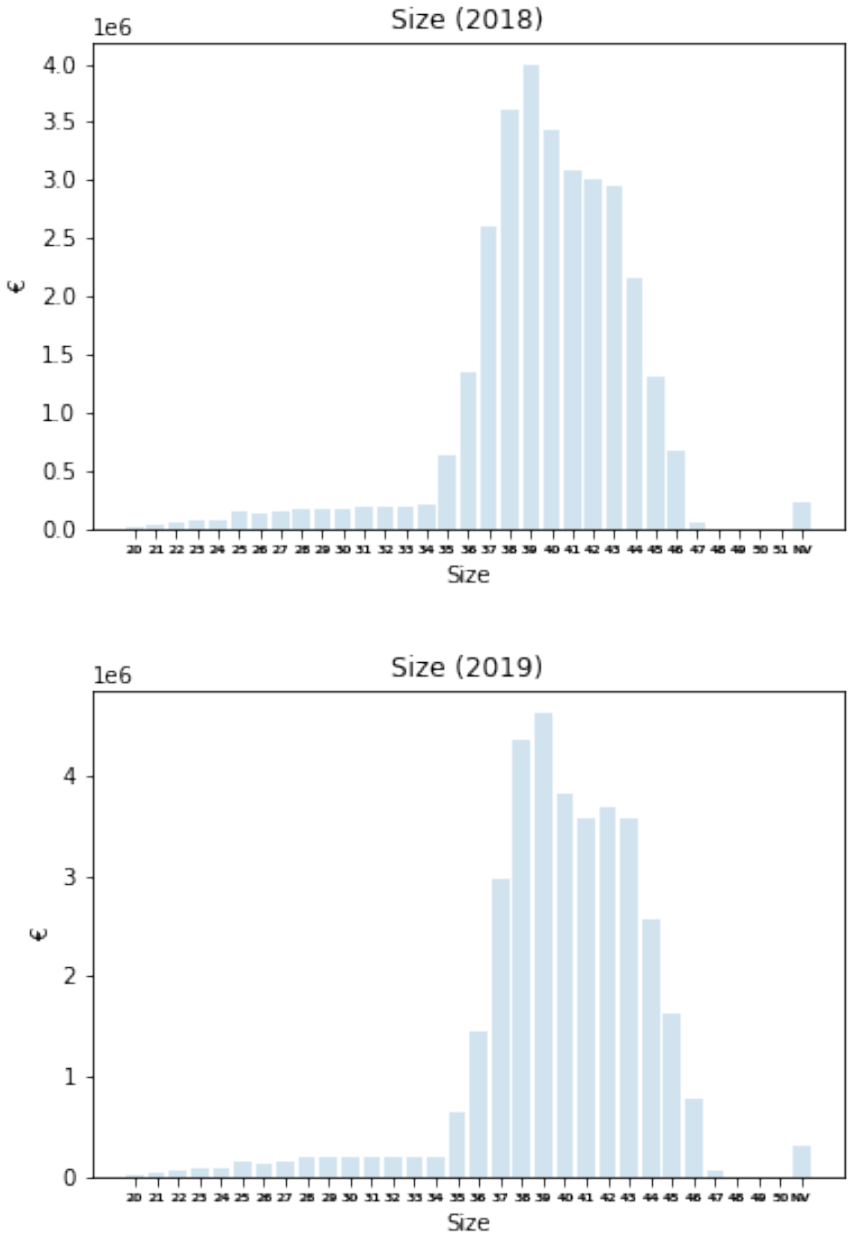

Figure 9: Size sales distribution in 2018 and 2019

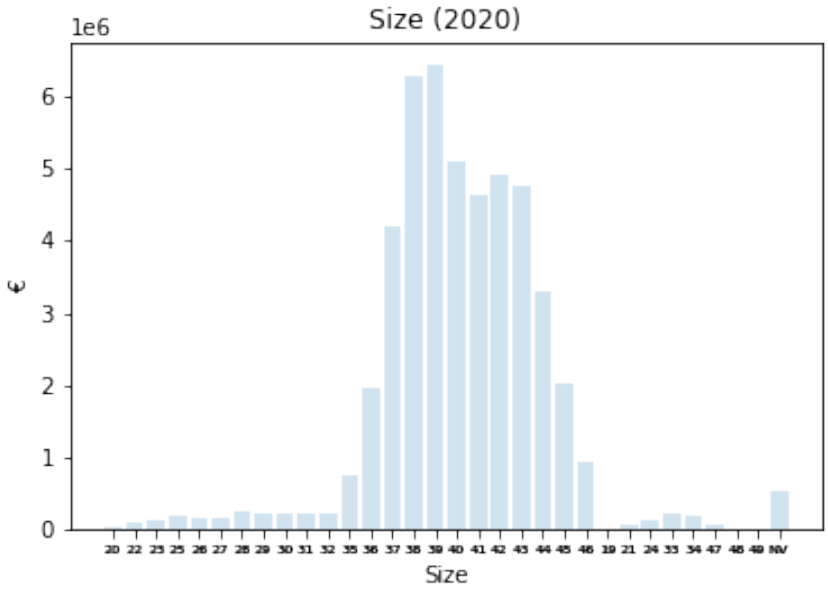

Figure 10: Size sales distribution in 2020

Product gender

Looking at the pie charts in figure [11] and [12], women products were the most sold, around 50% from the total, men were the following ones with more or less 40% and kids' category has been decreasing year by year. Kids' products were the worst in terms of sales. We conclude that the market should be oriented towards men and women. Also, we could extract from the data that one open market to exploit is Kids.
It could be seen that the unknown data in terms of product gender was insignificant.

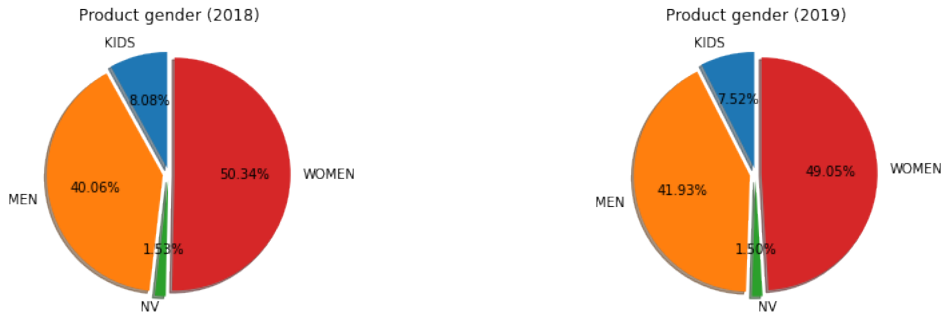

Figure 11: Product gender sales (2018 and 2019)

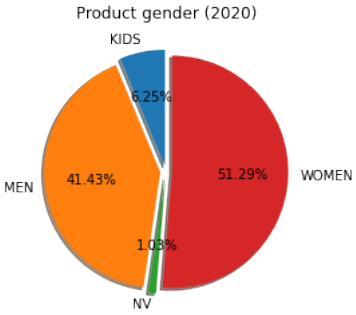

Figure 12: Product gender sales (2020)

Analysing this field more deeply [13], in all cases, Autumn - Winter season was the one with more contribution in each category. As a reminder, Autumn - Winter (odd numbers) and Spring - Summer (even numbers).

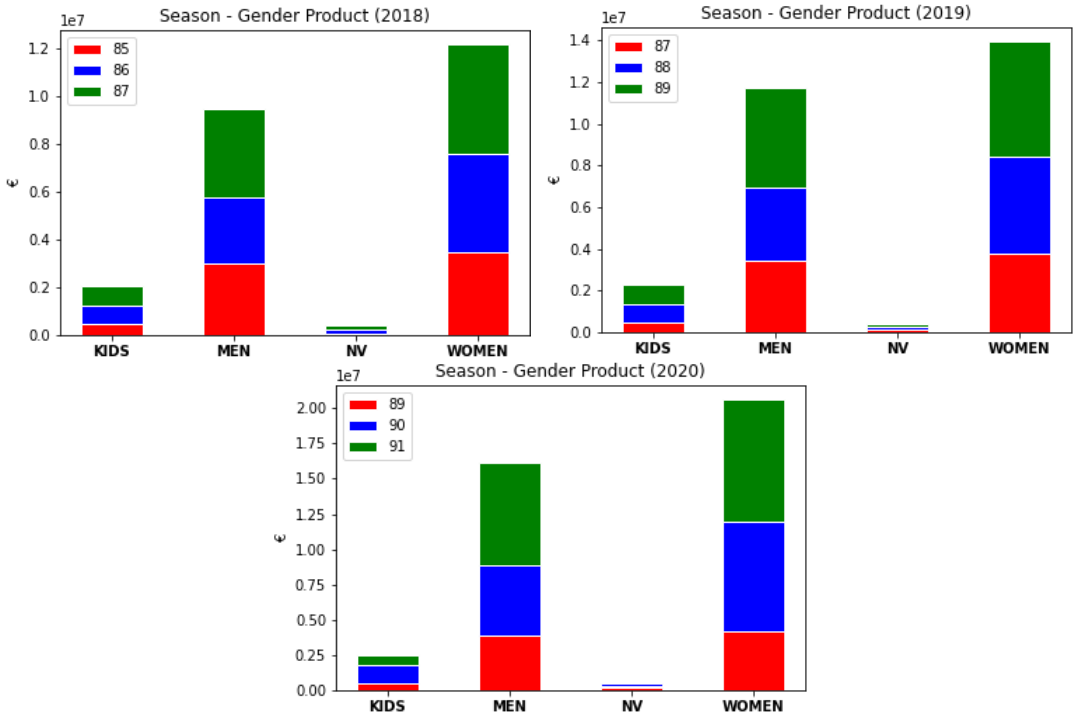

Figure 13: Seasons by gender product (2018, 2019 and 2020)

Age customer

The next step is to calculate the average customer age. To do it, we are going to consider only the informed data.

Average customer age (2018): 44.0 years old

Average customer age (2019): 45.0 years old

Average customer age (2020): 45.0 years old

We had almost the same average in each year.

In the figure [14], similar to the feature size, age also follows a normal distribution without considering the non-informed values (NV), having their center at range 40-49. So with the data given and without paying attention to the NV', the market should be oriented for people from 30 to 59 years old. This actually could be seen as an opportunity to invest in increasing the sales in other age ranges which were not favourable.

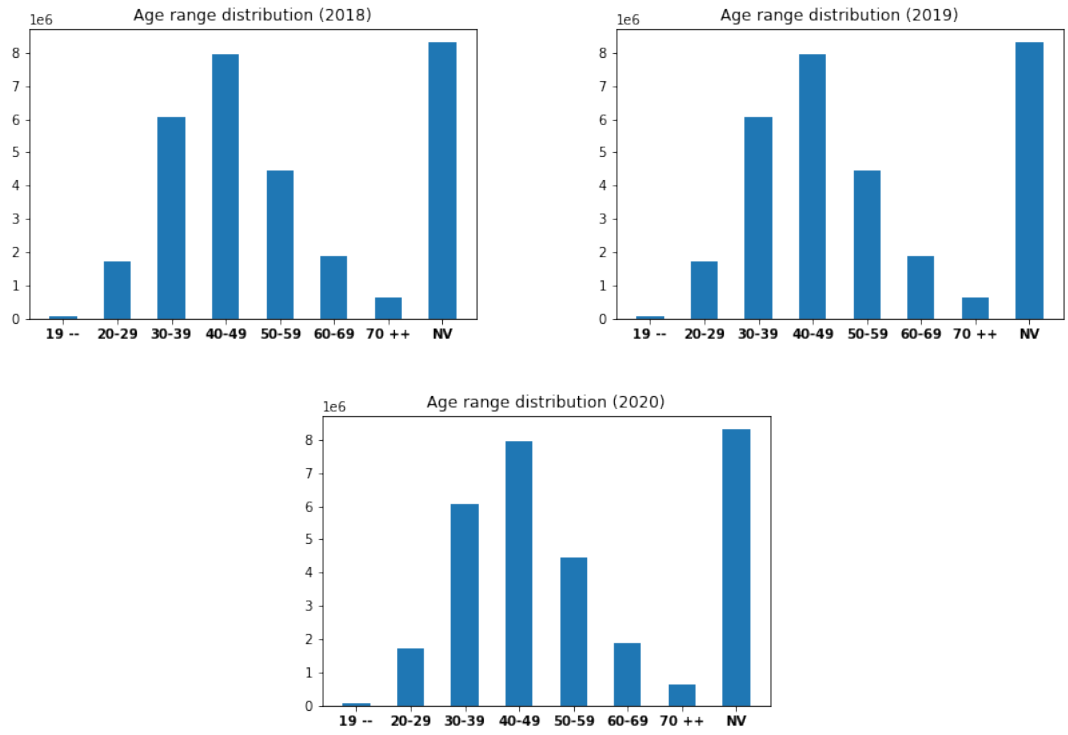

Figure 14: Range ages distribution (2018, 2019 and 2020)

Country profit

We are going to analyse the sales across all the countries where Camper sold its products.

Top 10 countries by year					
2018		2019		2020	
Country	€	Country	€	Country	€
US	5748569.97	US	6133213.07	US	6580937.33
DE	4175764.93	ES	4627414.71	ES	5840923.51
GB	3937674.21	DE	4603193.30	DE	5716981.59
ES	3433616.80	GB	4587061.22	GB	5203238.93
FR	1562331.39	FR	1920651.99	TR	2670125.12
IT	1548451.73	IT	1784742.96	JP	2662741.23
AU	1437315.65	JP	1361503.15	FR	2531096.81
JP	1259697.45	AU	1267920.77	IT	2498473.55
CA	998592.72	TR	1079165.43	AU	1725299.29
TR	960326.27	CA	1009711.97	CA	1615978.43

Table 7: Country sales

The top ten countries with the highest sales did not change throughout the years. Always the same countries but different positions in the ranking.

Filtering and ordering in descending order the countries by the amount of money earned by each product gender, we got the results reported in table [8].

Country	Product gender
US	WOMEN
DE	WOMEN
US	MEN
GB	WOMEN
ES	WOMEN
DE	MEN
AU	WOMEN
FR	WOMEN

Table 8: Most sold product gender in each country

The most dominant gender in almost all countries is women. Only US and DE had the enough amount of earning with the product gender Men, to be before others in the list, but in any case, their amount of earnings was less than their product gender women.

Language customer

The non-informed values in each year was 0.357% (2018), 0.292% (2019) and 0.221% (2020).

- The three most used languages in 2018 were:
  1. English
  2. German
  3. Spanish
- The three most used languages in 2019 and 2020 were:
  1. English
  2. Spanish
  3. German

In each year the top three most used languages are the same.

Weekly earnings

Looking at the results from figures [15] and [16], Monday was the day where people bought more products. This was an unexpected result for us. The reasonable thing to happen might have been that Saturday was the day of the highest sales. Otherwise, being Sunday the lowest day in each year looked feasible. In each year, they followed a decreasing and descending order.

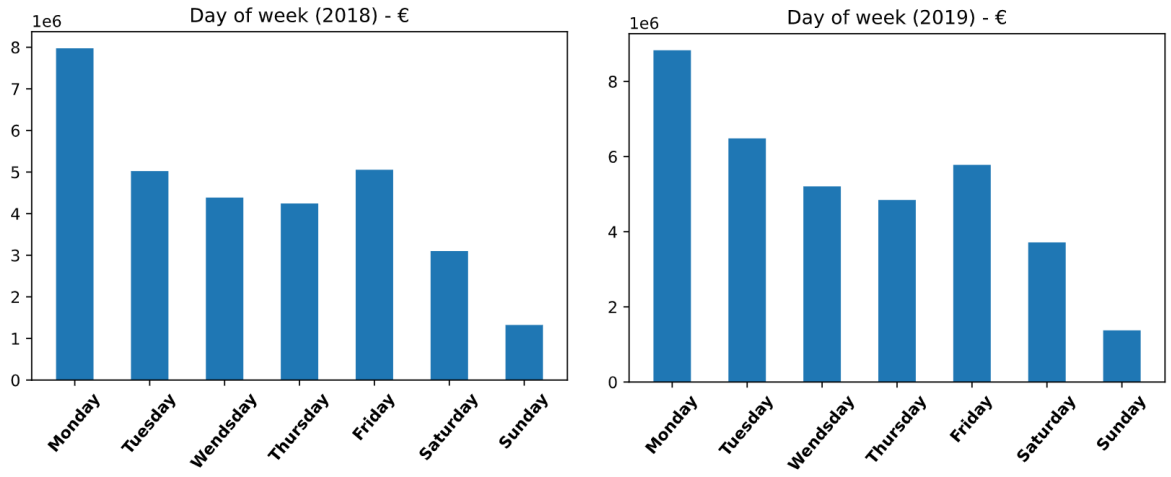

Figure 15: Weekly earnings 2018 and 2019

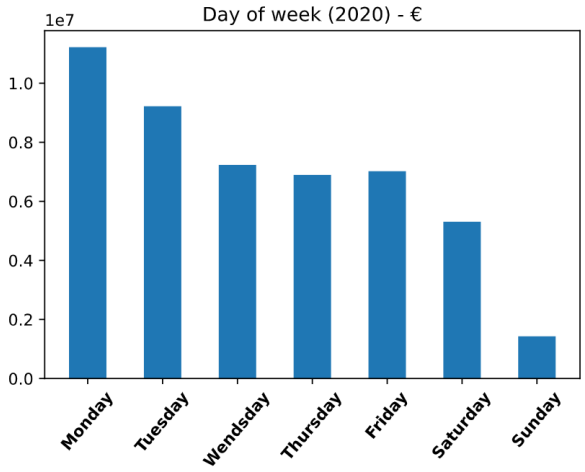

Figure 16: Weekly earnings 2020

3.3.5 Principal Component Analysis

Principal component Analysis (PCA) is used in order to reduce dimensionality, see the correlations among the variables, determine what features are more important and check if they cluster together.
The first step was to check the proportion of variance explained by each principal component (26 in total).

1. 1.54798916e-01
2. 1.19888398e-01
3. 1.12454756e-01
4. 9.86678817e-02
5. 7.43469948e-02
6. 5.23519217e-02
7. 4.77960572e-02
8. 4.35135247e-02
9. 4.21125379e-02
10. 4.07680860e-02
11. 4.03992169e-02
12. 3.68332989e-02
13. 3.22840354e-02
14. 2.81336821e-02
15. 2.70243288e-02
16. 1.92088221e-02
17. 8.10792321e-03
18. 6.96706479e-03
19. 5.05093749e-03
20. 3.28300431e-03
21. 1.99078839e-03
22. 1.73522779e-03
23. 1.30214077e-03
24. 5.03762168e-04
25. 3.90386595e-04
26. 8.63067643e-05

However this did not give us relevant information at all, what we could extract from these numbers was the cumulative proportion of variance, and from there, it could be seen from what principal component did not add valuable information.

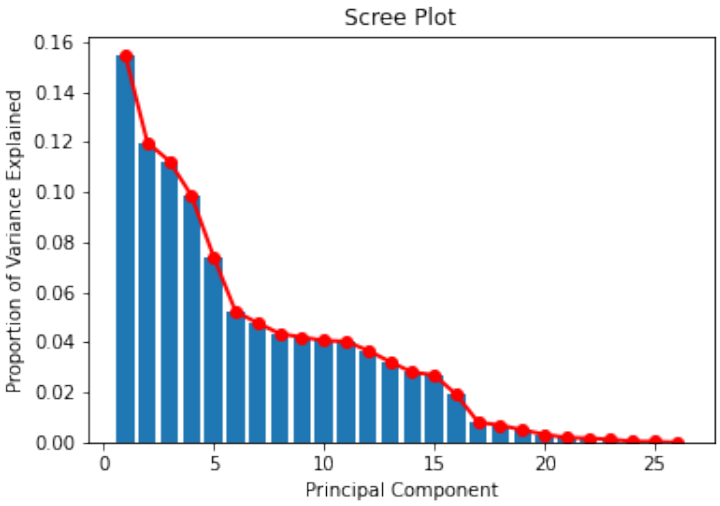

Figure 17: Scree plot Principal Component

Applying the *Last Elbow Rule* [17], we could determine the number of principal components that described the dataset are 16. This meant that more principal components would not give us relevant information.

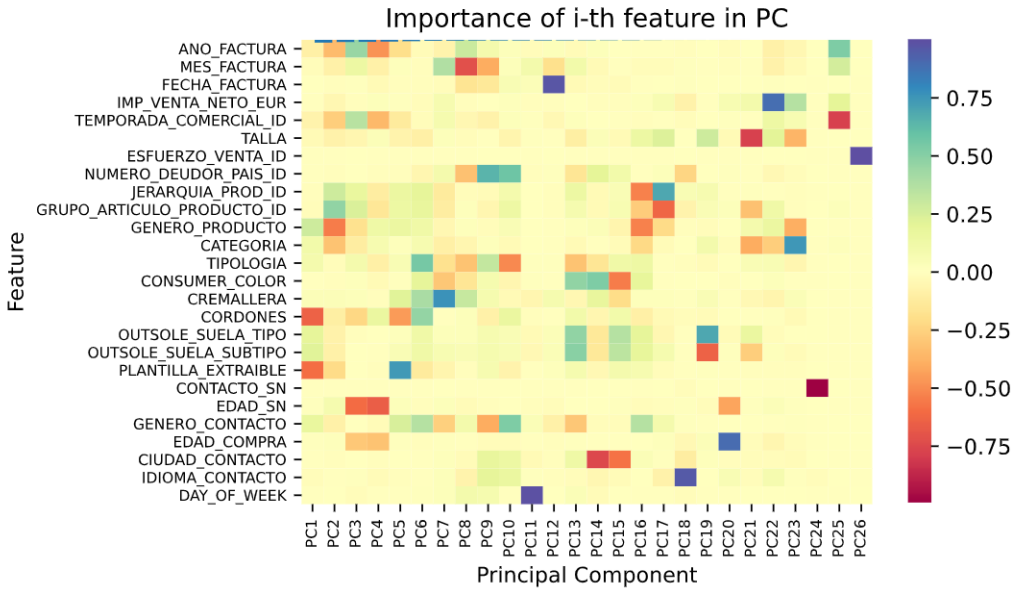

Figure 18: Importance i-th feature in each Principal Component

From figure [18], we could extract which features had more importance in each principal component. In the 1st, 6th, 7th 13th, 14th and 15th principal component, the features related to the product itself had the most contribution. On the other hand, the features related to the customer did not have the same importance at the same time in any principal component. For instance, the gender of the customer was only important in the principal component from 5th to 10th, 14th and 16th and the City was only in 14th and 15th PC.
Checking the features related to the date, we could see that the results are the same as the ones related to the customer, they did not have the most contribution at the same principal component.

To plot the features' contribution from the first two principal components, we have defined the following labels:

0. ANO_FACTURA
1. MES_FACTURA
2. FECHA_FACTURA
3. IMP_VENTA_NETO_EUR
4. TEMPORADA_COMERCIAL_ID
5. TALLA
6. ESFUERZO_VENTA_ID
7. NUMERO_DEUDOR_PAIS_ID
8. JERARQUIA_PROD_ID
9. GRUPO_ARTICULO_PRODUCTO_ID
10. GENERO_PRODUCTO
11. CATEGORIA
12. TIPOLOGIA
13. CONSUMER_COLOR
14. CREMALLERA
15. CORDONES
16. OUTSOLE_SUELA_TIPO
17. OUTSOLE_SUELA_SUBTIPO
18. PLANTILLA_EXTRAIBLE
19. CONTACTO_SN
20. EDAD_SN
21. GENERO_CONTACTO
22. EDAD_COMPRA
23. EDAD_RANGO_COMPRA
24. CIUDAD_CONTACTO
25. IDIOMA_CONTACTO
26. DAY_OF_WEEK

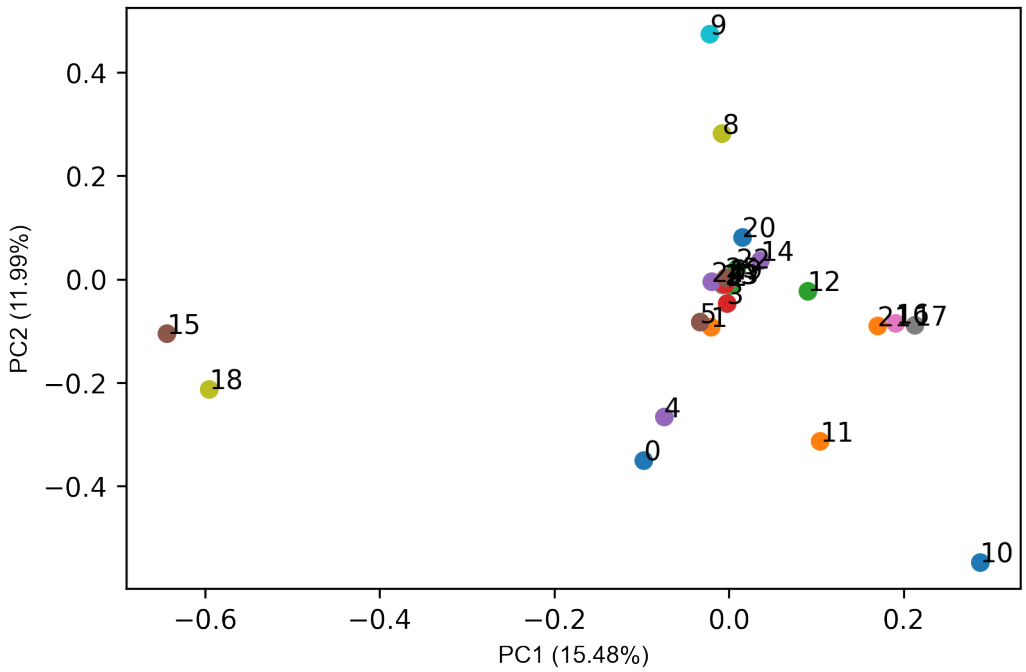

Figure 19: Features' contribution distribution

In the figure [20], we could see the representation of individuals and features components. We had several components correlated among them where we drew a circle.

Feature 8 and 9 (circle red)
These two features were JERARQUIA_PROD_ID and GRUPO_ARTICULO_PRODUCTO_ID. Basically they were codes to represent the product so it was normal that they were correlated as they represented similar information. Also, they had more importance in the second principal component.
Moreover, they were negatively correlated with the features 10 and 11 (GENERO_PRODUCTO and CATEGORY.

Feature 0 and 4 (circle green)
These two features were ANO_FACTURA and TEMPORADA_COMERCIAL_ID as season directly depends on the year, it was obvious that they were going to be correlated. In this case, they had also more importance in the second principal component.

Feature 10 and 11 (circle blue)
These two features were GENERO_PRODUCTO and CATEGORY. As we mentioned before, they were negatively correlated with 8 and 9. GENERO_PRODUCTO was one of the most influential features in these principal components and affected negatively to the second principal component.

Feature 15 and 18 (circle yellow)
These features were CORDONES and PLANTILLA_EXTRAIBLE, these two were the only ones which had a real influence on the first principal component. They were not

correlated with any other feature.

Non important features (circle white)
These features did not have enough influence in any of these principal components, and consequently they appeared close to the center.

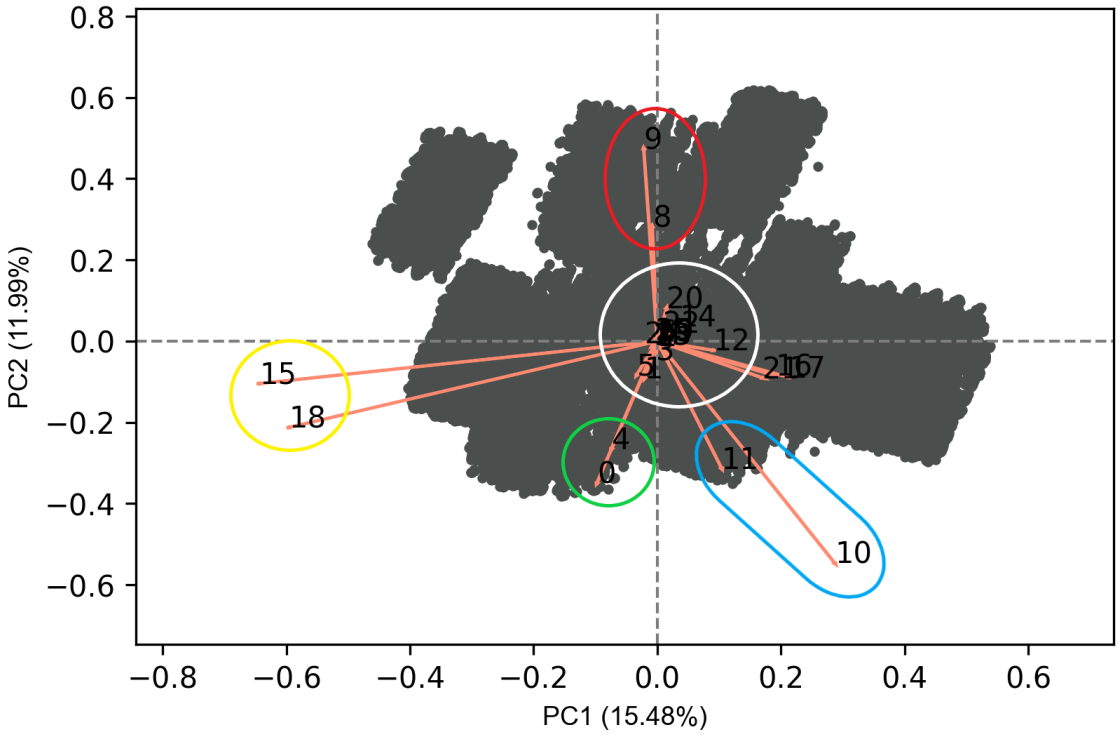

Figure 20: Biplot

3.3.6 Customer segmentation

Customer segmentation is the division of the customers into groups based on common characteristics. The goal of segmentation is to allow marketers to identify how related to each other are customers in the discrete groups formed, in order to maximize the value of each customer to the business.

The main goal was to create discrete groups of customers that would allow us to distinguish the different kinds of customers and what they bought, so given a certain product, we would be able to know what kind of customer is more likely to buy it.

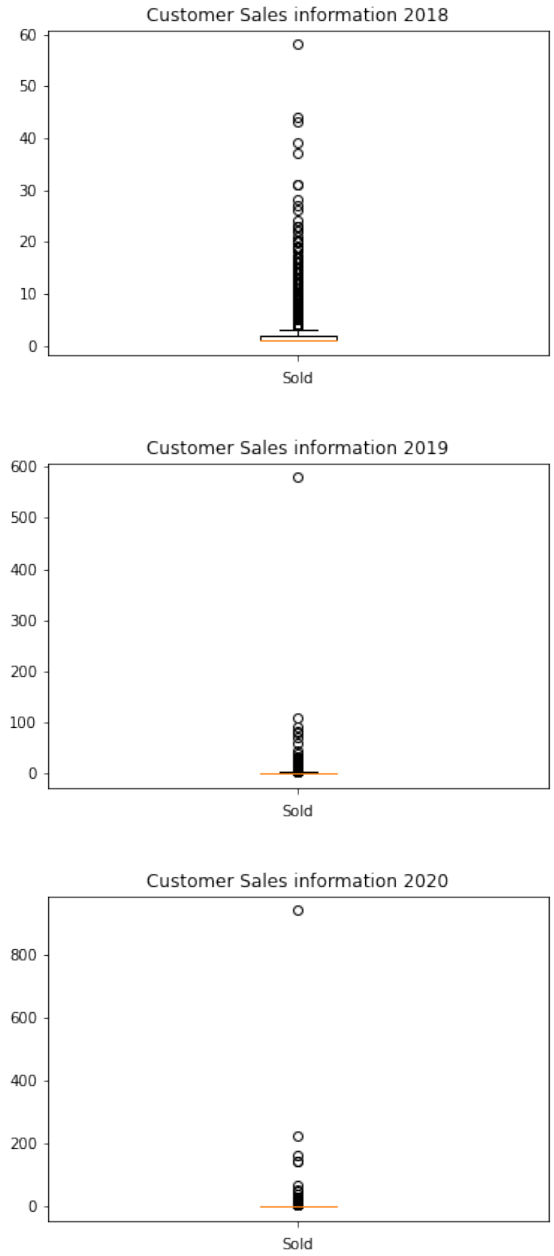

Figure 21: Customer sales information 2018, 2019 and 2020

In figure [21], we could observe the distribution of the customers data. What could be extracted from these pictures was that there were at least two groups of customers, those who bought more than 100/120 units and those who did not.

What we wanted to know is from which amount of sales, it was approximately constant so we would find out a possible customer pattern.

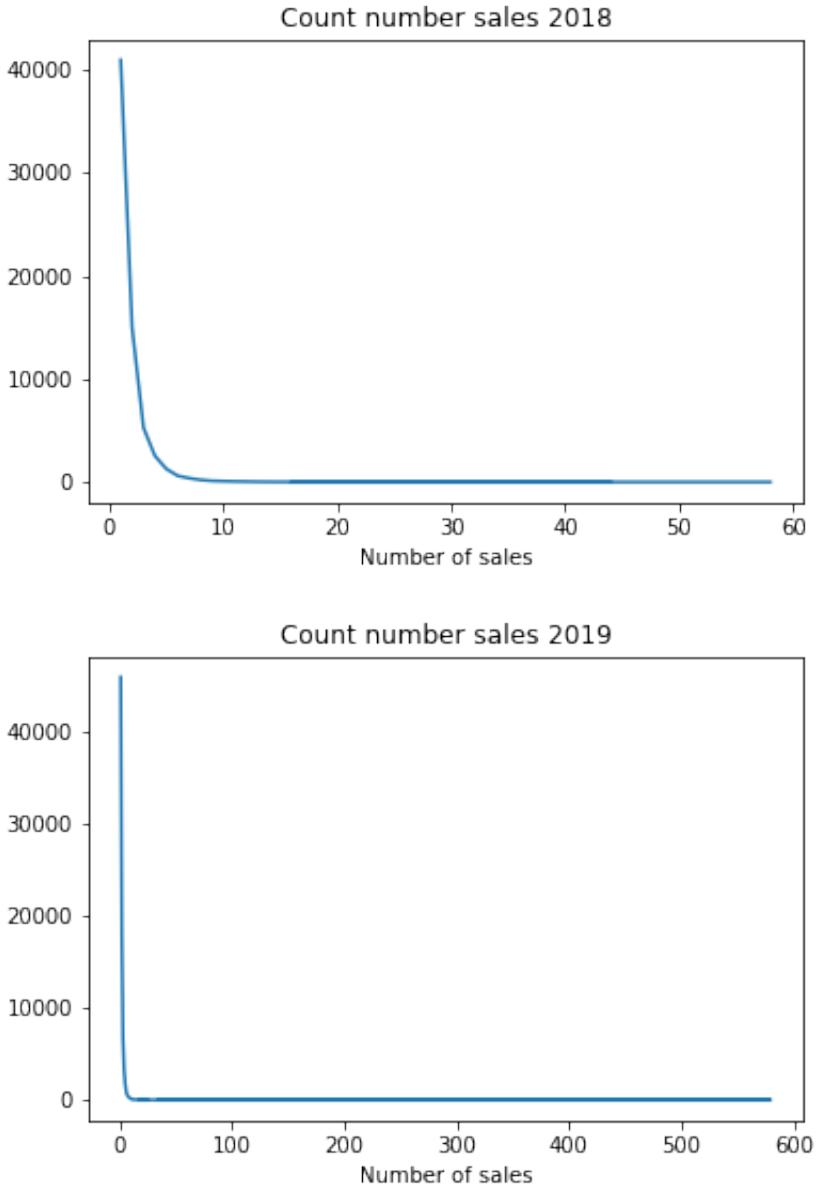

Figure 22: Number sales (2018 and 2019)

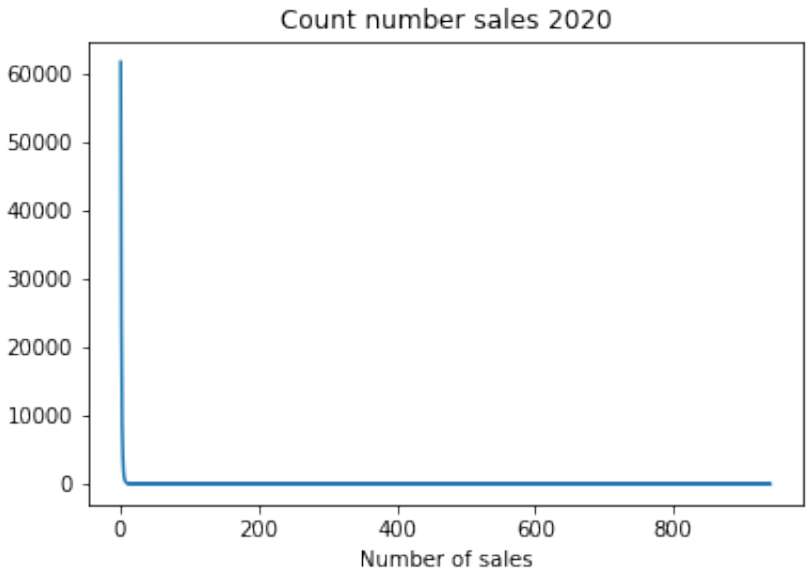

Figure 23: Number sales (2020)

We performed a division of the data into four defined intervals based on this information and how the values are compared to the full set of the observations.

From the quartile information of table [9], we could extract the relevant information in each division and this gave us an idea of how we could divide the data.

Quartile information			
	2018	2019	2020
min	1	1	1
25%	1	1	1
50%	8	7	9.5
75%	115	100.5	120.5

Table 9: Quartile information in 2018, 2019 and 2020

Moreover, zooming the previous pictures (figures [22] and [23]), we could see that there are variations until 7 to 9 amount of products and from there it was constant. That was what we were looking for.

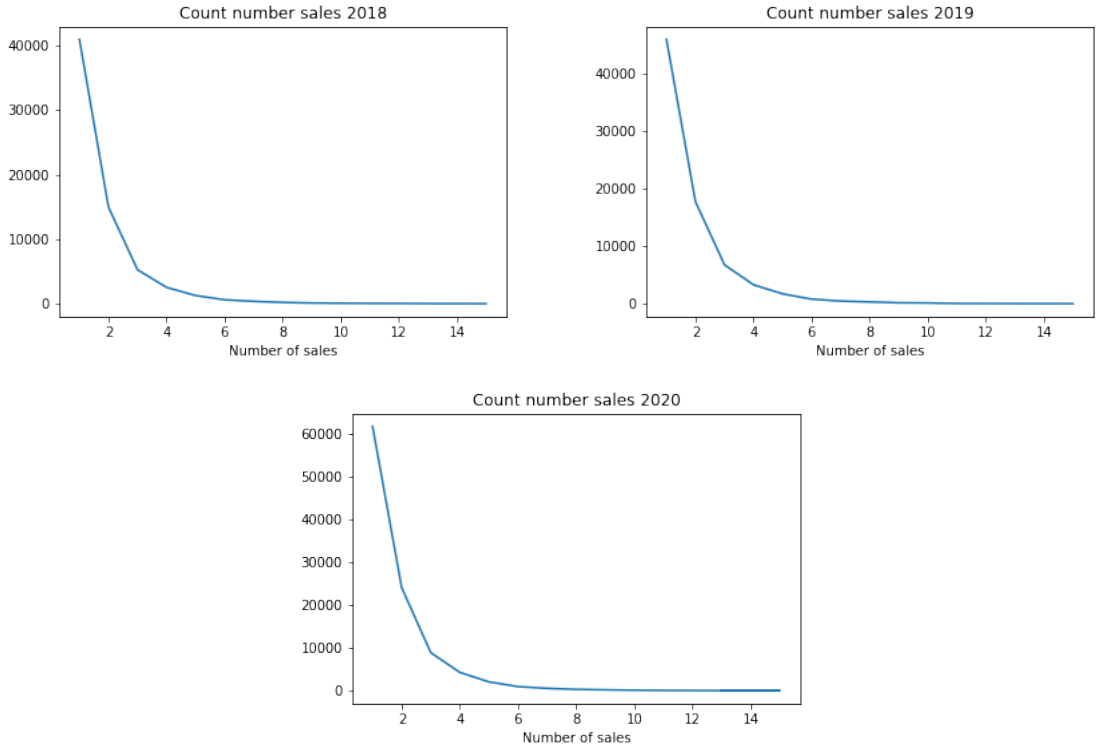

Figure 24: Zoomed number of sales (2019, 2019 and 2020)

Finally, with all the information aforementioned, we could divide the customers regarding their buying history, into the following categories:

0. Regular (1 - 9 items)
1. Confident (9 - 120 items)
2. Reliable (> 120 items)

Once we had our data labelled, we could proceed to calculate the propensity score of buying a product each type of customer.

3.3.7 Propensity scoring

Propensity score is a set of different techniques which allows you to estimate the likelihood that customers will perform certain types of actions.
There are different methods to calculate it, in this case, two methods have been performed:

- **Logistic Regression**, predict the probability that an event occurs
- **Inverse-probability treatment weighting** (IPTW), the main idea of this technique is weighting the observations in order to assign the customer to a particular treatment group

The approach followed to calculate the propensity scores was extracted from the paper written by **Tim Royston-Webb** [14]

Approach:

1. Select variables to use as features
2. Build a model and prepare data
3. Calculate propensity scores (Logistic Regression/IPTW)
4. Using the model for causal inference.

Customer profile features:

- CUSTOMER_ID
- ANO_FACTURA
- GENERO_CONTACTO
- EDAD_COMPRA
- CIUDAD_CONTACTO
- IDIOMA_CONTACTO
- NUMERO_DEUDOR_PAIS_ID

Using the features above, we will be able to code the data. After, we will calculate the propensity scores with the two techniques mentioned using the information from that features. Once this is done, the next step is to select which features are important in order to create a profile for products.

Product features:

- PRODUCTO_ID
- TALLA
- GENERO_CONTACTO
- JERARQUIA_PROD_ID
- GENERO_PRODUCTO
- CATEGORIA
- TIPOLOGIA
- CONSUMER_COLOR
- CREMALLERA
- CORDONES
- OUTSOLE_SUELA_TIPO
- OUTSOLE_SUELA_SUBTIPO
- PLANTILLA_EXTRAIBLE

Finally, three models have been implemented with the aim of comparing the benefits of each propensity score method. Random forest was the model chosen to use. Moreover, all models were implemented with the same hyper parameterization.

Results obtained		
Model	**RMSE**	**Accuracy (%)**
Random Forest	0.25	95.95
Random Forest + IPTW	0.26	95.67
Random Forest + Prob.	0.25	95.96

Table 10: Customer segmentation model results

Looking at table [10], we could extract that the use of the propensity score was not as determinant as we thought while implementing it. Although the model that used the probabilities calculated from logistic regression was the one with more accuracy, it only differed a 0.01% regarding the model without them. It could be observed that the RMSE also differed 0.01% among them.

However, we observe that although for this particular case, the propensity scores methods were not determinant to predict the labelled data, and the customer segmentation could be done without these techniques. We have produced a systematic way of determining where a new product could get adopted in the population. This means that if Camper designs a new product, given its characteristics, we are able to predict what type of customer is more susceptible to buy it.

3.4 Modelling methods

3.4.1 Multiple Linear Regression

Multiple linear regression is an extension of linear regression method. The technique attempts to model the relationship between two or more explanatory variables depending on a response variable.
This model is defined by the next linear equation:

$$y = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + ... + \beta_i x_i + \epsilon$$

Every value of the independent variable **x** is related to a value from the response variable **y**.
β_0 represents the predicted value when **x** is equal to 0. It is constant
β_i are the regression coefficients which represents the average predicted value changed in **y**.
ϵ is the residual terms of the model

3.4.2 Extreme Gradient Boost

Extreme Gradient Boost (XGBoost) is a decision-tree-based ensemble machine learning algorithm that uses a gradient boosted decision tree.
Briefly explained, a gradient boosted decision tree is a machine learning technique which generates a prediction model in a form of ensemble of decision trees. This allows that in each iteration of the decision-making, all the parametric values can be adjusted to minimize the loss function.
XGBoost does not explore all possible tree structures, it builds a tree greedily. Also, the regularization term penalizes building complexes trees with several leaf nodes.

3.4.3 Multi-layer Perceptron Regression

Multi-layer Perceptron Regression (MLP), is a class of artificial neural network (ANN). It is a supervised machine learning method that consist of three layer nodes.

- Input layer

  As the name indicates, takes the input from the dataset and passes the information to the next layer. The input has to be numerical.

- Hidden layer

  It receives the data from the input layer. It contains a specific number of neurons which perform the calculations to solve the problem.

- Output layer

  This is the final layer which is responsible for getting the output value or values that correspond to the solution of the original problem.

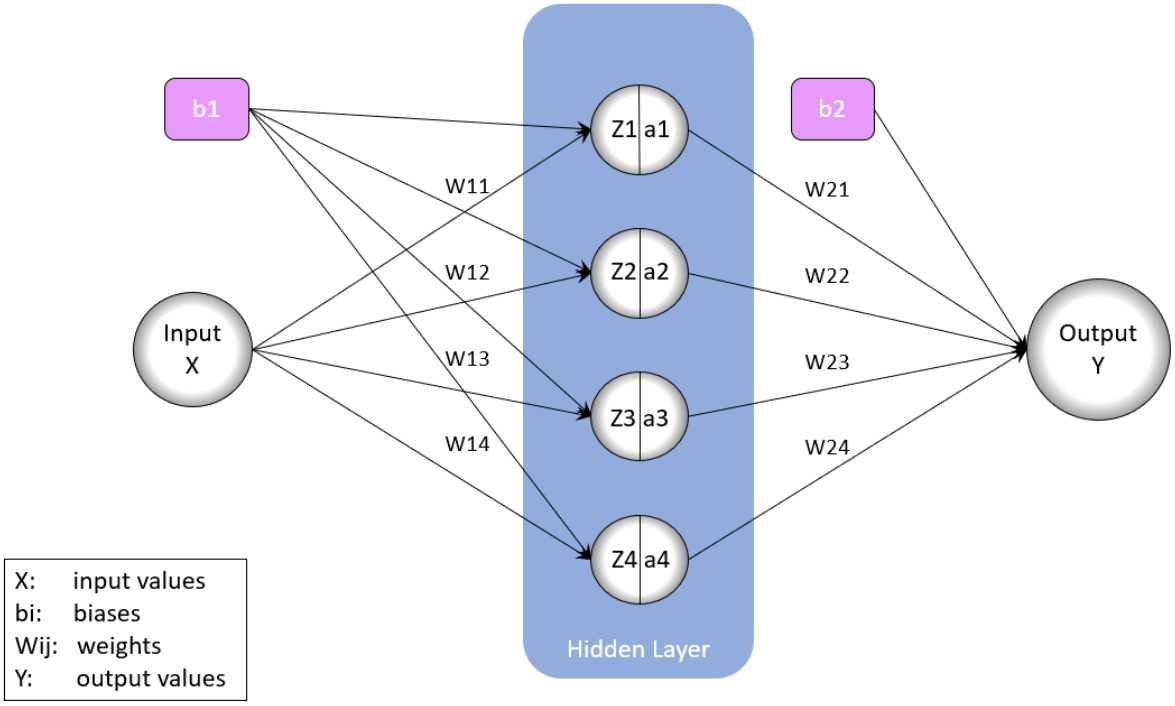

Figure 25: MLP scheme [15]

3.4.4 Long Short-Term Memory

Long Short-Term Memory (LSTM) is a type of recurrent neural network (RNN) designed to avoid long-term dependencies problems.
As with **MLP**, this neural network follows a chain structure but instead of having a single neural network like **MLP**, a **LSTM** unit is commonly composed of a cell, an input, an output and a forget gate.

The usual architecture is composed by 2 or 3 layers, and each layer computes the following:

$$i_t = \sigma(W_{ii}x_t + b_{ii} + W_{hi}h_{t-1} + b_{hi})$$
$$f_t = \sigma(W_{if}x_t + b_{if} + W_{hf}h_{t-1} + b_{hf}$$
$$g_t = tanh(W_{ig}x_t + b_{ig} + W_{hg}h_{t-1} + b_{hg}$$
$$o_t = \sigma(W_{io}x_t + b_{io} + W_{ho}h_{t-1} + b_{ho}$$
$$c_t = f_t \odot c_{t-1} + i_t \odot g_t$$
$$h_t = o_t \odot tanh(c_t)$$

where h_t is the hidden state at time t

c_t is the cell state at time t

h_{t-1} is the hidden state of the layer at a time $t-1$

i_t is the input gate

f_t is the forget gate

g_t is the cell gate

o_t is the output gate

σ is the sigmoid function

$\odot$ is the Hadamard product

We are going to use the machine learning framework called **PyTorch** [16] to implement our two layer **LSTM** model.

3.4.5 Gaussian Process Regression

Gaussian process regression models are non-parametric kernel-based probabilistic models. This algorithm requires specifying a kernel function which controls how related data points pairwise are, defining the covariance function of the data. This algorithm is defined by a mean vector μ and a covariance matrix $\mathbf{K}$.
Each component of the vector represents the mean of its corresponding dimension in the feature space. Otherwise, $\mathbf{K}$ defines the covariance among dimensions and determines how different random variables are correlated among them. This matrix has to be always **Symmertric** and **Positive Semi-Definit (PSD)**

Unfortunately, applying the Gaussian process regression approach to a data set with size **n**, is well-known that it has a time complexity of $O(n^3)$ and with a storage complexity of $O(n^2)$. For further information about the time and storage complexity, have a look at [2].
In our case of study, we cannot apply **GPR** directly to our dataset because we will have storage problems, considering that our dataset has approximately 1 million rows and we divide it into training (60%) and testing (40%).

- $n = 1000000$
- $n_{train} = 641.638$
- $n_{test} = 427.760$

Calculating the complexity, we would obtain the following results:

- Time: $O(n^3_{train}) = 264.161.930.239.306.072$

- Storage: $O(n_{train}^2) = 411.699.323.044$

Assuming that for every complex operation our computer spends 0.001 seconds, it would take 264.161.930.239.306,072 seconds (approx. 8.376.519,86 years) to finish all the calculations. Supposing that each row takes 1KB, the memory that we should use for it would be 383.42 TB.
As we have hardware and time limitations, we are not able to perform it. Otherwise, we have created a function in order to know what the amount of data that we can use avoiding a memory size error is. The only percentage of training data we can use without getting memory error is the 0.01%, and from this, divide it into training and testing. Basically, we should fit the model only using 4.478 registers.
With the aforementioned, we have discarded applying this method to our study.

3.4.6 Hybrid approach

This approach is based on a combination of some of the models aforementioned using ensemble methods. We have chosen to use Model Stacking because it is an efficient ensemble method in which the predictions are generated by using different machine learning algorithms.

Basically, this approach will have two layers:

- Model layer

  This will have as an input data the training set generated

- Meta-Regressor layer

  This will have as an input the results generated by the previous layer, and it will generate the final predictions.

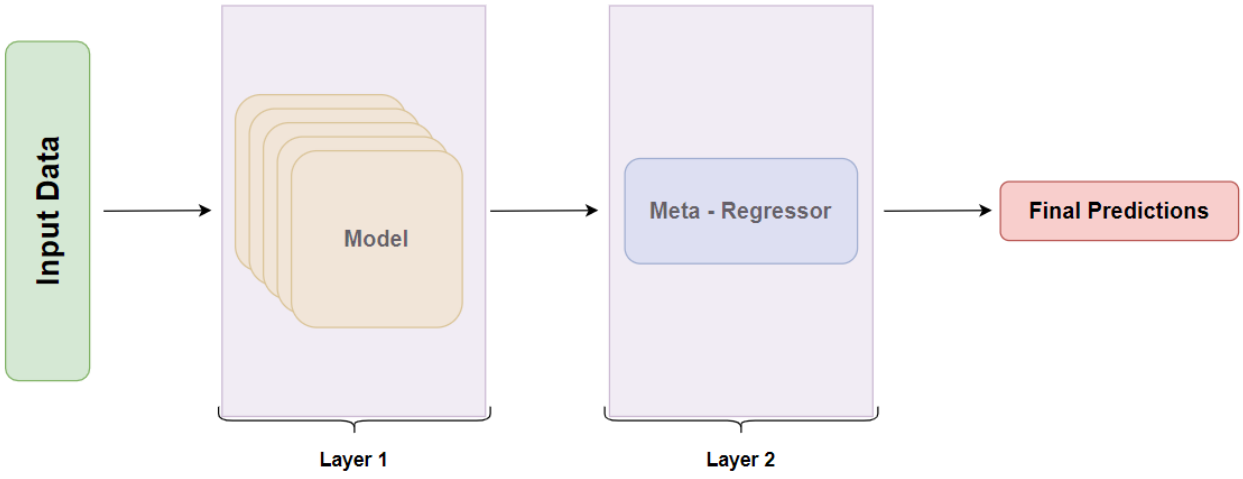

Figure 26: Hybrid approach scheme

The models that we are going to use in the Model layer are **LSTM** and **XGBoost**. Each model will have different initial training data, with m observations and n & p features (so each model will be $m \times n$ and $m \times p$ respectively).
Then, the models will provide predictions for the outcome (y) which will be then cast into the meta-regressor as an input training data. So now, the dimensions are going to be $m \times M$ (being $M = 2$). Namely, the M predictions will become features.

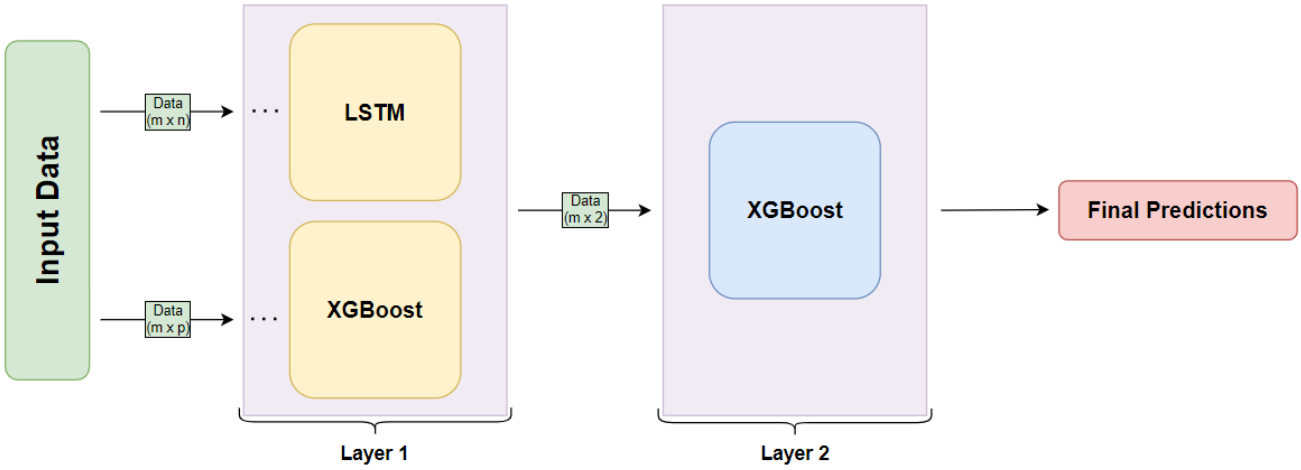

Figure 27: Hybrid final approach scheme

3.5 Hyperparameter and model optimization

Hyperparameter optimization refers to tune objectively different parameters from the machine learning models which are not updated during the learning process. Those parameters are called hyperparameters.
The aim of this optimization is to get the model that achieves the best performance results given a specific dataset. In other words, it is the configuration of the model to get the best possible result. Otherwise, choosing the wrong hyperparameters could result into a poor performance, or vulnerabilities thus underfitting or overfitting.

3.5.1 Grid search

Grid search is a hyperparameter technique that computes the optimum values in an exhaustive search way. Basically, given a set of values for the model's hyperparameters, this method tries all the possible combinations among them. Then, it will have a high computation complexity if the amount of hyperparameters and their number of values increase.
For the evaluation of the best parameterization, we will use the coefficient R^2 and the resampling technique called **Cross Validation**, more concretely, **5-CV**.

3.5.2 Random Search

Random Search is a hyperparameter technique that computes the optimum values using random combinations of the hyperparameters. So, at each instance, the technique selects randomly a combination of parameters and compute them to find the solution for the built model.
Oppositely with Grid Search, not all the values of the parameters are tried out so it can lead to have high variance during the computations.
This technique is very useful when there are a lot of values for each hyperparameter and we want to reduce them.

3.5.3 Recursive Feature Elimination

Recursive Feature Elimination (RFE) is a wrapper type feature algorithm for selecting features. Given a model, RFE selects the features recursively considering smaller sets of them iteratively, removing the no significant features.
Fundamentally, this algorithm removes all unnecessary features from the datasets for a given model. Then, the dimensionality of the dataset is reduced so it decreases the complexity of the model and improves the performance.

3.6 Explainability layer

Nowadays, machine learning models are similar to black boxes, we know what algorithm we are applying and we trust the reasons why the algorithm has made a certain decision and ignore another. In most cases, as long as the machine learning model gives good results, the understanding of the reason does not matter, but in other cases, knowing the *why* can help to understand about the problems and the reasons of an unsuccessful model.

The aim of this layer is to bring light to these machine learning models and solve this problem. This layer will have the ability to explain the reasons behind the decisions made by all the models we have so far discussed. Also, we will bring all the model results at a level at which business people can understand the explanation.
Essentially, we are going to extend this cause and effect observed in the system and put it in a way to understand the intern mechanisms of the machine learning models in a human being form.

To do this, we have done a review of the literature and we have selected some algorithms to add to our system. Our main goal is to understand a certain model, and extract business information which could help Camper in the decision-making phase and how to take advantage of it.

3.6.1 Shapley Value

Shapley Value is a formal rule for predicting how a game will be played in cooperative game theory. The main idea is to calculate for each player (feature) the average marginal contribution across all possible coalitions among them.
So considering we have three features for one product, colour, laces (yes or not) and shape (boot, sandal), and we want to analyse what feature is more important for to sell. Using this technique, we have to calculate all possible coalitions:
(For instance, our target is colour = black)

- No features
- laces = yes
- laces = no
- laces = yes + shape = sandal
- laces = no + shape = sandal
- laces = yes + shape = boot
- laces = no + shape = boot
- shape = boot
- shape = sandal

For each of these combinations, it is evaluated with and without the colour = black. We get the marginal contribution and calculate the average to get the final contribution of the feature. This procedure is done with all the features.

We are going to apply this method to our **Multiple Linear Regression** and **Extreme Gradient Boost**. The implementation source is from the package **DALEX** [19]

3.6.2 Local Interpretable Model-Agnostic Explanations

Local Interpretable Model-Agnostic Explanations (LIME) is a technique that explains the predictions of any classifier based on the assumption that every complex model can be described as linear on a local scale.
It can be represented by the next formula:

$$\xi(x) = argmin_{g \in G} L\{f, g, \pi_x\} + \Omega(g)$$

g is defined as the explanation of a model

f is defined as a model denoted by $f : R^d \longrightarrow R$

π_x defines a neighborhood of x

L is defined as locally weighted square loss

$\Omega(g)$ is the complexity term

We can extract the general approach from the paper [9]:

1. Select the neighbourhood from each x_i
2. Calculate the explanations
3. Compute the feature importance
4. Extract the feature weights from the simple model and use these as explanations for the complex models local behavior.

We will apply this method to our **Multi-layer Perception Regression** model. The implementation source is from the package **LIME** [9]

3.6.3 Integrated Gradients

Integrated Gradients is an algorithm that is part of the methods used in Primary Attribution It evaluates the contribution of each input data feature to the model output for model interpretation.
The main goal of this algorithm is to attribute a value for the importance of each input feature from a specific machine learning model, based on the gradients. It is calculated

as the integral of gradients respect to an input.

In this case of study, this method will help us to understand and predict the importance of the input features from the data of the **LSTM** model. The implementation source is from the package **Captum** [21]

3.6.4 Layer Conductance

Layer Conductance is an algorithm that is part of the methods used in Layer Attribution, that evaluates the contribution of each neuron in a given layer to the model output.
This method also uses the integration of the gradients to calculate the importance of each input feature, like the previous method, but in this case, the total conductance of the hidden neuron is calculated by summing all the values over all the input variables.

We are going to apply this method to our **LSTM** model. The implementation source is from the package **Captum** [21]

3.6.5 Neuron Conductance

Neuron Conductance is an algorithm that is part of the methods used in Neuron Attribution. It computes the attribution of each input feature respect to a particular hidden neuron.

We are going to apply this method to our **LSTM** model. The implementation source is from the package **Captum** [21]

3.6.6 General method based on Coalitional Game Theory

This method is based on the paper by Erik Strumbelj and Igor Kononenko [4]. It provides an approximation of the contribution of the i-th features algorithm.

The general approach that takes to achieve the results of the contribution for the features is as follows:

1. Randomly select a permutation of the total features (O)
2. Choose a random instance from testing data (y)
3. Select the features that are before the one you are calculating the contribution (target inclusive) (Pre^i (O))
4. Create a prediction (v_1)
5. Remove the target feature from the data
6. Create a prediction (v_2)
7. Calculate φ from predictions ($\varphi = \varphi + (v_1 - v_2)$)

To sum up, v_1 and v_2 are the classifier's predictions where the first one contains the i-th feature and the second one does not. Therefore, these two predictions only differ in the number of features used to fit the model, so the higher the difference in φ is, the more influence we can find in the model by i-th feature.

In this study, we are going to apply this method to all our models explained except **LSTM**, because it would have a high computational cost and we have hardware limitations.

3.7 Software developed

In this section, we have collected the main functionalities used during the project. The main idea of the software developed is to be able to encapsulate the functionalities and create containers to build them in a cloud-based system.
Implementing the software, we kept in mind that the system must have the ability to scale-up, in order to handle peaks of traffic load. Also, the functionalities were created in a way that we could apply the concept of parallelization at every step of the design. As a consequence, we disassociated the components among them, reducing their dependencies on each other.

However we have created several functionalities to try to automate all the processes during the exploratory data analysis, modelling, explainability, etc. We could not create a fully automated system due to the uncertainty of the data.
Our system is composed of three basic layers, Data layer, Modelling layer and Explainability layer.

Data layer:

- **StandardisingData**

  This functionality is responsible to translate all non-English values from the features to English, and convert the values to the same type. It could not be automated because the system is not able to know which items must be translated and the values of them.

- **Encoder**

  Transforms the data into codes.

- **tagCustomer**

  Generate a label specifying which profile each customer has from the dataset, based on the results from 3.3.6. Even though this function is automated, if the categories from 3.3.6 change, the function must be updated.

- **groupData**

  This function is able to group data given a lists of columns.

- **getISOCountry**

  This function map the Country with the encoded value generated by the functionality Encoder.

- **getValuesFilter**

  Returns the values given features filters.

- **getIndexFilter**

  Returns the indexes given features filters.

- **getValues**

  It is responsible of preparing the data to be used in the Modelling layer.

- **takeVal**

  Select data given a time window.

- **sumPredictions**

  Calculate the cumulative prediction given the same time windows as the functionality takeVal.

Modelling layer:

- **Hyperparameter optimization**

  For each model, we have created different procedures in order to be able to get the best parameterizations. We could not automate these procedures because they depend on the data and on the range of values of each hyperparameter that we decide. So this has to be updated constantly if the data change.

- **removeFeatures**

  Removes the non important features from the dataset. These features will change for each model.

- **saveModelToFile**

  To avoid calculations, once the models are fitted, we proceed to save them in the memory. This function allows us to do it.

- **loadModelFromFile**

  This function loads the different models saved. When the functionality cannot load one of them, the system will perform the calculations to model it.

- **Error score**

  This procedure enables us to calculate the errors explained in the evaluation methodology, section 3.1

- **predictplot**

  This function allows us to see the Time-Series prediction from the model LSTM.

Explainability layer:

- **DALEX wrapper**

  This procedure allows us to create a wrapper from the models **XGBoost** and **MLR** to generate the importance variable graphic, and perform the calculations of the Shapley Values.

- **LIME wrapper**

  This procedure allows us to create a wrapper from the model **MLP Regressor** to generate the local explanations for each input data features to the output model.

- **IntegratedGradient**

  This procedure allows us to evaluate the contribution of each input data features to the **LSTM** model output.

- **Visualize_importance**

  This function represents the information returned by the IntegratedGradient procedure in a graphic.

- **LayerConductance**

  This procedure allows us to evaluate the contribution of each neuron in a given layer to the **LSTM** model output.

- **NeuronConductance**

  This procedure allows us to compute the attribution of each input feature respect to a particular hidden neuron.

- **visualize_importances_LC**

  Represents the information given by the Layerconductance procedure.

- **visualize_importances_NC**

  Represents the information given by the Neuronconductance procedure.

- **Distributor Estimator**

  This procedure will produce upper and lower bounds given a prediction to calculate the prediction interval. Although it can be applied to any model performed, we must specify manually which is the best model to use. So we could not automate this process.

4 Experiments and Results

In this section, we are going to show the results of the implementation of all methods previously mentioned. Also, we are going to find out if we could extract valuable information to help the business decision-making.

The data is going to be split in two datasets, training and testing, 60% and 40% respectively. For all the models will be the same.

4.1 Hardware specifications

The following hardware specifications have been used to perform all the calculations during the project:

- Processor: Intel Core i5-10300H (2,50 GHz, max 4,50 GHz with Turbo Boost, 4 cores, cache memory 8 MB)
- RAM: 16 GB (8 + 8) DDR4 2933 MHz SoDIMM
- System type: 64-bit Operating System, x64-based processor
- Video card: NVIDIA GeForce GTX 1650 Ti 4 GB GDDR6 128
- SSD memory: 128 GB
- HDD memory: 1 TB

4.2 Model hyperparameter optimizations

We are going to follow the same approach to optimise the algorithms for each model, with only a few difference depending on the model.
Firstly, we will apply the RFE technique to remove all the non important features from the dataset. Secondly, we will use a Random Search to try to reduce the amount of hyperparameter to test. In this way, when we apply the Grid Search technique, it will not have a higher computational cost cause of the reduction of the possible values list. Finally, our model will be ready to be used.
For Multiple Liner Regression, Extreme Gradient Boost, Multi-layer Perceptron Regression, we will follow this approach. Otherwise, for Long Short-Term Memory, we will use only Random Search to reduce the amount of hyperparameter to test and after we will remove all the non important features from the dataset based on the value of the coefficients.

Multiple Linear Regression

Applying the approach aforementioned, we get the results of figure [28].

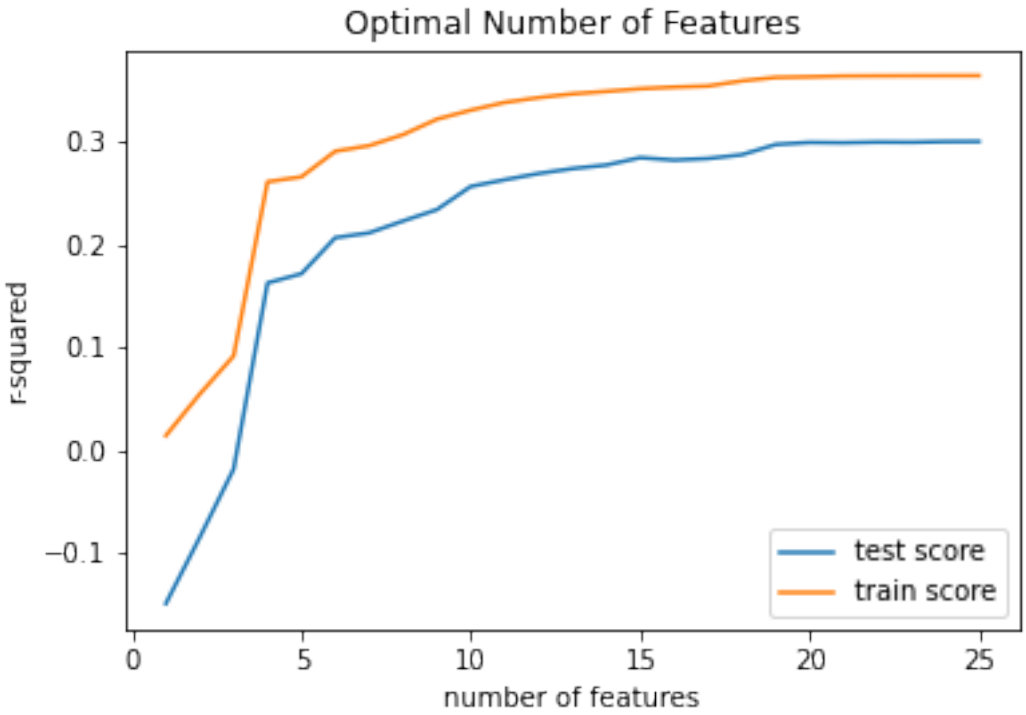

Figure 28: Optimal number of features MLR

Number of features selected	Mean train score (R^2)	Mean test score (R^2)
1	0.014517	-0.154453
2	0.055841	-0.088889
3	0.094035	-0.020074
4	0.269281	0.167642
5	0.274105	0.176802
6	0.299850	0.213036
7	0.305436	0.217924
8	0.316330	0.229673

Table 11: RFE numerical results part I

Number of features selected	Mean train score (R^2)	Mean test score (R^2)
9	0.331989	0.241193
10	0.341076	0.264694
11	0.348848	0.271376
12	0.353616	0.277602
13	0.357326	0.282523
14	0.359692	0.285973
15	0.362034	0.288684
16	0.364028	0.290868
17	0.365175	0.292730
18	0.370336	0.296196
19	0.373721	0.306946
20	0.374110	0.308037
21	0.375198	0.308403
22	0.375373	0.309110
23	0.375468	0.308944
24	0.375581	0.309665
25	0.375616	0.309749

Table 12: RFE numerical results part II

Checking closer the results, we can conclude that the best scoring results are with all the features. Although we cannot appreciate a real difference between 22 and 25 features. In this case, we are going to use all features to fit the model because in terms of computational cost it is not high and we will get a small improvement.

XGBoost

Applying the approach aforementioned, we get the results from table [13] and figure [29].

Number of features selected	Mean train score (R^2)	Mean test score (R^2)
16	0.624829	0.505730
17	0.629749	0.516068
18	0.631637	0.516907
19	0.631576	0.518757
20	0.631235	0.518727
21	0.631181	0.518832
22	0.631174	0.518543
23	0.630993	0.518744
24	0.630811	0.518378
25	0.630811	0.518378

Table 13: RFE numerical results

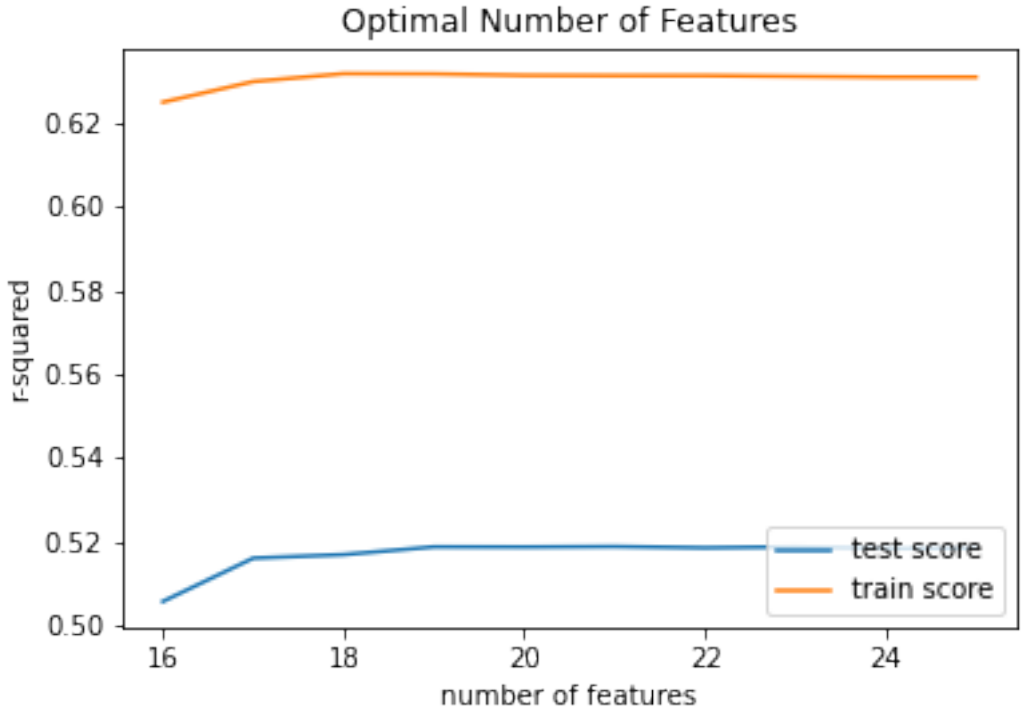

Figure 29: Optimal number of features XGBoost

In the results above (figure [29]), it can be observed that the model XGBoost works better with 21 features than with the other combinations, with a mean test score of 0.518832.
The features we are going to keep are:

- ANO_FACTURA
- CATEGORIA
- CIUDAD_CONTACTO
- CONSUMER_COLOR
- CORDONES
- CREMALLERA
- EDAD_COMPRA
- ESFUERZO_VENTA_ID
- FECHA_FACTURA
- GENERO_PRODUCTO
- IDIOMA_CONTACTO
- JERARQUIA_PROD_ID
- MES_FACTURA
- NUMERO_DEUDOR_PAIS_ID
- OUTSOLE_SUELA_SUBTIPO
- OUTSOLE_SUELA_TIPO
- PLANTILLA_EXTRAIBLE
- PRODUCTO_ID
- TALLA
- TEMPORADA_COMERCIAL_ID
- TIPOLOGIA

The features discarded are:

- CONTACTO_SN
- EDAD_SN
- GENERO_CONTACTO
- GRUPO_ARTICULO_PRODUCTO_ID

Final hyperparameter optimization:

- learning_rate =0.01
- n_estimators=215
- max_depth=10
- min_child_weight=0.8
- subsample=1
- nthread=4

MLP Regression

Applying the approach aforementioned, we get the next results.

Final hyperparameter optimization:

- hidden_layer_sizes=(300, 300)
- activation= logistic
- solver= adam
- alpha=0.01
- batch_size = auto
- learning_rate= constant
- learning_rate_init=0.01
- max_iter=1000
- shuffle=False
- tol=0.0001
- verbose=True
- early_stopping= True
- validation_fraction=0.1
- beta_1=0.9
- beta_2=0.999
- epsilon=1e-08

LSTM

In this case, the approach is a quite different, as this has the highest computational cost, we cannot perform a Grid Search as we have hardware limitations and it would take more than 10 hours to finish. So, we are going to use Random Search technique to select the best fit for the hyperparameters. After, we will reduce the number of features from the dataset checking their coefficient values.

Discarded variables:

- CIUDAD_CONTACTO
- GENERO_CONTACTO
- TIPOLOGIA
- FECHA_FACTURA

Final hyperparameter optimization:

- Epochs = 50
- Learning rate = 0.01
- Input size = 21
- Hidden size = 175
- Layers = 2
- Batch size = 200

Hybrid approach

For this approach, we are going to use the best two methods in terms of results (**XGBoost** and **LSTM** [7]), so we do not need to optimise them. For the Meta-Regressor, we will choose the **XGBoost** model. Instead of using the same hyperparameters, as we will reduce the amount of features from 21 to 2, we can tune again the model to get better results without compromising the performance.

Final hyperparameter optimization:

- learning_rate =0.01
- n_estimators=320
- max_depth=20
- min_child_weight=0.8
- subsample=1
- nthread=4

[7]Check the section **Results obtained** to see the values obtained

4.3 Results obtained

In this section, we are going to compare the results obtained evaluating the errors from the models.

Model	MAE	MSE	RMSE	R^2
MLR	68.43	6233.39	78.95	-2.11
XGBoost	21.71	901.89	30.03	0.56
MLP Regressor	38.5	2485.44	49.85	-0.2
LSTM	27.74	1375.54	37.08	0.34
Hybrid	17.77	628.45	25.06	0.7

Table 14: Prediction results

Looking at the table [14], our **Hybrid approach** has outperformed all the other methods. We got the best results minimising the errors and maximising the proportion of variance explained by the independents variables, although **XGBoost** was relatively close.
We could extract from the results that there were not linear relationships among the independents and the dependent variables such that the **Multiple Linear Regression** could not achieve decent results.
However we have 3 years of data, the frequency with which people buy shoes is very different from the one that, for instance, films are selected on Netflix, where every day thousands and thousands of users interact with the platform. This could be seen during the exploratory data analysis, in the distribution of the different features of the dataset. We had highly imbalanced data and this affected negatively to our neural networks algorithms. That was why **LSTM** did not perform as well as **XGBoost**.

Prediction intervals

The main goal of this prediction interval, is to create a system to compare and analyse models, observing where the future data should fall. We are going to create a **Distributor Estimator**, with the principal assumption that the predictions follow a normal distribution. Therefore, the system will produce a prediction and will create bounds subtracting and adding the standard deviation.

The final bounds will be calculated as follows:

$$Upper_{bound} = model_{prediction} + sd$$
$$Lower_{bound} = model_{prediction} - sd$$

For our purpose, we are going to analyse what should be the total amount of money that Camper will earn in 20 days in a row, grouped in 3 different countries (Spain, US and Italy) and calculate the prediction interval for them. To do this, we will calculate the cumulative prediction for all the products sold on the same day for each country, and after we will perform the RMSE as our standard deviation (sd in the previous formula), to create upper and lower intervals.

We are going to analyse two cases: a preliminary analysis where each model will act as the final system and we will try to check how many points are not in the interval, and the final implementation based in the model we got best prediction results, **Hybrid approach**.

Multiple Linear Regression:

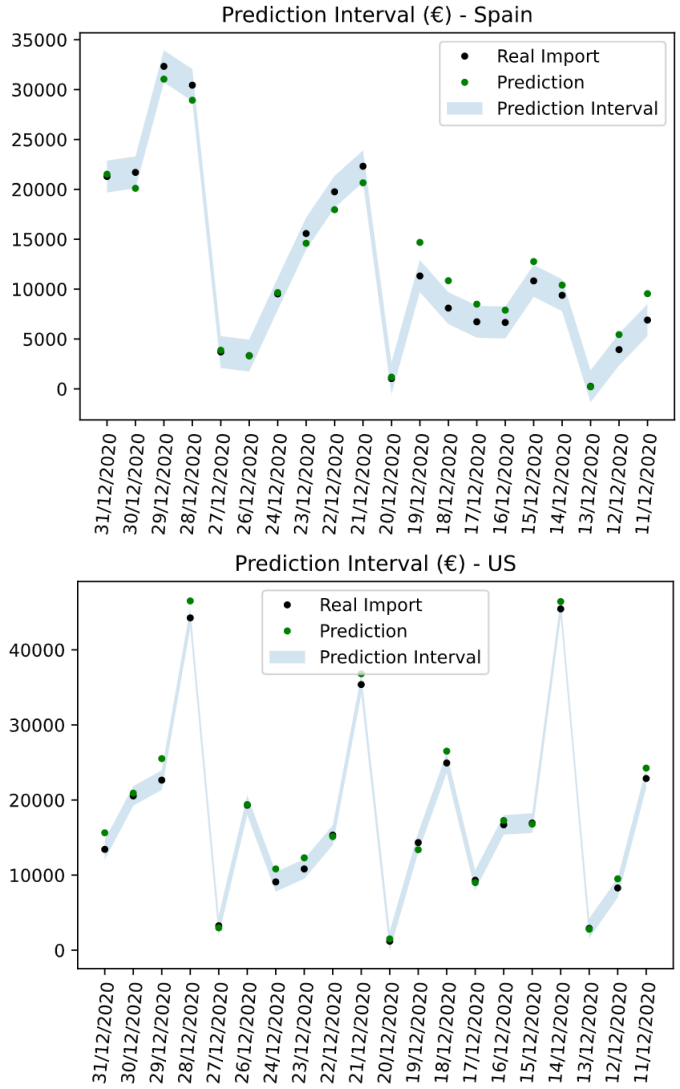

Figure 30: Prediction interval MLR (Spain and US)

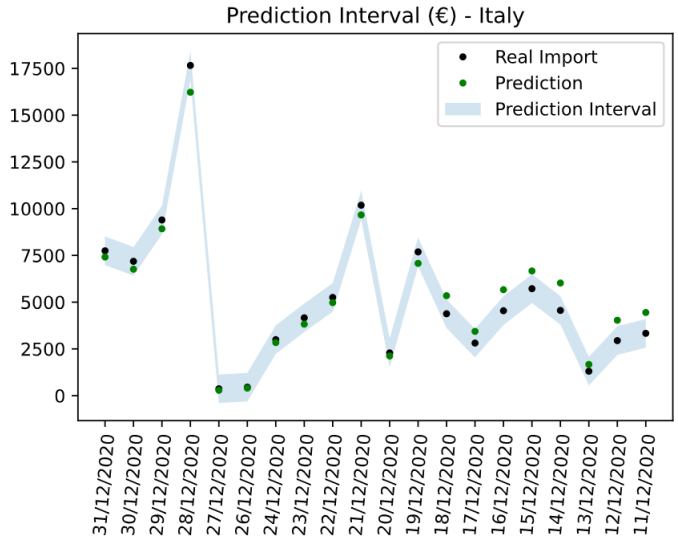

Figure 31: Prediction interval MLR (Italy)

Looking at the results from Spain (figure [30]), we can see that there are half of the points outside the range or just in the limit, with only 3 points really close to the correct import. All the others are correctly placed in the interval. Otherwise, for the US (figure [30]) the results differ, we only have 3 points which are correctly placed in the interval, all the other values are in the limits or outside. Finally, the best results found are in Italy (figure [31]), only 8 imports are out of the interval.
Considering the results given, we can extract that the total incorrect points are 27, almost half of the total points, that means that 47% of the points are misplaced. Not good results at all.

XGBoost:

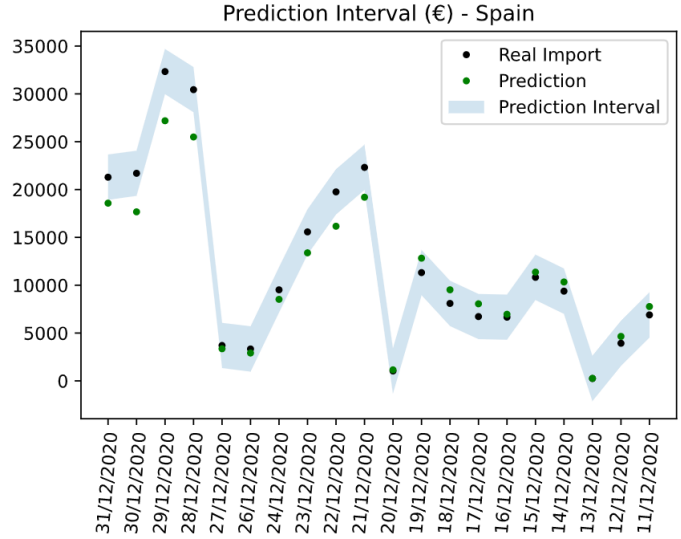

Figure 32: Prediction interval XGBoost (Spain)

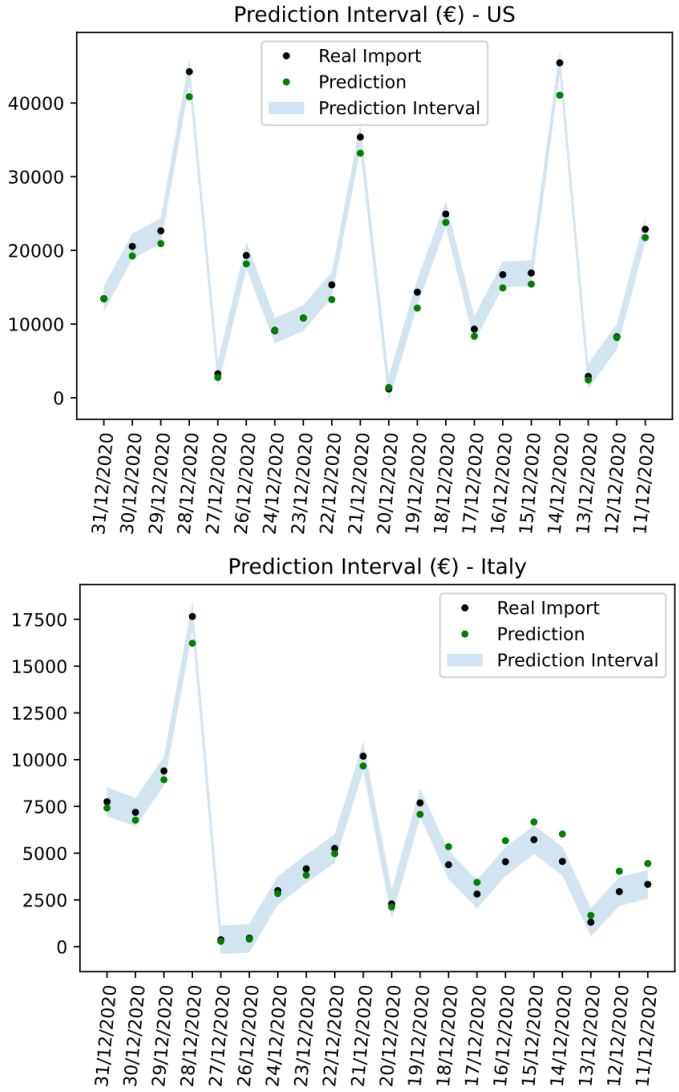

Figure 33: Prediction interval XGBoost (US and Italy)

Checking the results from the three countries (figures [32] and [33]), we only have 6 points out of the prediction interval in each graph. As it can be seen, this model is more consistent across the different inputs, it works much better.
Considering the results given, we can extract that the total incorrect points are 18, less than half of the total points, that means that less than approximately 32% of the points are misplaced. A good result compared to the aforementioned.

MLP Regressor:

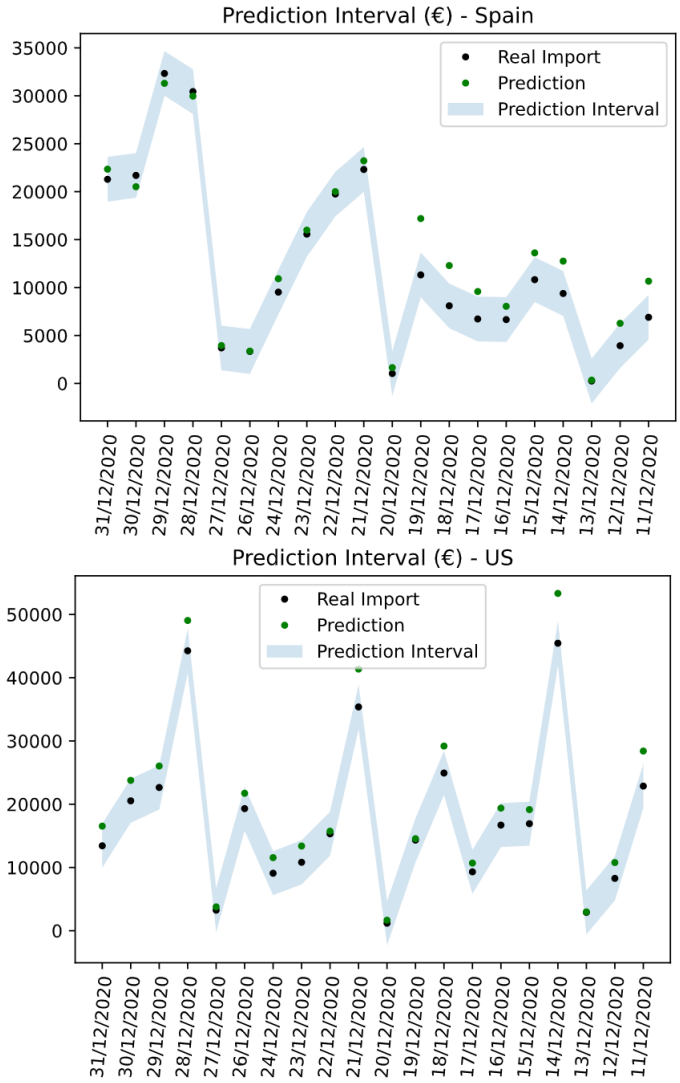

Figure 34: Prediction interval MLP Regressor (Spain and US)

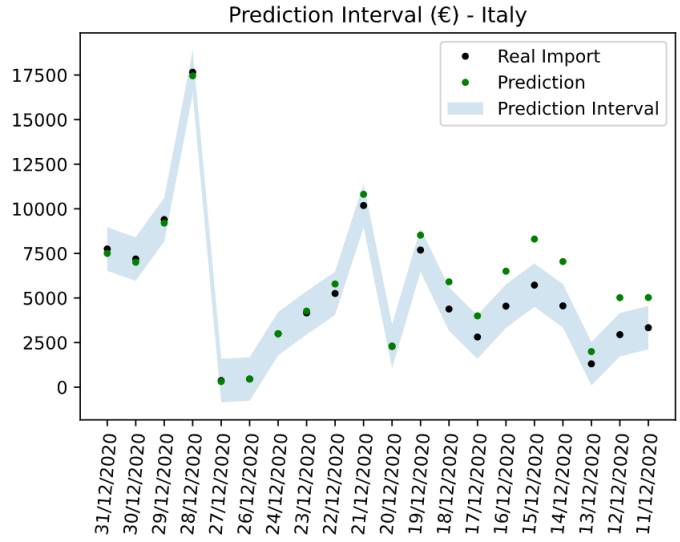

Figure 35: Prediction interval MLP (Italy)

Looking at the figures [34] and [35], this model works similar to the second one in terms of constancy, but with worse results. Although, it is not the worst but it can be seen how unstable it is for its peaks.
Considering the results given, we can extract that the total incorrect points are 21, less than half of the total points, that means that approximately 35% of the points are misplaced.

LSTM:

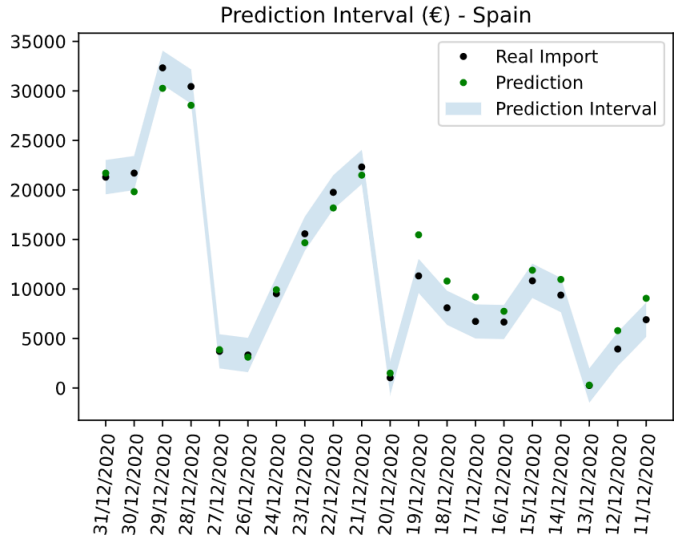

Figure 36: Prediction interval LSTM (Spain)

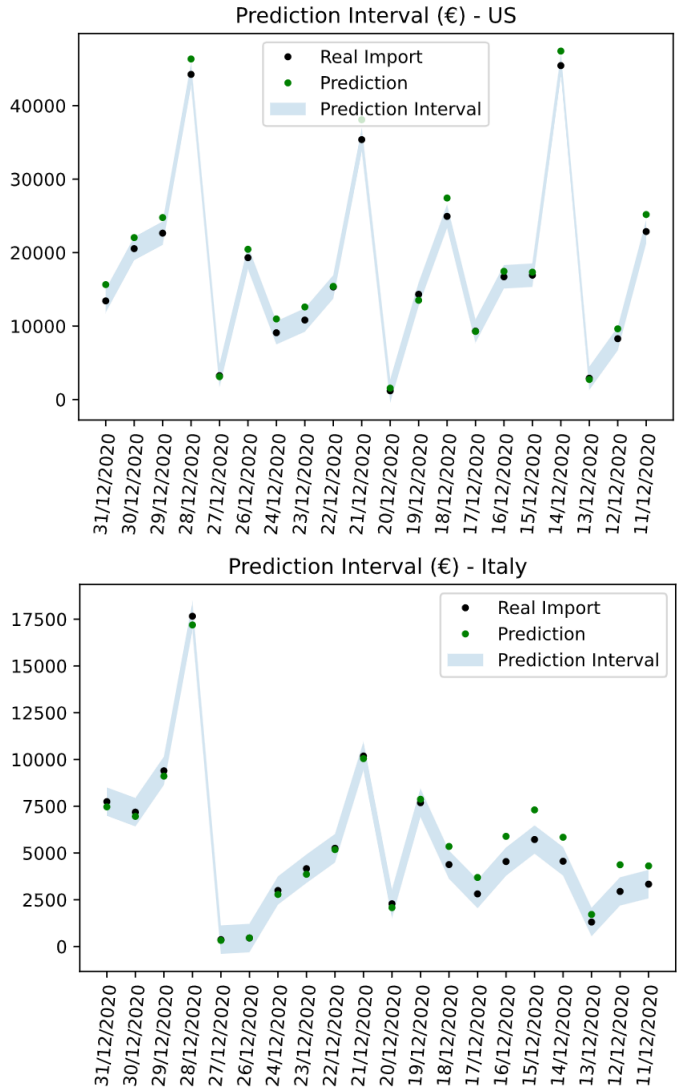

Figure 37: Prediction interval LSTM (US and Italy)

Checking the figures [36] and [37], we are struggling with almost the same points as the MLP Regressor but with a slight improvement. We got almost the same results in terms of error that XGBoost and outperforming MLR.
Considering the results given, we can extract that the total incorrect points are 19, less than the half of the total points, that means that approximately 32% of the points are misplaced.

Hybrid approach:

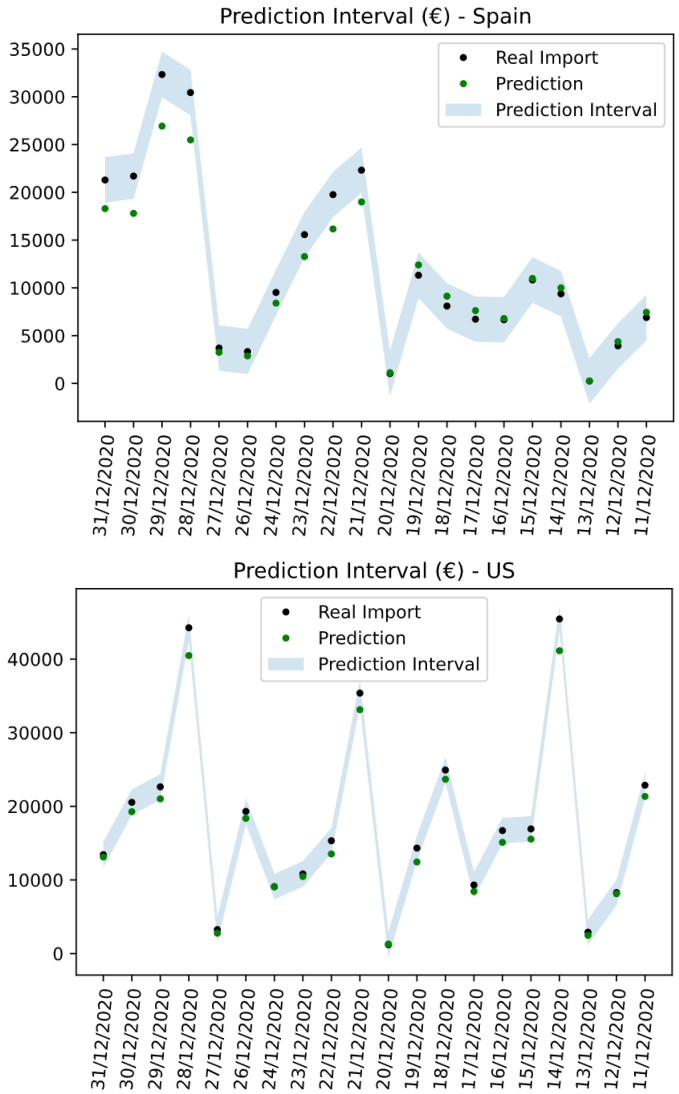

Figure 38: Prediction interval Hybrid approach (Spain and US)

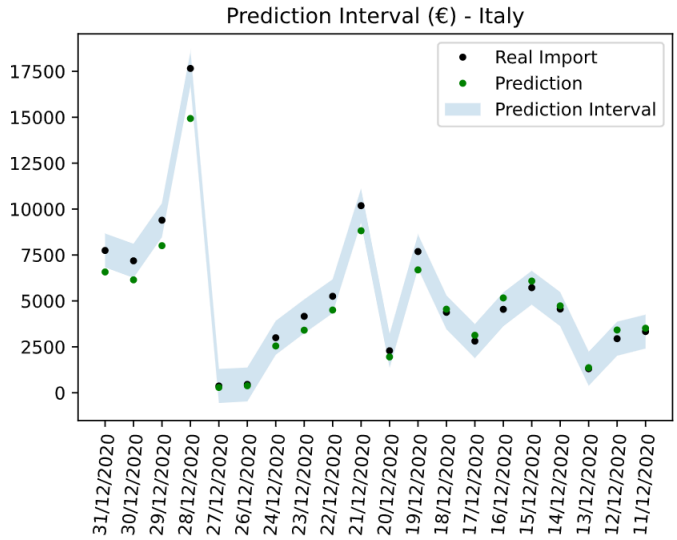

Figure 39: Prediction interval Hybrid approach (Italy)

Looking at the figures [38] and [39], we have less than 6 points out of our prediction interval in each graph. As it can be seen, this model is more consistent across the different inputs. By far this is the best model we have created.
Considering the results given, we can extract that the total incorrect points are 13, less than the half of the total points, that means that less than approximately 22% of the points are misplaced. A good result compared to the aforementioned models.

Final system:

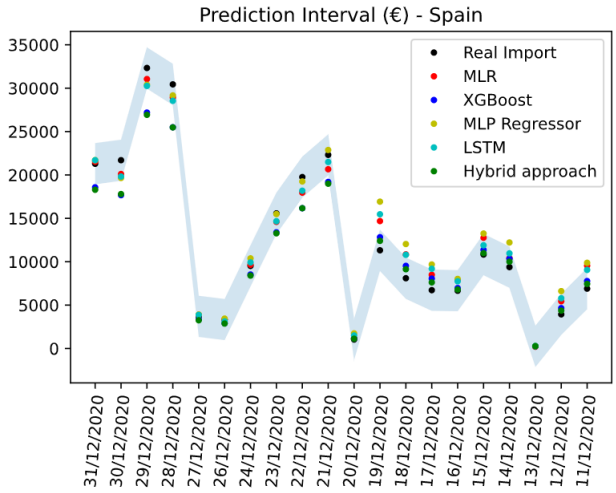

Figure 40: Prediction interval final system (Spain)

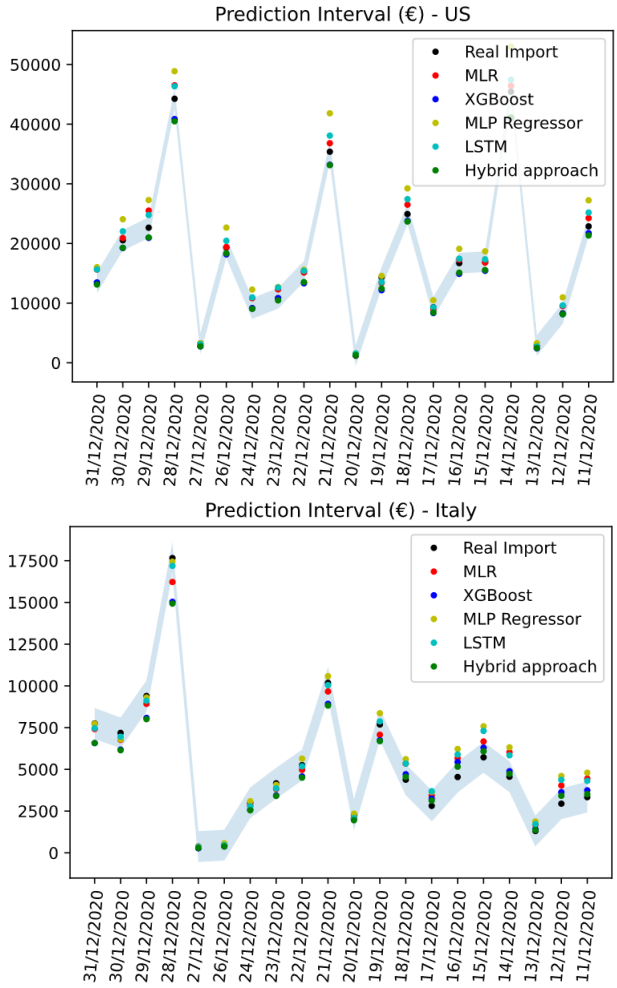

Figure 41: Prediction interval final system (US and Italy)

Looking at the results of our final system implemented (figure [40] and [41]), we can see the reality of each model when we try to predict where should fall a point. It could be observed that for the models like **MLR** and **MLP Regressor**, in most cases the points are out of the range or really close to bounds. As before, when there were peaks, **LSTM** struggles to keep the predictions inside the range.
The main goal of using this system is to be able to compare methods and see possible improvements. The good point of using this **Distributor Estimator** implementation is that as much as the principal prediction system improves, it refines the upper and lower bounds, so the interval gets narrower and the results are more accurate.

4.4 Interpretability

In this section, we are going to create an interpretability layer to help us understand the decision-making of each model and why the features are important in each prediction.

Apart from analysing the features individually, in some cases we can group all the common features to create the following dictionary:

- **Date**
  ANO_FACTURA, MES_FACTURA, FECHA_FACTURA, TEMPORADA_COMERCIAL_ID
- **Product**
  PRODUCTO_ID, TALLA, GRUPO_ARTICULO_PRODUCTO_ID, GENERO_PRODUCTO, CATEGORIA, TIPOLOGIA, CONSUMER_COLOR, CREMALLERA, CORDONES, OUTSOLE_SUELA_TIPO, OUTSOLE_SUELA_SUBTIPO, PLANTILLA_EXTRAIBLE
- **Age**
  EDAD_SN, EDAD_COMPRA
- **ContanctInfo**
  NUMERO_DEUDOR_PAIS_ID, CIUDAD_CONTACTO, IDIOMA_CONTACTO, GENERO_CONTACTO, CONTACTO_SN
- **SalePerson**
  ESFUERZO_VENTA_ID

In addition, we are going to explore what features are important for a certain input data, so with this aim, we are going to separate the data depending on the Country. This way, it could be seen what changes on the customer behaviour there will be. The two subsets are created filtering by the column called *NUMERO_DEUDOR_PAIS_ID* by ES (code 14) and US (code 50). In the **Appendix A**, section [6.4], you can find the input data for each output representation from both countries.

Multiple Linear Regression:

The coefficients of the model give us the relationship between the predictor variable and the response variable, being positive as an increase and negative as a decrease. We can observe in the image below, how the variables related with the date are very influential, and some features from the product itself. In general, only a few coefficients are determinant in this model, the others do not have a important weight.

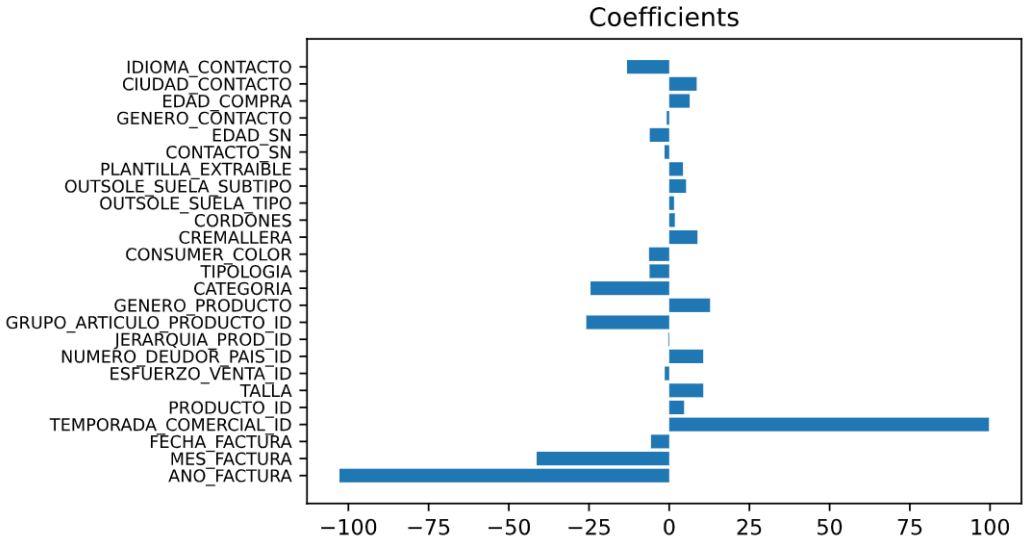

Figure 42: Coefficients MLR

Adding the interpretability layer created using the package **DALEX** [19], we can observe that the feature called category, related to the product, is the most relevant feature. This makes sense as this field has coded the information of the product. Also, the age of the customer and the language are determinants to define the dependent variable.

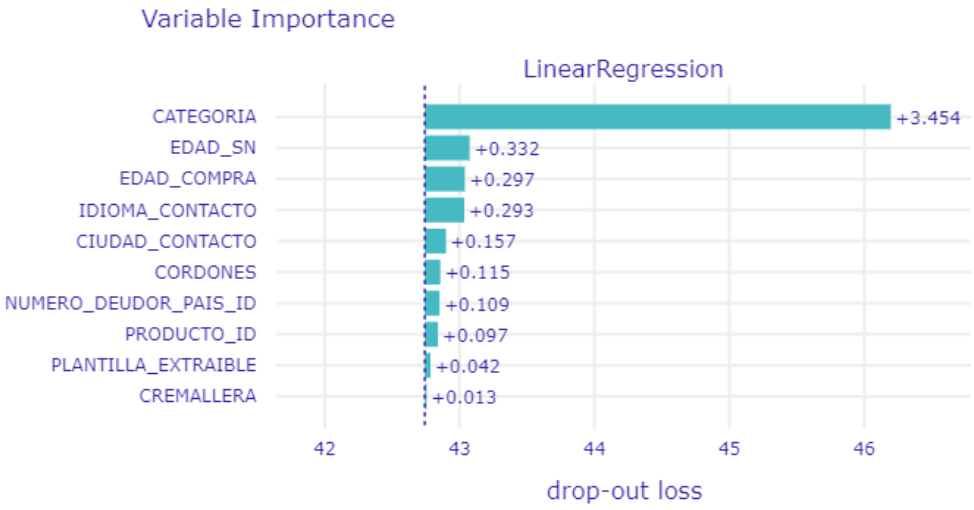

Figure 43: Variable importance

Grouping by common features, the most important ones are those related to the product itself. Age and nationality are also important.
To sum up, this **MLR** model considers the features related with the product as the most important, concretely the feature called category. Also, contact info and age have approximately the same importance between them but far from the product features.

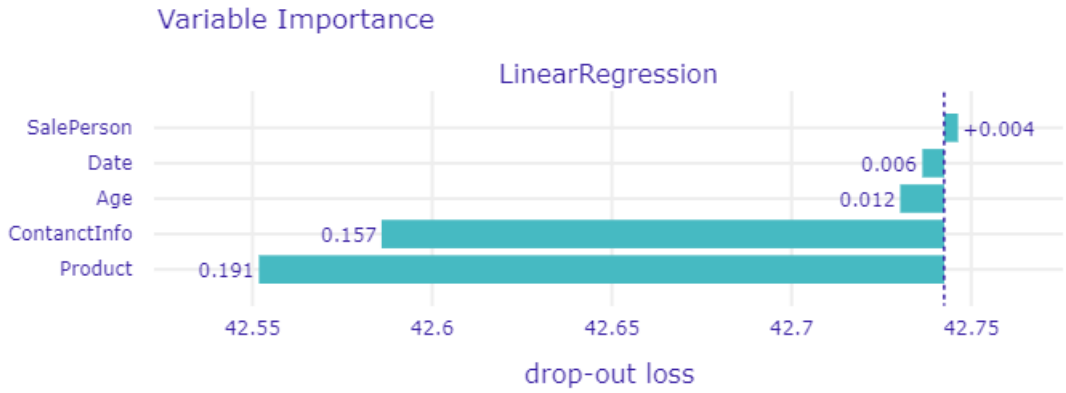

Figure 44: Grouped variable importance

As we aforementioned, now we are going to predict the behaviour of the model given an input data.

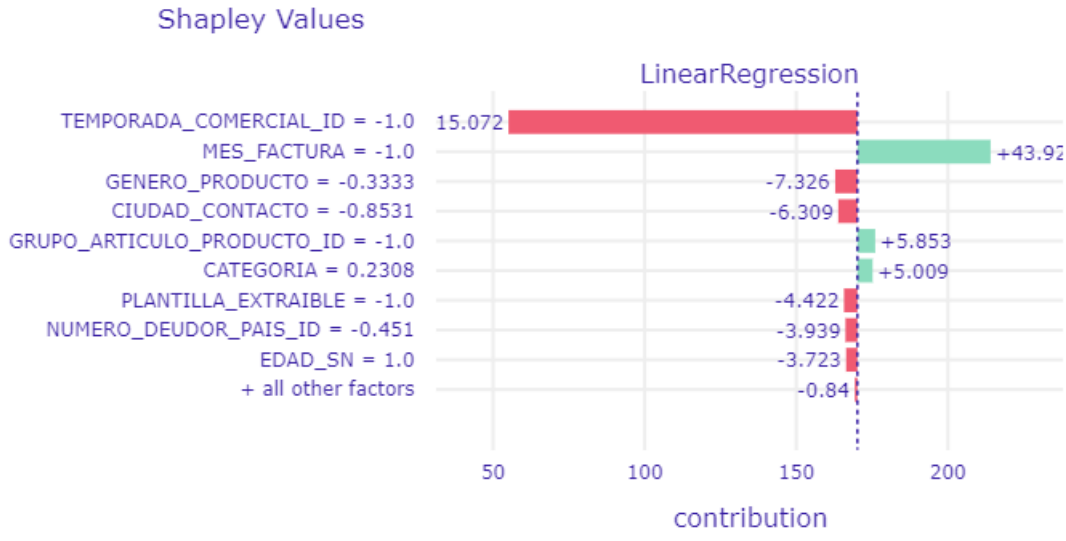

Figure 45: MLR Shapley Values Spain I (Input 1)

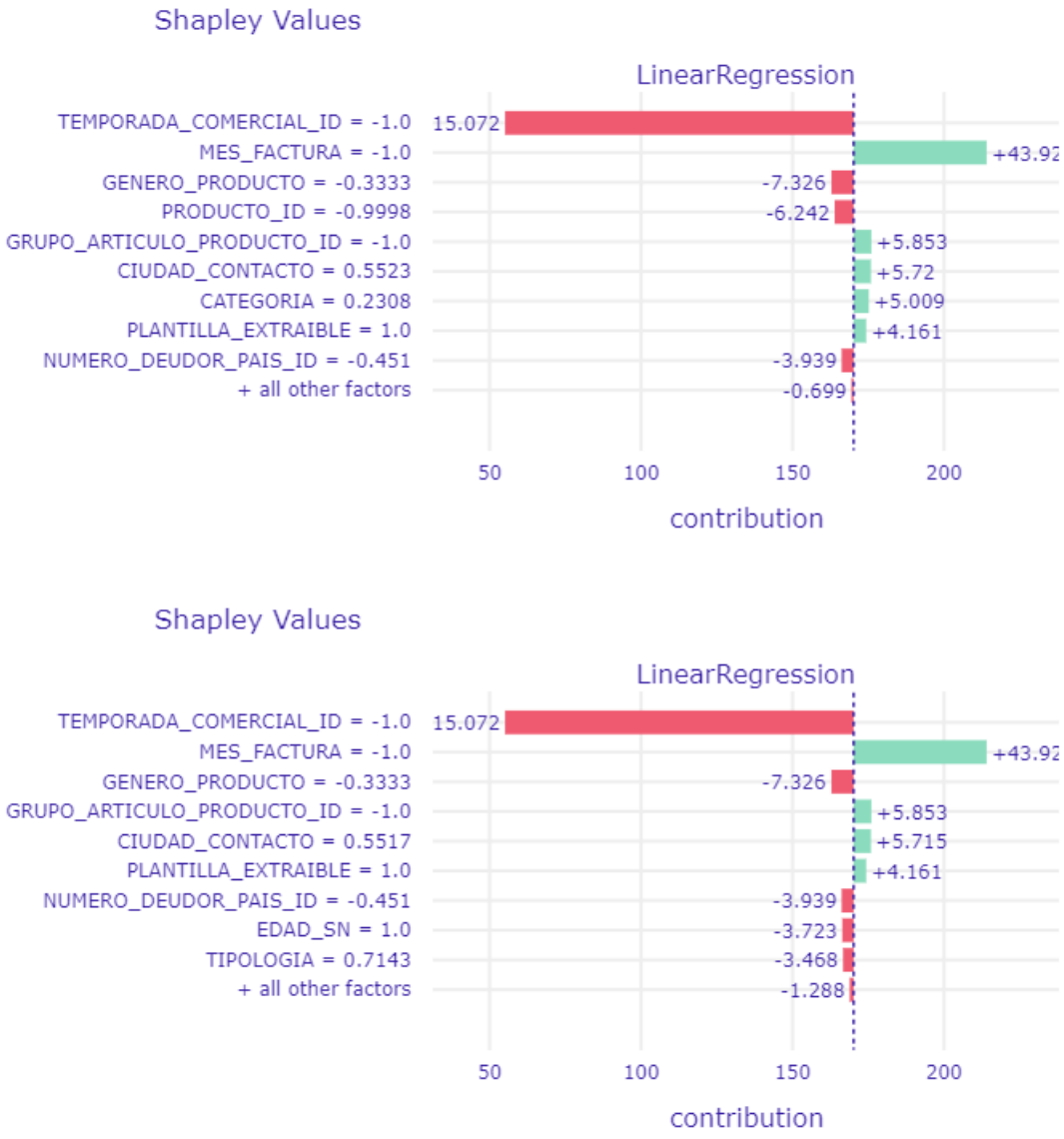

Figure 46: MLR Shapley Values Spain II (Input 2 and 3)

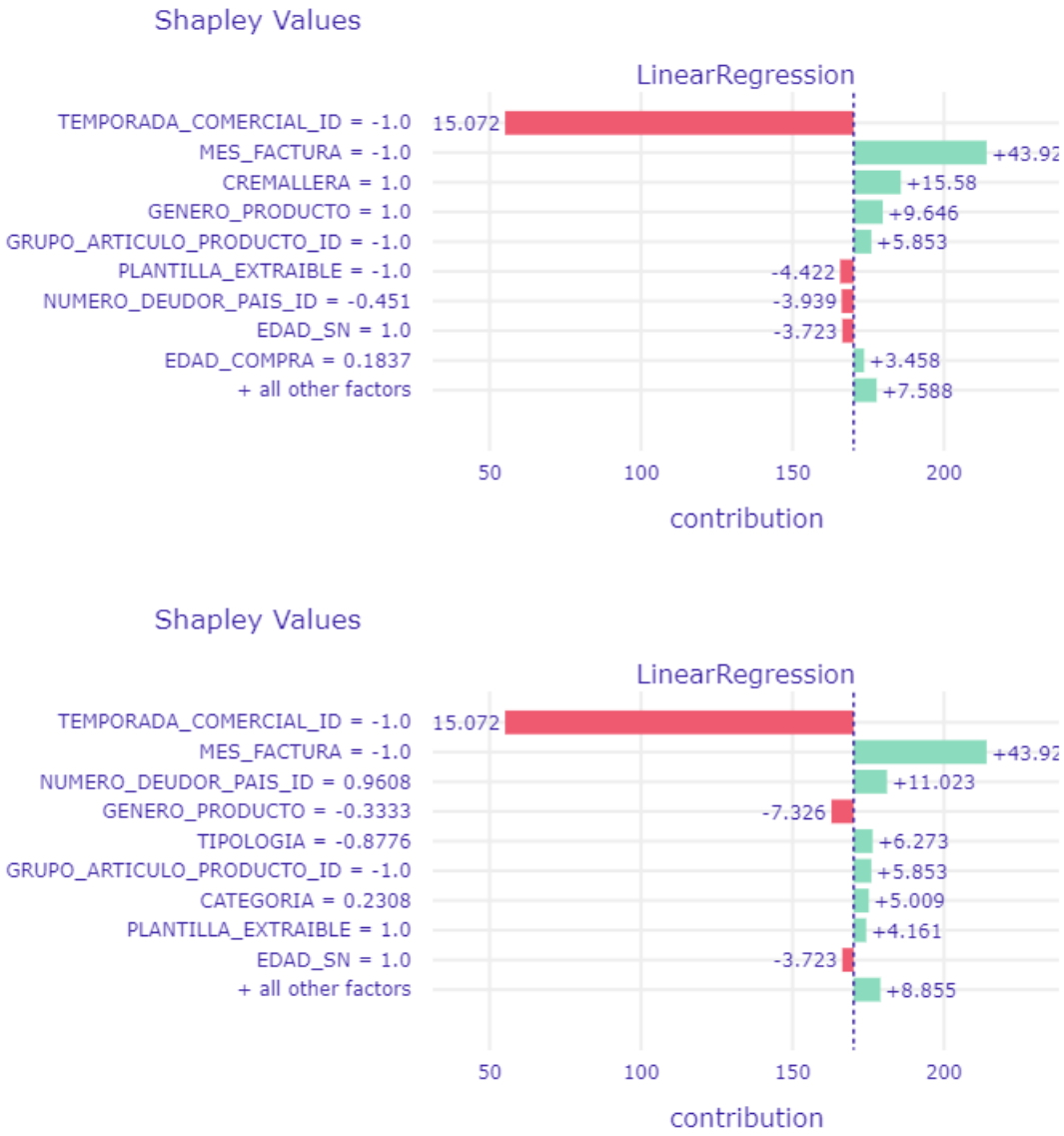

Figure 47: MLR Shapley Values Spain III (Input 4) and Shapley Values US (Input 1)

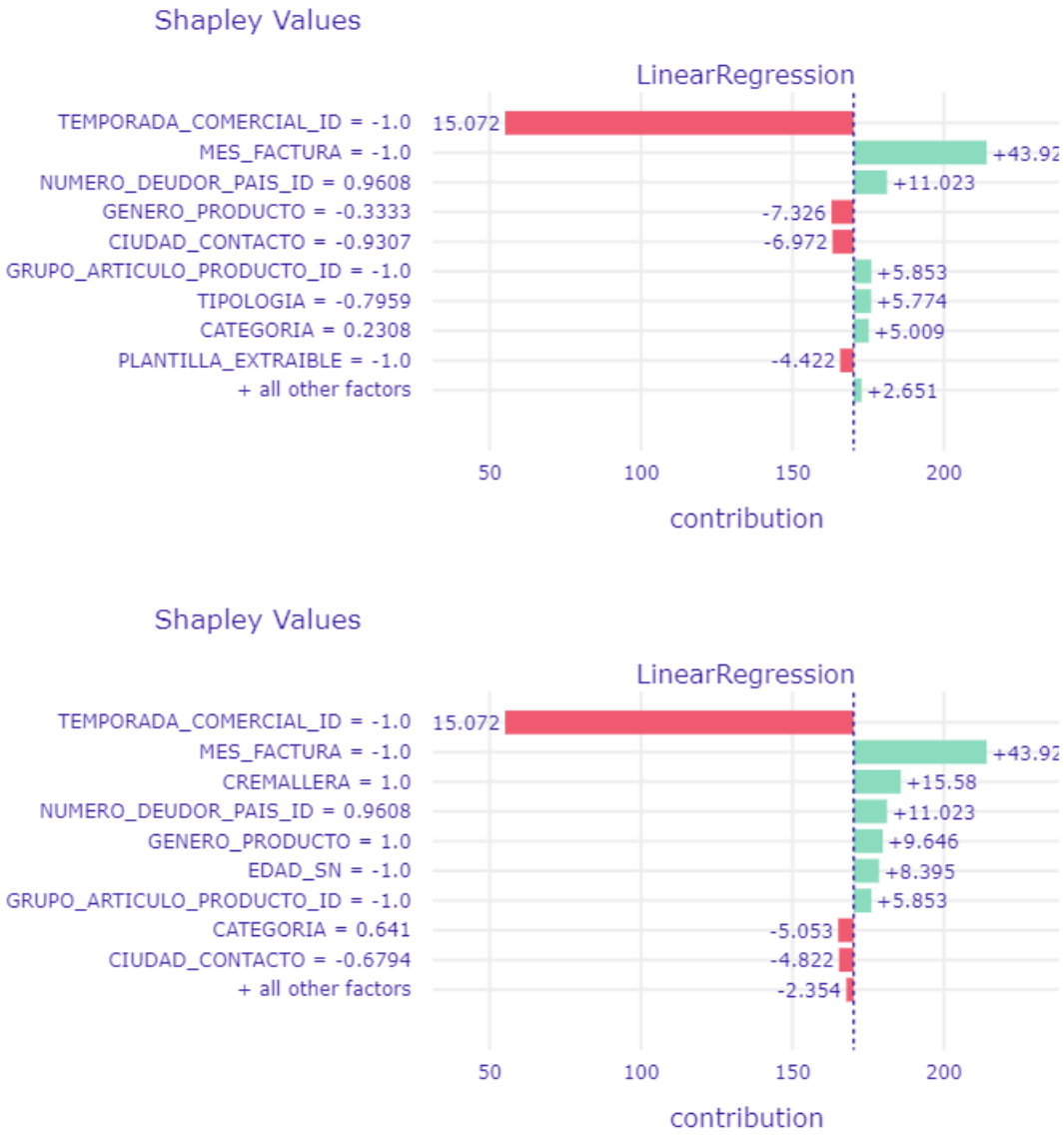

Figure 48: MLR Shapley Values US (Input 2 and 3)

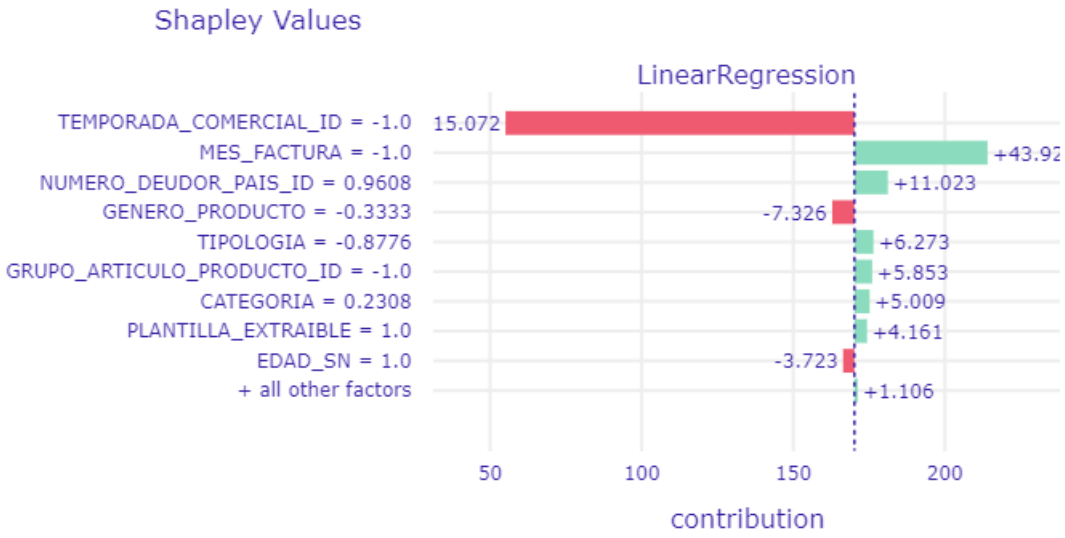

Figure 49: MLR Shapley Values US (Input 4)

As you could see in figures [45], [46], [47], [48] and [49], we got similar outputs, they are very homogeneous. All features related to date were the most significant, season and month. The most these features increased, the better it was. What this means is that, for instance, for March (month 3), it would not affect the results as much as it would do July (month 7). The reason for this is that the highest earnings have been received during summer sales, July and June. But in October (month 10) Camper usually has less earnings and in December (month 12), it rises up again.
As we could see, in this case, Spain and US do not have a significant difference in terms of features importance. We could not find a different model behaviour.
We selected different types of woman ankle boots [6.4] (Spain input 4 we will call it A and US input 3 we will call it B) to see if their features affected to the final prediction. We observed that having different values for each feature CREMALLERA, in both cases it has a relevant importance. This means, that in both models, the fact of having and not having laces increased their value. Also, not having removable insole in the boot A affected negatively to the model. This implies that if the boot would have had a removable insole, it would have increased its value. Moreover, we saw that the colour and the type of the outsole that at first sight they should have a relevant influence in the prediction, they did not in this model.

Using the algorithm explained in section [3.6.6], we are going to analyse the model calculating the contribution of the i-th features. We have tuned the algorithm in the way that for each features it will calculate the contribution 100 times.
Looking at the figure [50], the features that have more influence are the year, month and season. We can extract that the features related with the date are the ones which determine the prediction, they increase the average contribution of the prediction. Otherwise, ESFUERZO_ID is decreasing the average contribution of the prediction, so it has a negative influence. In comparison, all the other features don't have enough impact in

the prediction, the contribution is not relevant at all.

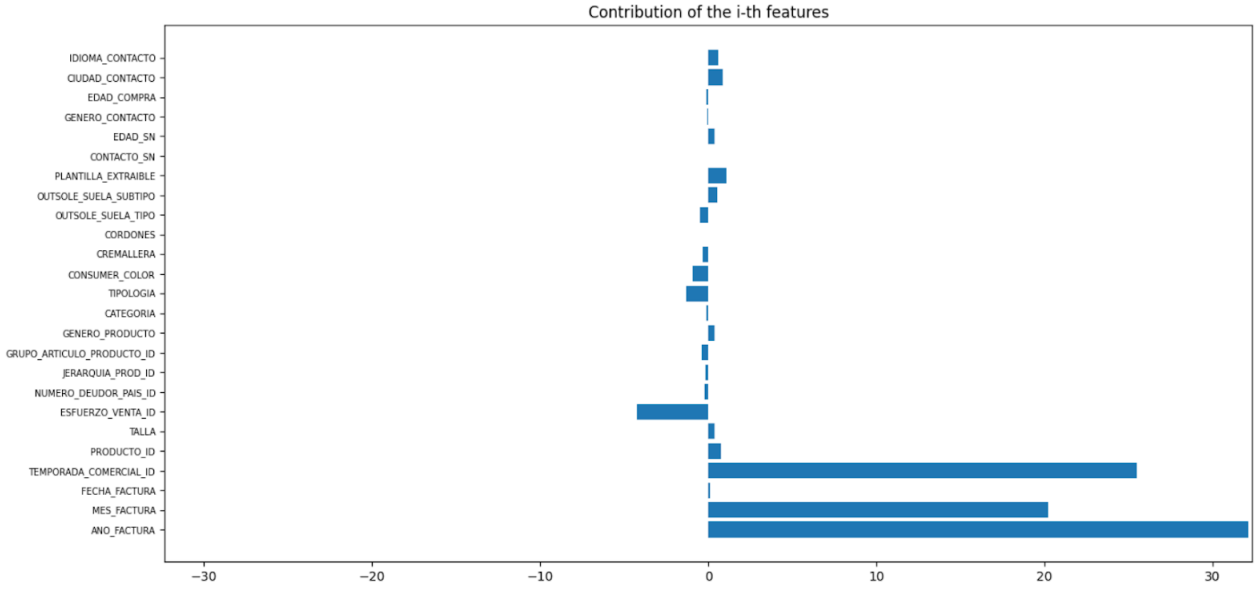

Figure 50: MLR contribution i-th feature

XGBoost:

Adding the interpretability layer created using the package **DALEX** [19], we can observe in the figure [51] that the most important features are the variables which belong to the product characteristics and to the customer's information.

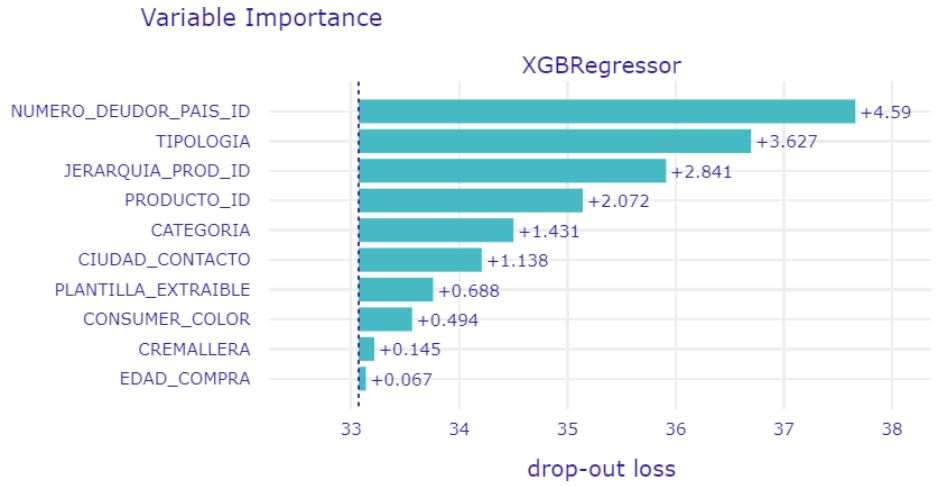

Figure 51: XGBoost variable importance

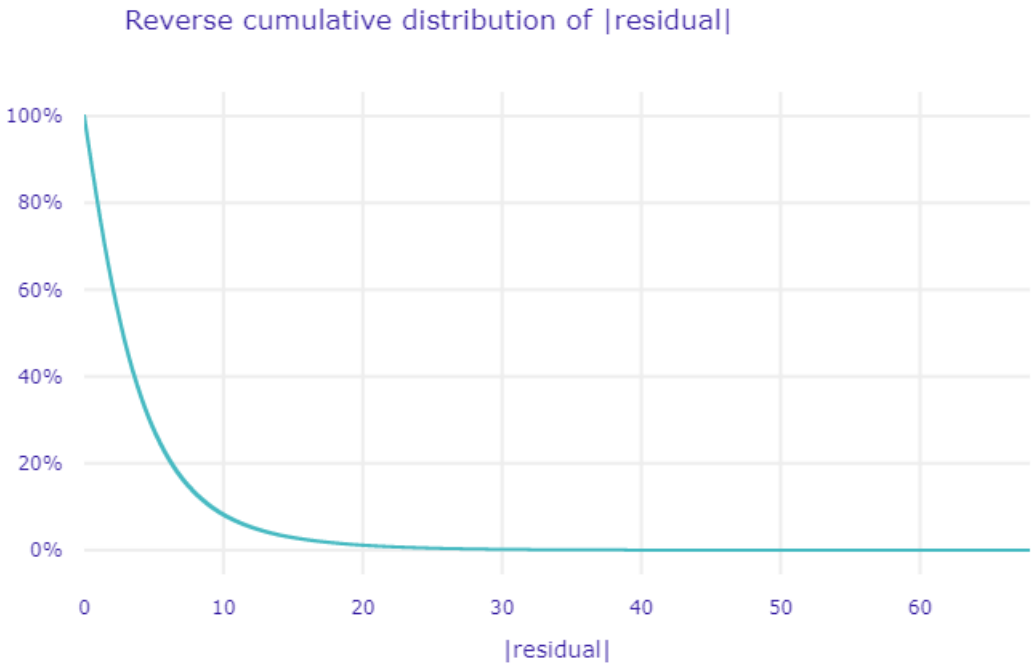

Figure 52: Reverse cumulative distribution of residual

Grouping by common features (figure [53]), the most important ones are those related to the product itself. Nationality and age are also important, otherwise the variables related with the information of the seller or date are insignificant in this model.
To sum up, this **XGboost** model considers the features related with the product as the most important. Also, contact info and age have approximately the same importance between them but far from the product features.

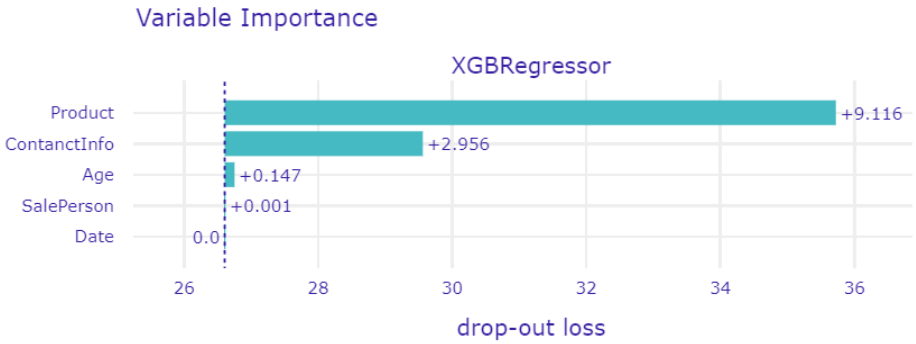

Figure 53: XGBoost grouped variable importance

Using the algorithm explained in section [3.6.6], we are going to analyse the model calculating the contribution of the i-th features.
We have tuned the algorithm in the way that for each features will calculate the contribution 10 times. Because of the hardware limitations, we cannot increase the number of times.

Looking at the results from the figure [54], the obvious observation is that there are multiple features that the model considers important. First, the features that increase the average contribution of the prediction are CONSUMER_COLOR, TIPOLOGIA, CORDONES, CREMALLERA, OUTSOLE, PLANTILLA_EXTRAIBLE, GENERO_PRODUCTO, PRODUCTO_ID, ANO_FACTURA and MES_FACTURA. Clearly, we have two subgroups, features related with the characteristics of the product, and related to the date. We could say that this model has a logic human being thinking to predict the money earn, as it is based on how is the model and when it is sold.
Also, from the results we can discard the features ESFUERZO_VENTA_ID and CATEGORIA such that they have a negative influence on it. All the other negative contributions are $\varphi \leq -2$, so we have decided to keep them.

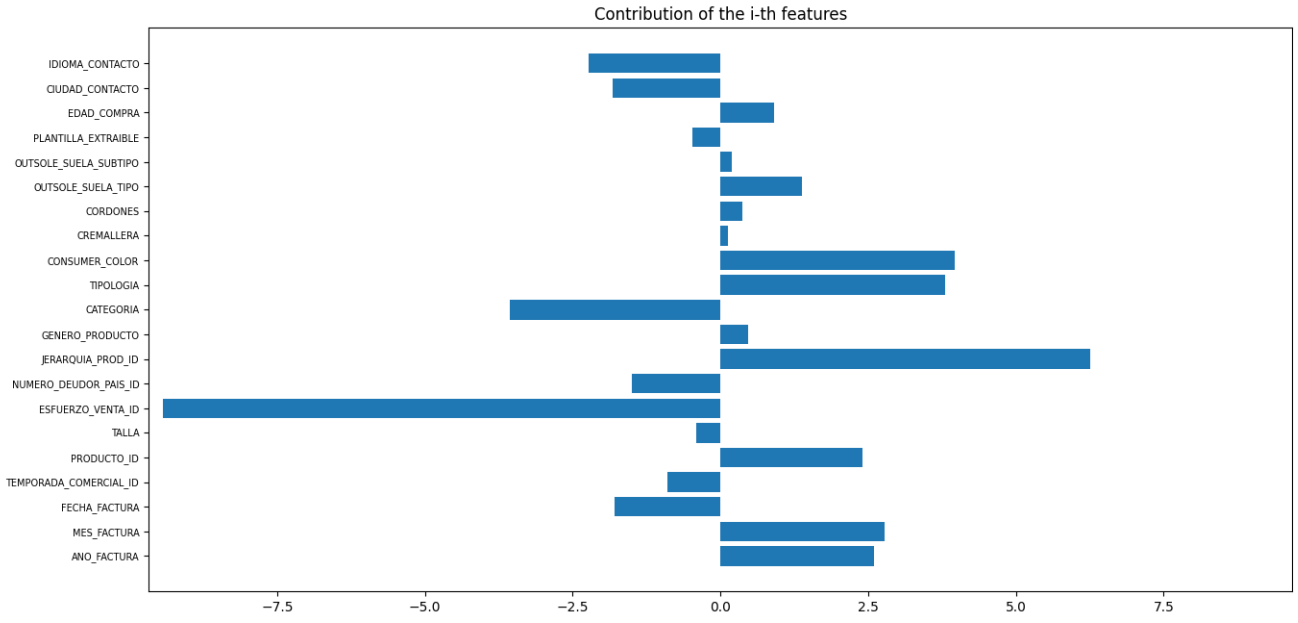

Figure 54: XGBoost contribution i-th feature

MLP Regressor:

Adding the interpretability layer created using the package **LIME** [9], we are going to interpret what affects the prediction using the input data given in section [6.4].

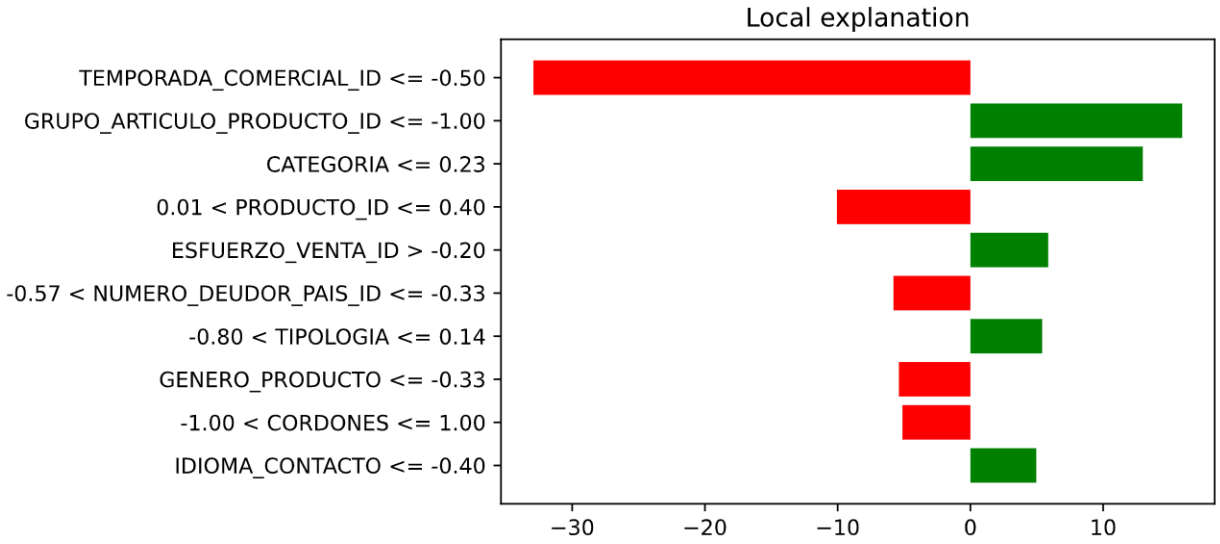

Figure 55: MLP LIME Spain input 1

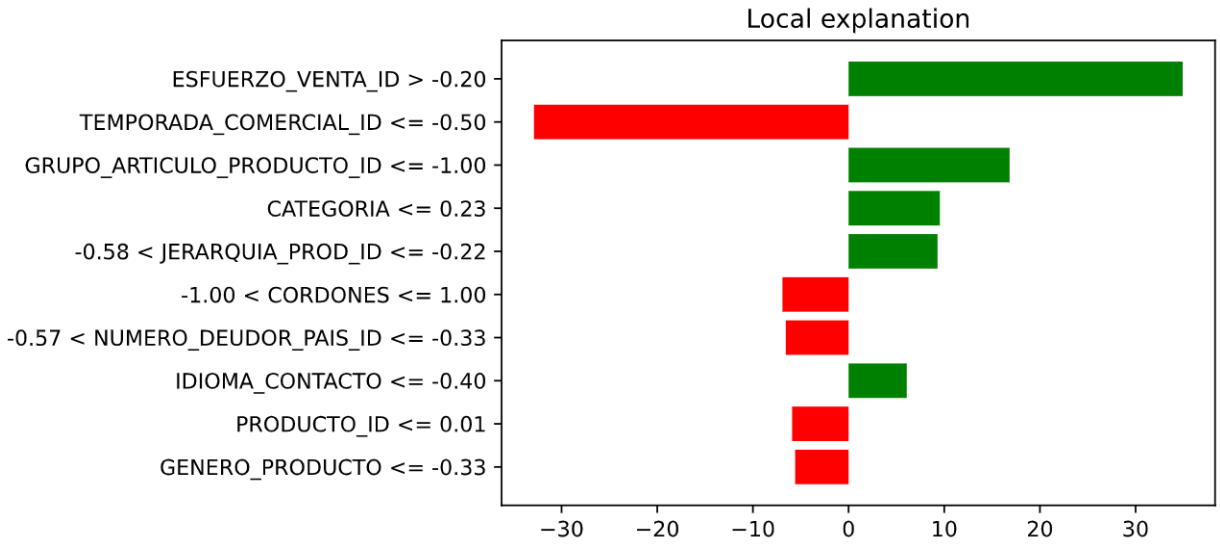

Figure 56: MLP LIME Spain input 2

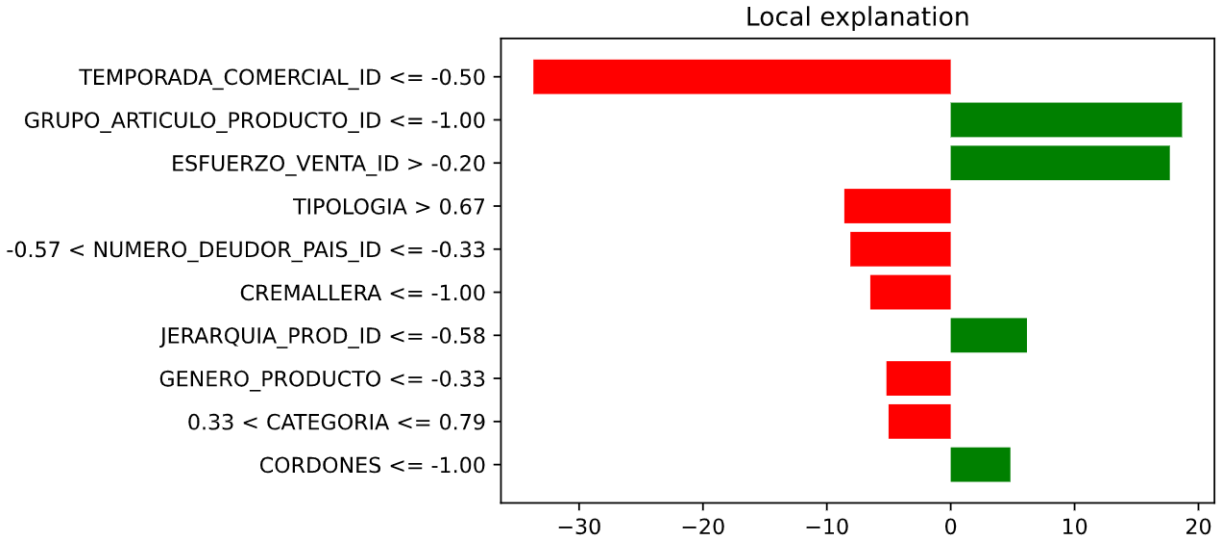

Figure 57: MLP LIME Spain input 3

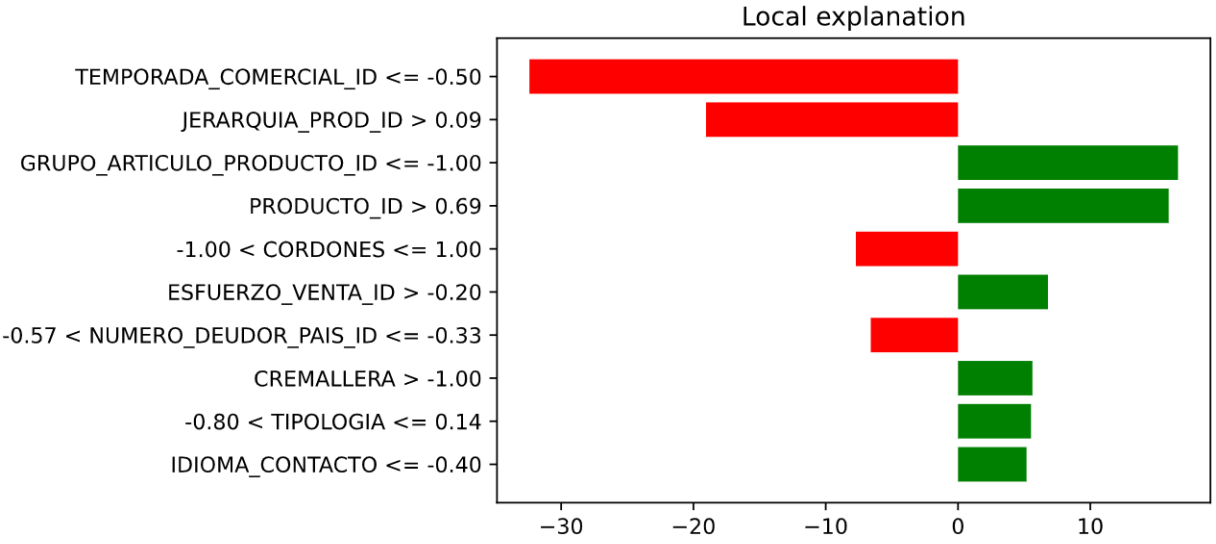

Figure 58: MLP LIME Spain input 4

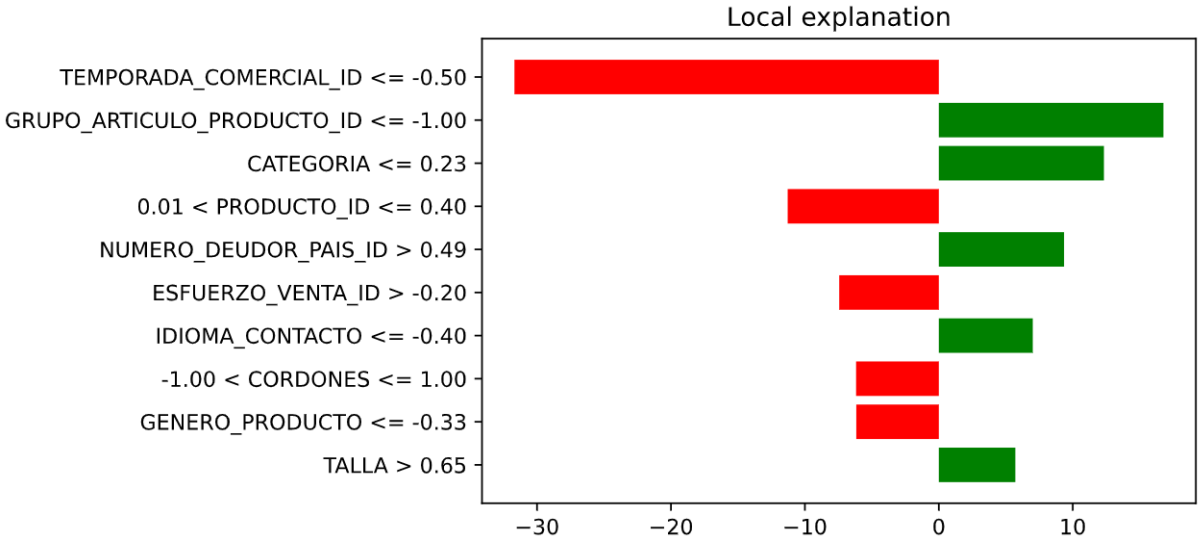

Figure 59: MLP LIME US input 1

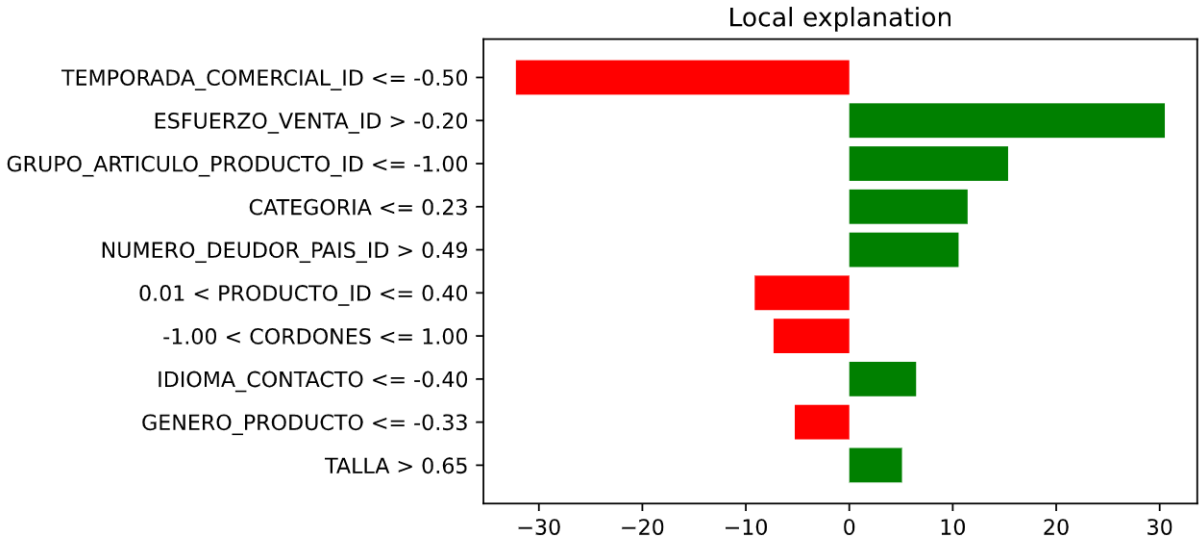

Figure 60: MLP LIME US input 2

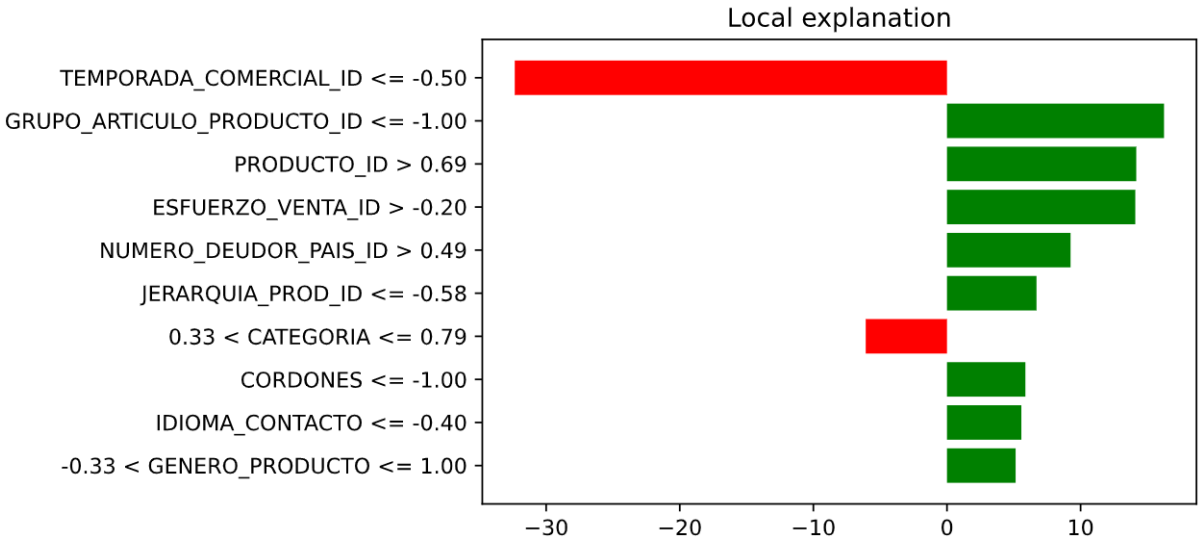

Figure 61: MLP LIME US input 3

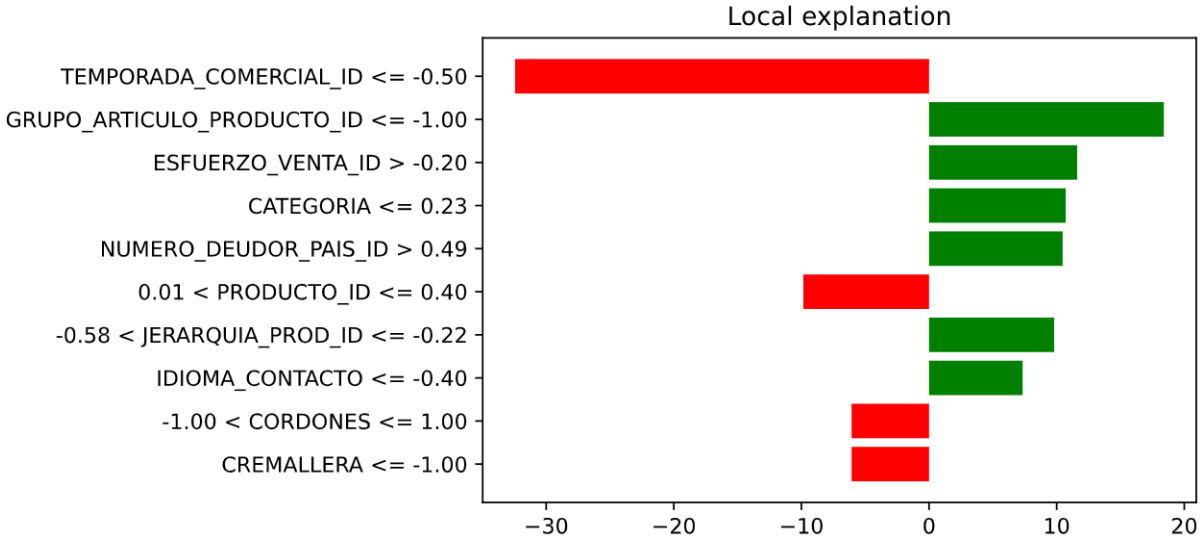

Figure 62: MLP LIME US input 4

Looking at the figures [55], [56], [57], [58], [59], [60], [61] and [62], we could observe some patterns. The variable TEMPORADA_COMERCIAL_ID has always negative influence in the local explanation among the input data due to the season (89) which is not the most successful in that year (the most successful was 87). In case that we had values closer to the ones in season 87, it would turn into a positive influence. Also, the variable GRUPO_ARTICULO_PRODUCTO_ID has always positive influence in the local explanation among the input data due to all the variables belong to the group 1 which it is Adult shoes. As we saw in the section [3.3.4], all the most products sold were from group 1. Therefore this variable has a positive impact in the prediction.
Another interesting observation was the different behaviour of the feature CATEGORIA. We found two cases, for the input data Spain 1, 2 and US 1, 2, 4, it has a positive influence

in the prediction, we are going to call it case A. Oppositely, for the input data Spain 3, 4 and US 3, it has negative influence, we are going to call it case B. All the shoes in case A belong to the category men but the input data Spain 3 from the case B belongs to the category men too, so why does CATEGORIA have a different influence if all of them have the same value? This happens because the men shoe from the case B is more expensive than the usual value for a shoe from that category. Furthermore, we could observe the same behaviour but with category women, in the input data Spain 4 and US 3 from the case B. In this example, the category has a negative influence in the input US 3 because it is cheaper than the usual value for a shoe from that category. Otherwise, for input data Spain 4, the category does not have a relevant influence such that it has an average import for that shoe.
Moreover, we could observe that NUMERO_DEUDOR_PAIS_ID in the case of the input data from Spain has a negative influence in all the cases, elsewhere we could observe the opposite in all the data from US. This means that those products sold in Spain decrease the value of the final prediction. The cause could be that these products are not well regarded in the Spanish market. Otherwise, the positive impact in the US market could be caused because these products are trendy in the US market and could be possible that the fluctuations of the currency affected the amount money earn.

Finally, we have tuned the algorithm from section 3.6.6 in the way that for each features will calculate the contribution 5 times. Because of the hardware limitations, we cannot increase the number of times.
Looking at the figure [63], we could observe that for this model the features related with the characteristics are the most influential. As an example, PLANTILLA_EXTRAIBLE and CORDONES would increase the average prediction. Also, we could observe that the customer language and city have a positive influence in the prediction. These features are important to get an accurate prediction. Although the features related to date, as year, season and day, are important, they are not as much relevant as the one related with the product.
Looking at the negative contributions, we could discard the features ESFUERZO_VENTA_ID, CATEGORIA and GRUPO_MODELO_PRODUCTO_ID, which they have $\varphi \geq -10$. Similar as **XGBoost**, these features do not add any valuable information. Although there are more negative features, we cannot remove them as their φ value is small. Increasing the number of times that a contribution is calculated, we would probably see how the features current φ value smaller than -10/-5 would get closer to 0. Otherwise, the φ from the features discarded would increase. But how do we know this? We have concluded this because we tested more than 25 times the procedure and we got the phenomenon aforementioned.

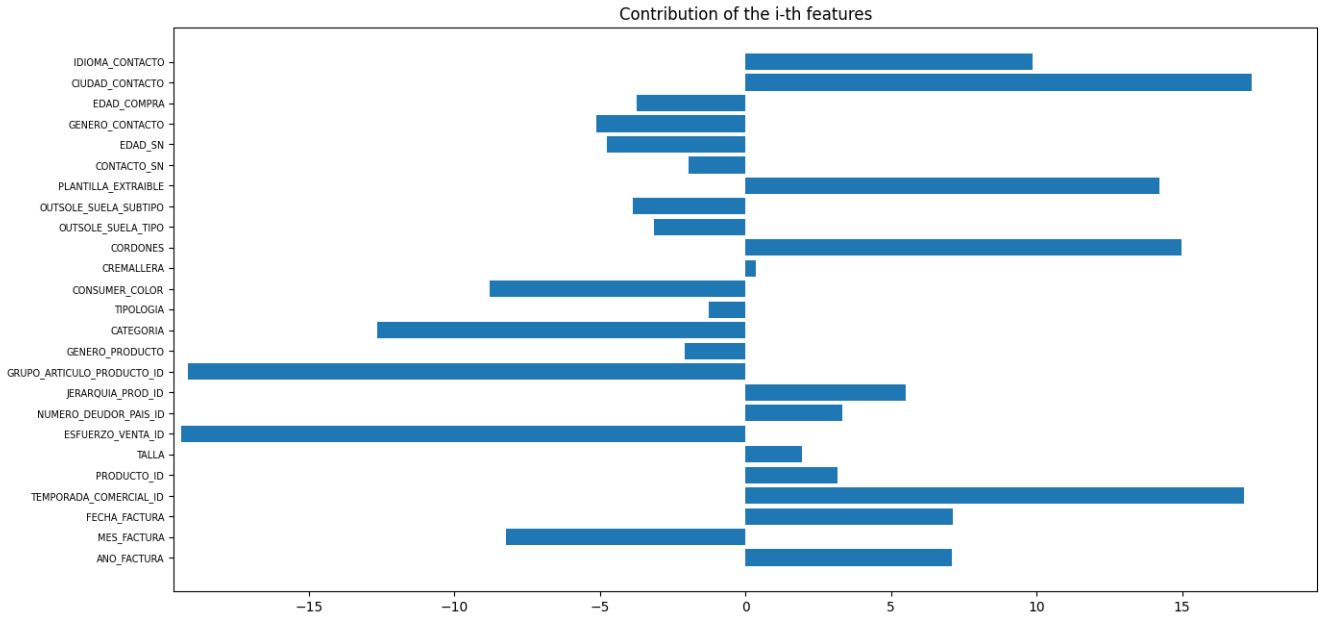

Figure 63: MLP contribution i-th feature

LSTM:

Adding the interpretability layer created using the package **Captum** [21], we are going to interpret what affects the prediction using the input data given in section [6.4].

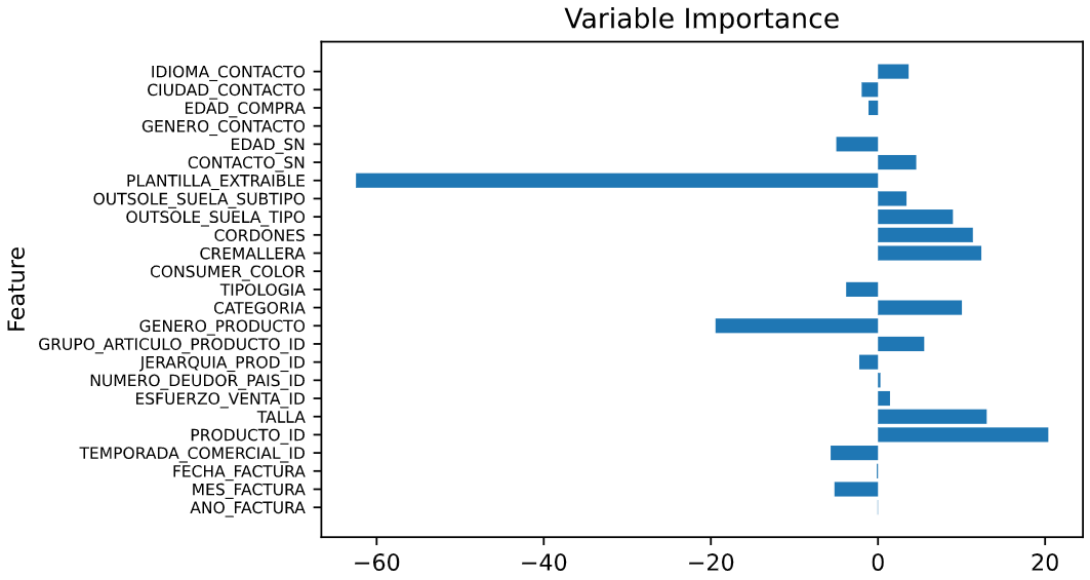

Figure 64: LSTM Captum Spain input 1

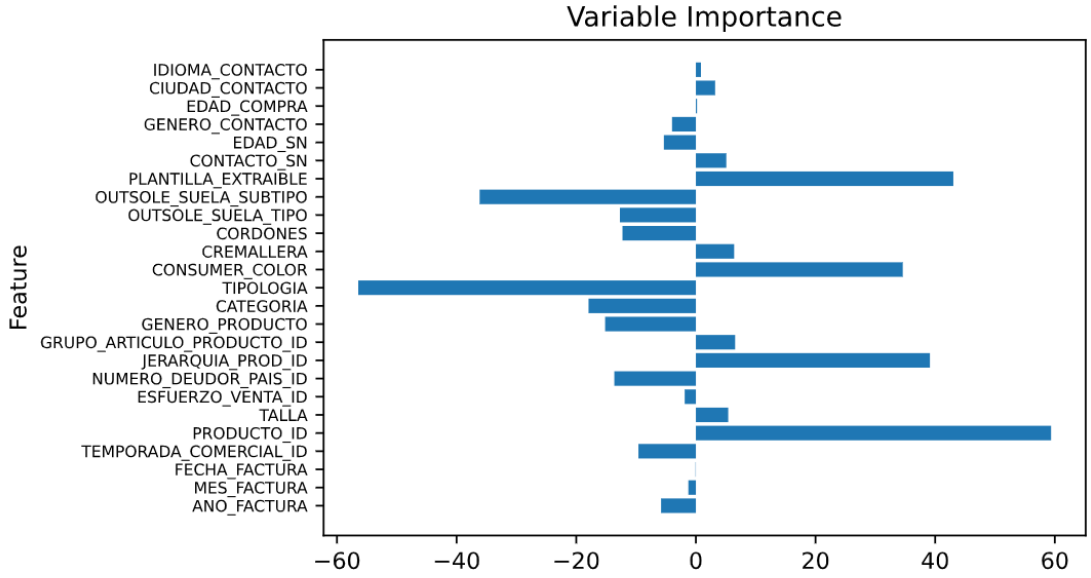

Figure 65: LSTM Captum Spain input 2

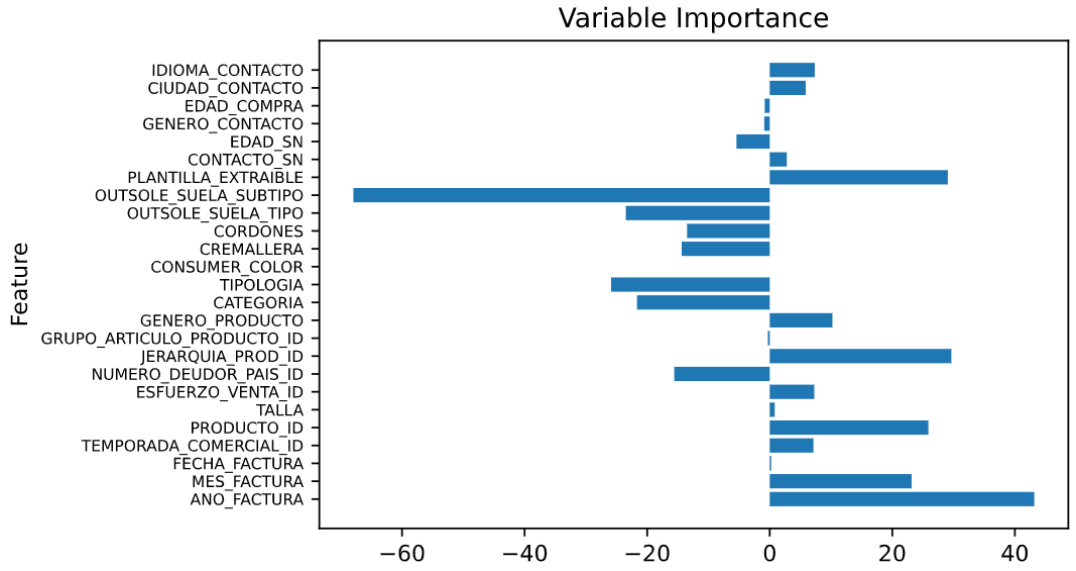

Figure 66: LSTM Captum Spain input 3

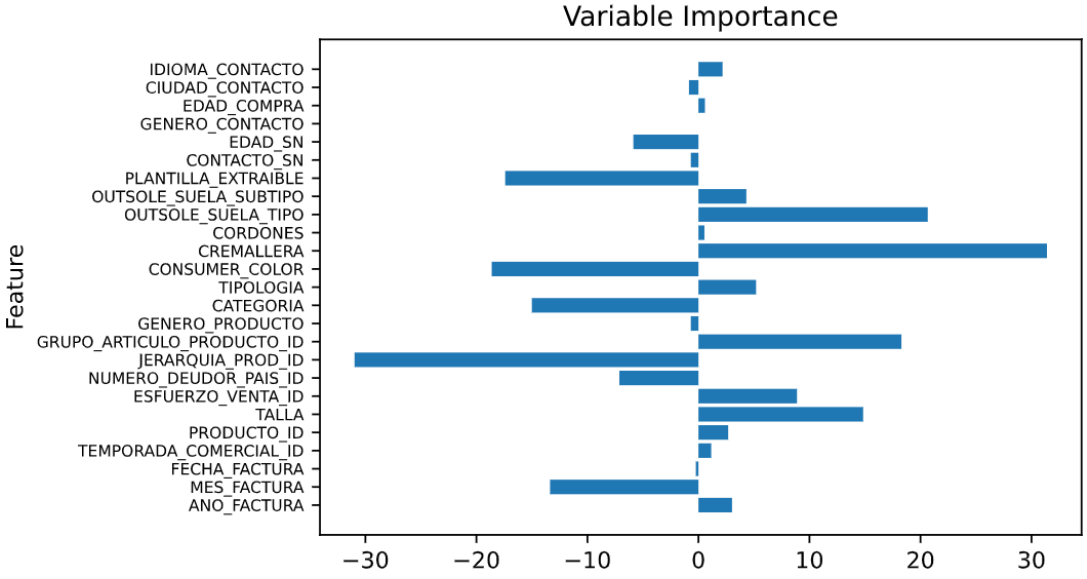

Figure 67: LSTM Captum Spain input 4

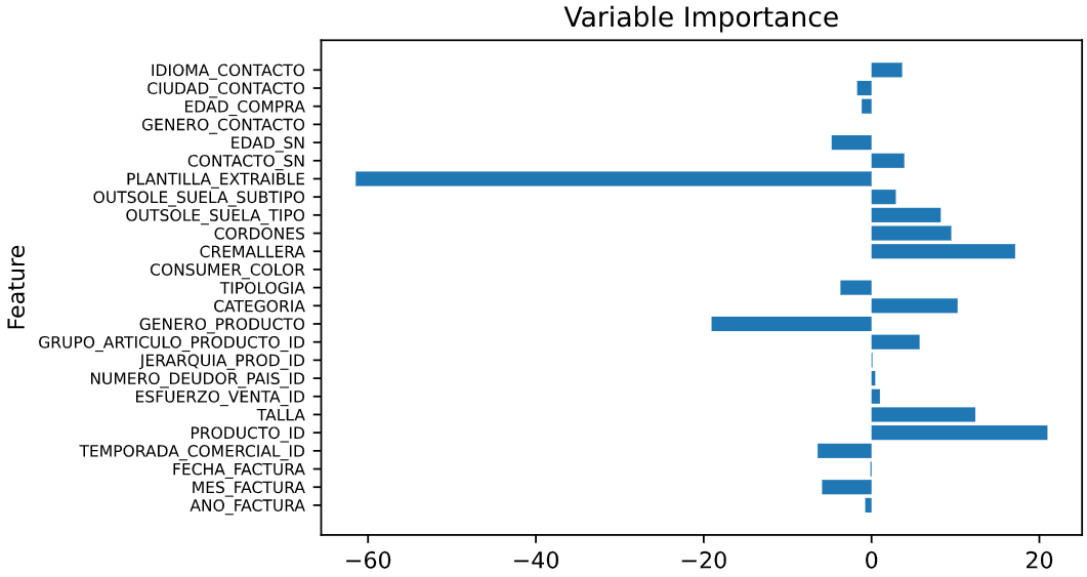

Figure 68: LSTM Captum US input 1

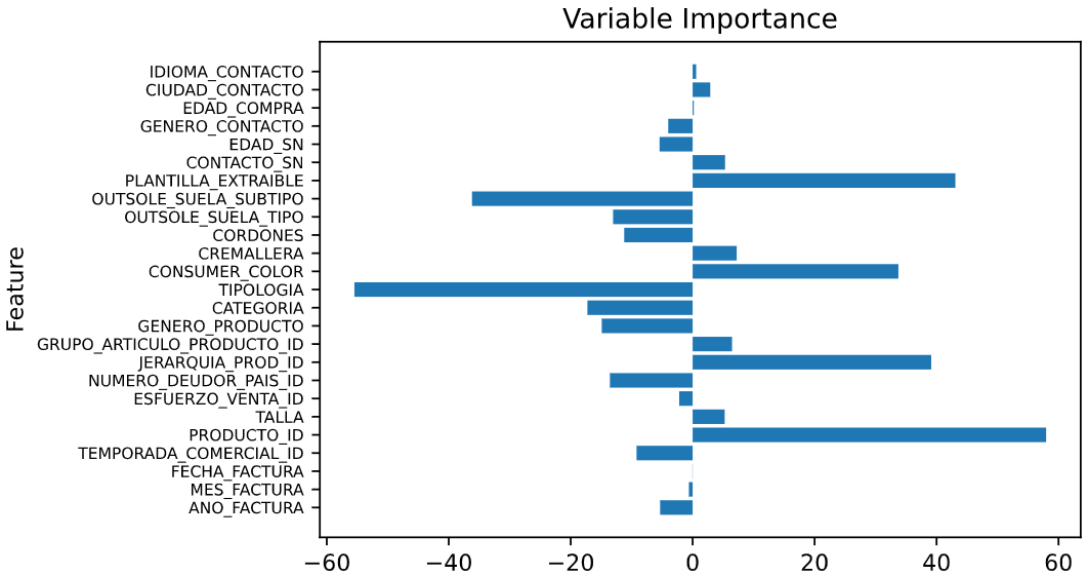

Figure 69: LSTM Captum US input 2

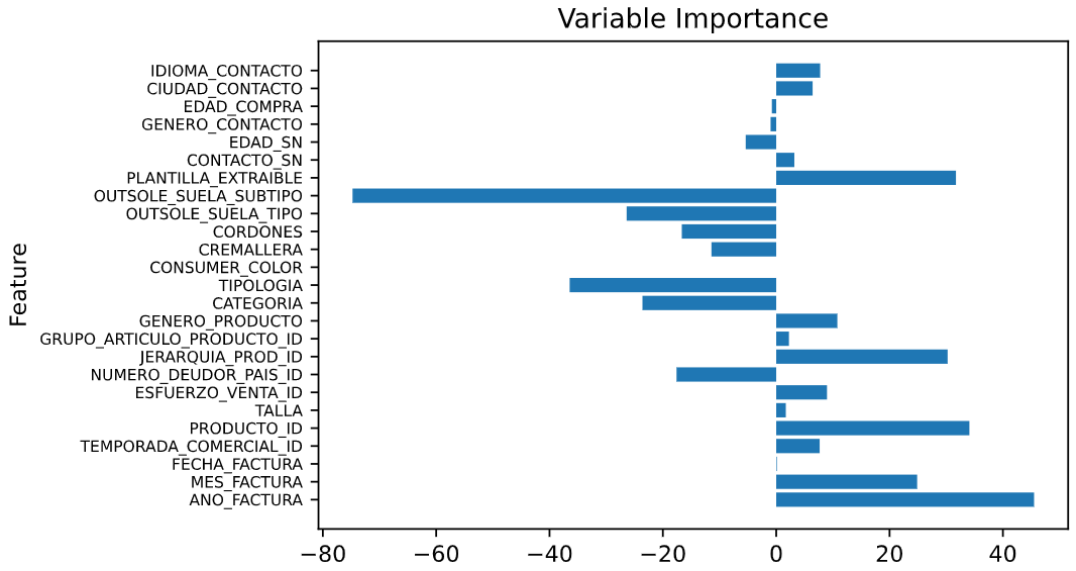

Figure 70: LSTM Captum US input 3

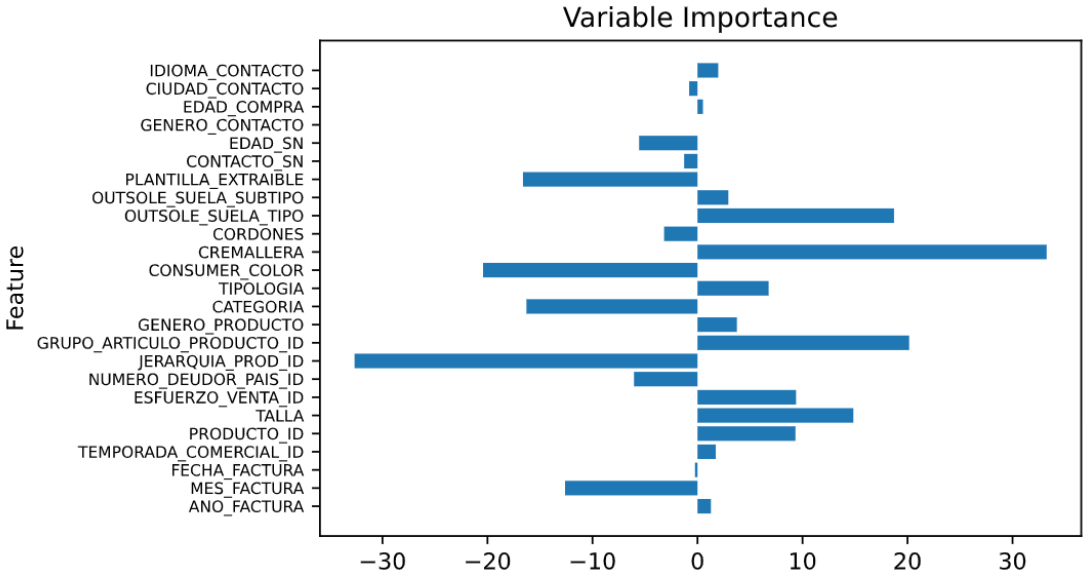

Figure 71: LSTM Captum US input 4

In the figures [64], [65], [66], [67], [68], [69], [70] and [71], we could extract that the features FECHA_FACTURA and GENERO_CONTACTO do not add valuable information in any test perfomed. So we could remove them and it would not change the finals results obtained but it would decrease the computation cost. In general terms, it could be seen that the features related to the characteristics of the product are the most influential.
Comparing the women ankle boots (figures [67] and [70]) we could see different importance for the features. PLANTILLA_EXTRAIBLE in [67] is contributing negatively to the final prediction but oppositely it is contributing positively in [70]. This means that if we had added a removable insole in the first case, we would have increased the value of the prediction for that shoe model, and it would have been easier to predict. Other features we got the opposite results are in OUTSOLE_SUELA_SUBTIPO, OUTSOLE_SUELA_TIPO and CORDONES. In [70], we could observe that these features had a negative contribution for the final prediction when in figure [67] they had a positive contribution, except for CORDONES that is close to zero. This implies that for predicting the final amount of money earned, the best option is to have heel high outsole.
We could also observe that for the model in figure [67], being black is an important factor in the shoe model.
Otherwise, when we analysed the men shoes, we saw how different affected the features for the different input data. Having PLANTILLA_EXTRAIBLE positive and negative contributions in the predictions. Also, in figure [68], having removable insole contributed negatively but in figure [66] we observed the opposite behaviour. In addition, we observed some correlations among the features. OUTSOLE_SUELA_SUBTIPO, OUTSOLE_SUELA_TIPO, CORDONES and CREMALLERA are correlated, they always have the same positive/negative contribution with different weight in the system. Oppositely, these features are negative correlated with PLANTILLA_EXTRAIBLE, they always have the opposite values in all cases.

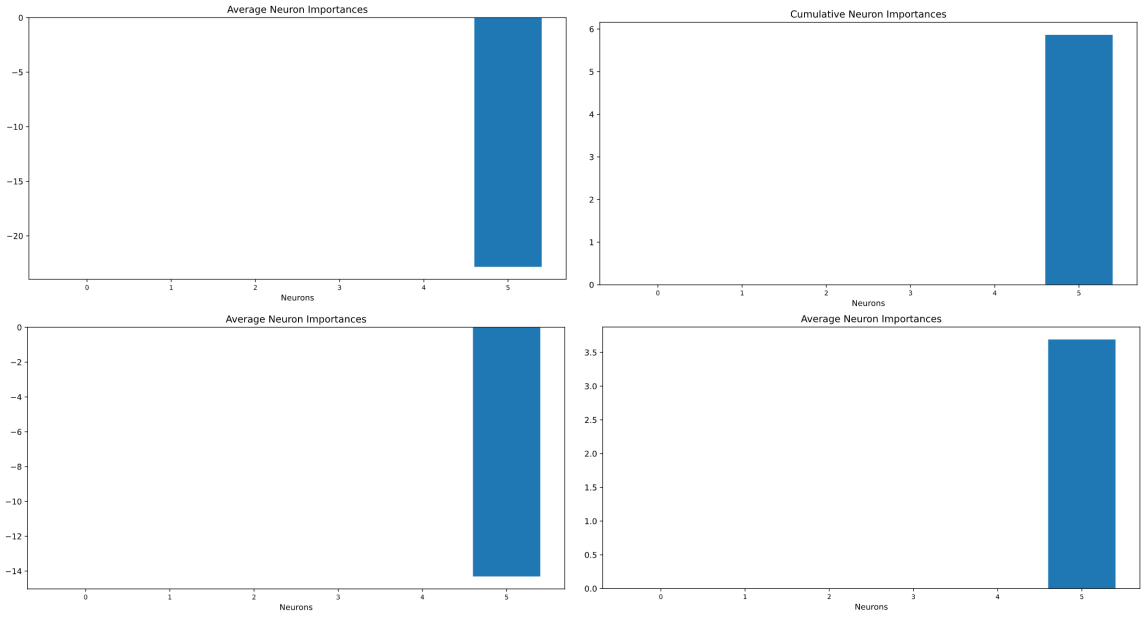

Figure 72: Contribution layer Spain (Input data 1, 2, 3 and 4)

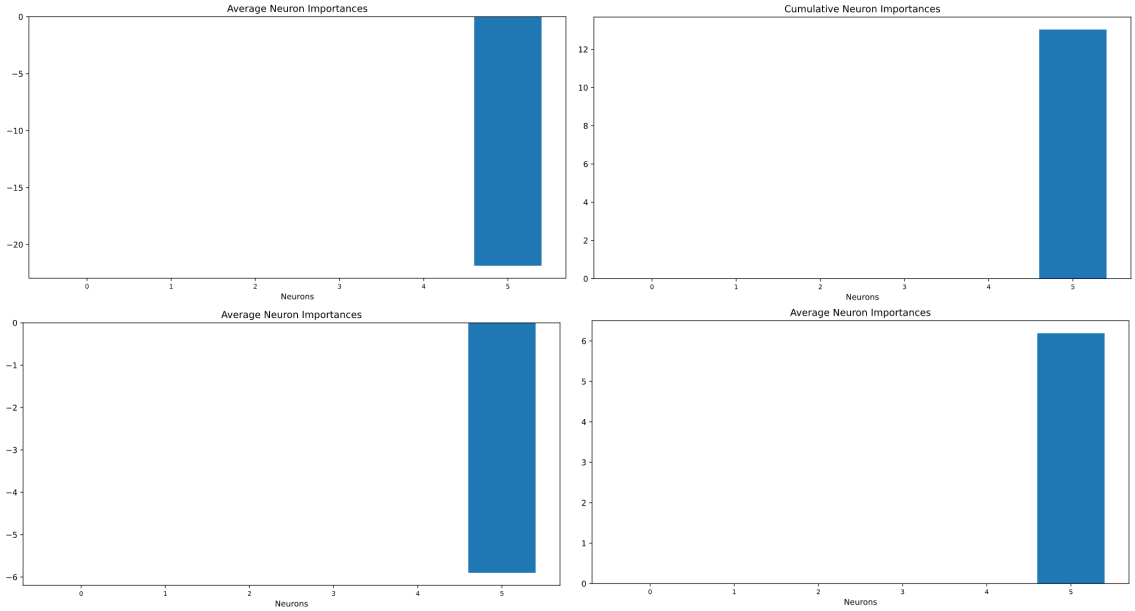

Figure 73: Contribution layer US (Input data 1, 2, 3 and 4)

In the figures [72] and [73], we could evaluate the contribution of each neuron in the first layer to the model output. We only plotted some neurons since the contribution mean of the 87 neurons that we had in the first layer is 0 except for neuron 5.
This means, that the neurons with mean 0 are not learning important features from the data. Otherwise, we are going to analyse this neuron 5 to extract what features are relevant in its process of learning.

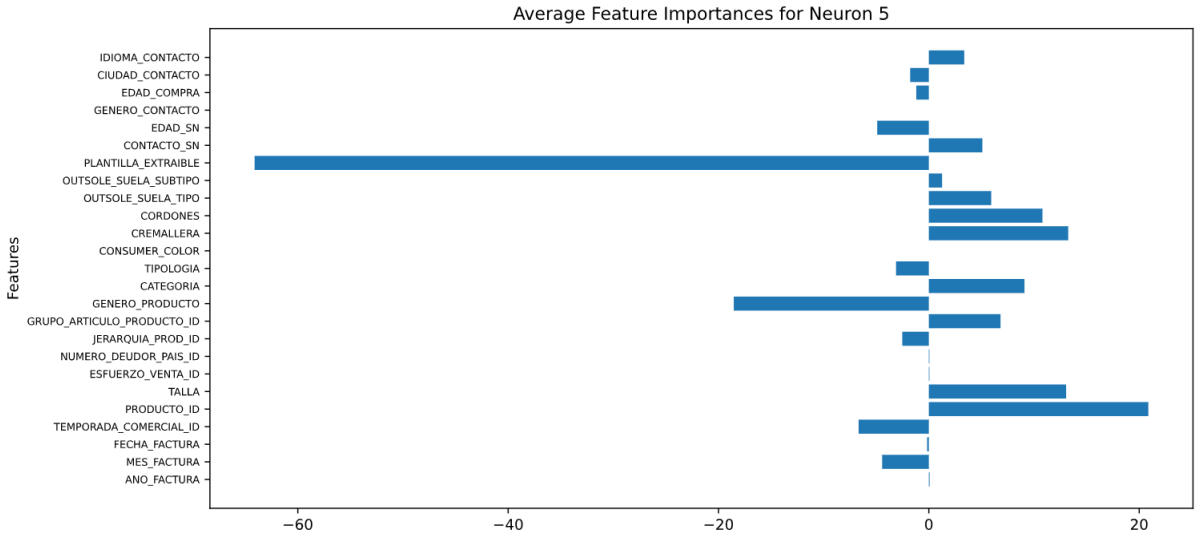

Figure 74: Neuron importance Spain input 1

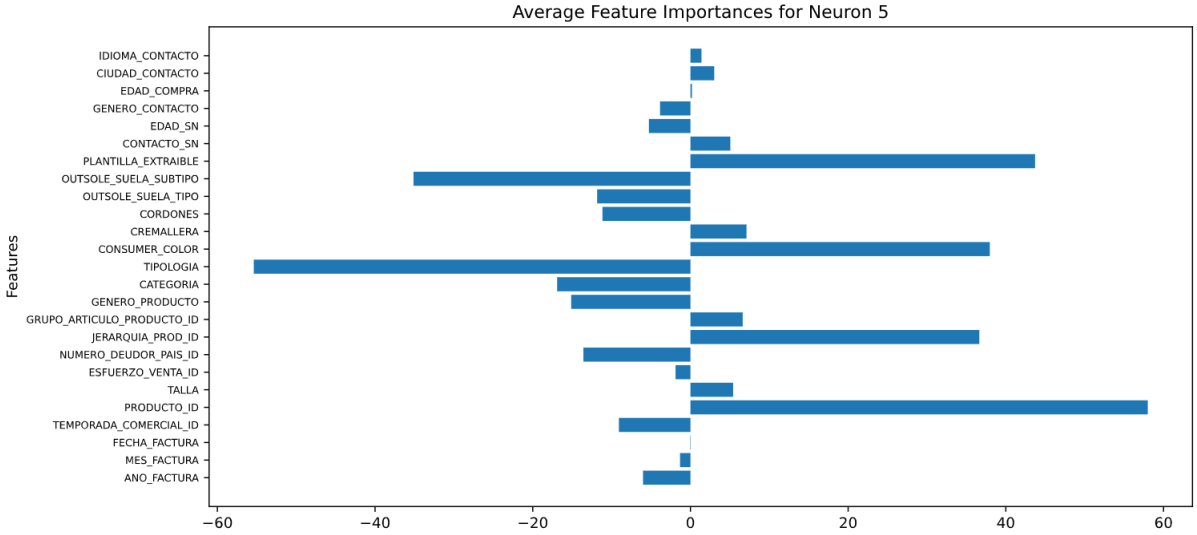

Figure 75: Neuron importance Spain input 2

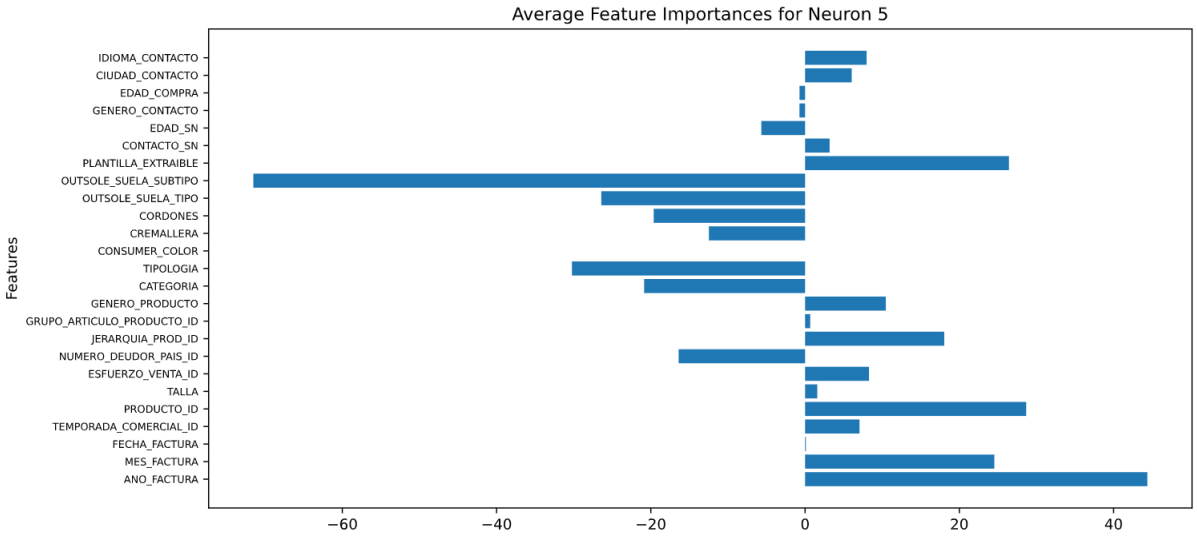

Figure 76: Neuron importance Spain input 3

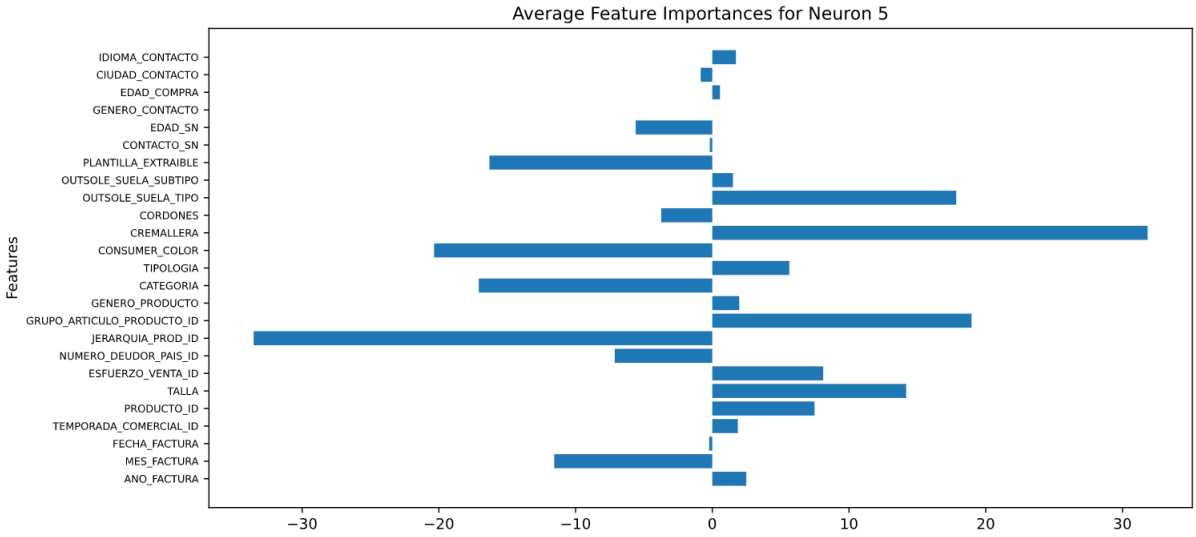

Figure 77: Neuron importance Spain input 4

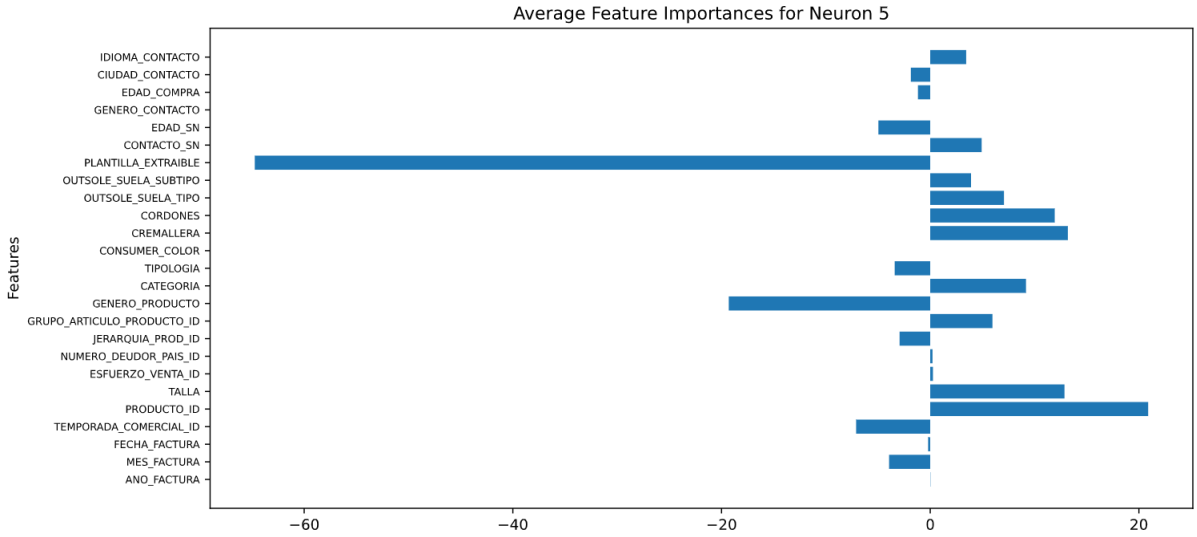

Figure 78: Neuron importance US input 1

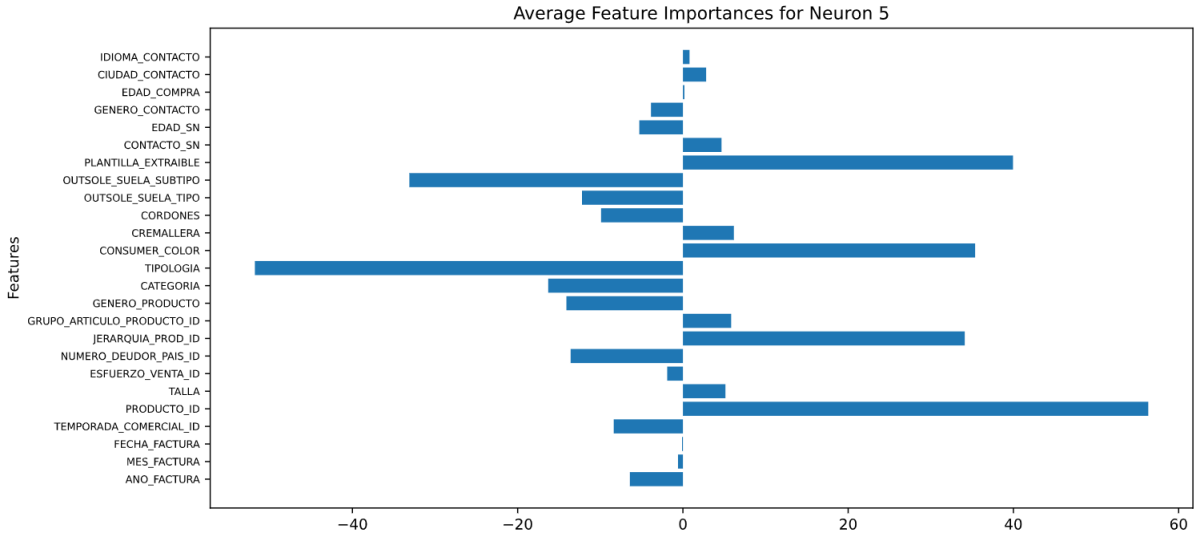

Figure 79: Neuron importance US input 2

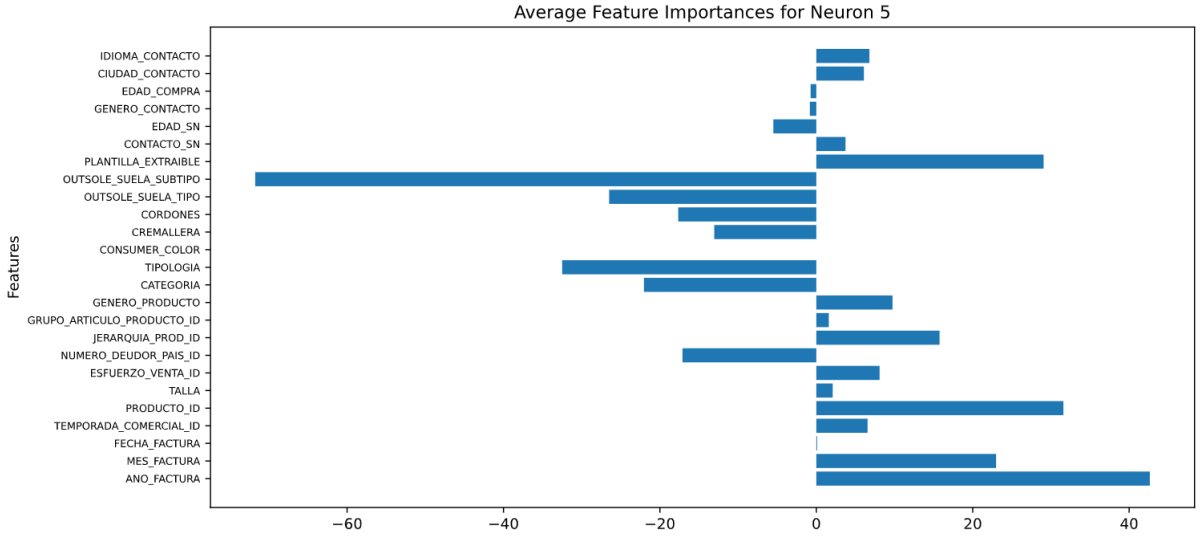

Figure 80: Neuron importance US input 3

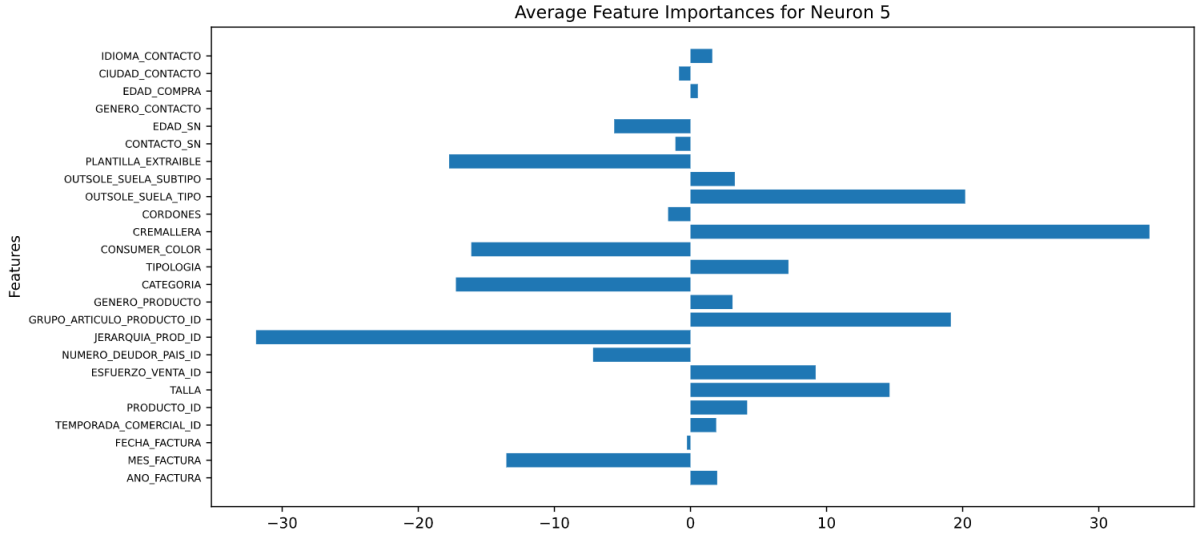

Figure 81: Neuron importance US input 4

In the figures [74], [75], [76], [77], [78], [79], [80] and [81], we could check the attribution of each input feature respect to the particular hidden neuron 5.
Looking at the figures [74] and [78], PLANTILLA_EXTRAIBLE is the primary input feature used by the neuron. This feature triples the value of the next features in terms of importance. But why this result? Checking the data, we could observe that for these products, they are only built without removable insole. This is the reason why it has this amount of importance, we could say that this feature is an identification feature of the products.
Examining the figures [75] and [79], we had CONTACTO_SN, PLANTILLA_EXTRAIBLE, CONSUMER_COLOR, TIPOLOGIA, and PRODUCTO_ID as the primary input features used by the neuron. As before, to understand these results, we had to check the original data, where we could observe that for these input data, these features had unique values. So this neuron understood that having these combinations of values in the features aforementioned are determinant to predict the value of the items.
Inspecting the figures [76] and [80], the primary input feature was OUTSOLE_SUELA_SUBTIPO. In this case, these products always had this feature with the same values. So the neuron is able to find this pattern and use it to predict the outcome of the items.
Taking a look into the figures [77] and [81], CREMALLERA, CONSUMER_COLOR, JERARQUIA_PROD_ID, OUTSOLE_SUELA_SUBTIPO, PLANTILLA_EXTRAIBLE and CATEGORIA. For this case, the neuron was able to relate these characteristics of the products with the type of the product it is and how those will affect with the final outcome.

Eventually, what we could extract from these results is that having the same characteristics for the same product, it will help the neurons to find hidden patterns among the features and it will increase the learning rate of the neurons, helping to the final outcome prediction. Otherwise, having different options for the same feature will hinder the learning process of the neuron.

5 Conclusions

The aim of this project was to perform a comparative study of demand forecasting models for an online retailer business, which we accomplished. We studied the data available by the company Camper [24], we used a methodology to be able to iterate towards our goals and finally, we managed to reach our objectives exposed in the introduction of this project. However we faced several problems, for instance, the data heterogeneity, that it could be seen in the exploratory data analysis (section 3.3) and also, the lack of data in several columns limited us.

We were able to assess the company's current business outlook, and also possibles open markets to explore, as the shoes for kids. Moreover, with the explainability layer, it helped to show what factors were important in the machine learning models, but more important it helped us to understand why one product with specific features will have more success than another. So that helped to understand the future businesses.

One of the main findings of our machine learning models analysis is that the **Hybrid approach** had the best results (section 4.3) in terms of error evaluation. Moreover, we tested this approach predicting the total earnings in a forecast window of 20 days. The model was able to keep the outcome prediction in the interval range calculated with the distributor estimator in the 78% of times, leading to a 22% of error. However, this does not seem a good result, and as we observed, probably due to the heterogeneity and the lack of data in several columns limited our forecast system. Despite that, comparing our final system results with similar approaches in the literature about demand forecasting in e-commerce, we realised that we ended up with a result of similar quality. For instance, looking at the results from [23] in table 9, our approach has three times fewer the RMSE value. Another example, observing the results from [13] in table 2, we got similar results with their best model. Therefore, our hybrid model is as competitive as current machine learning frameworks reported in the literature for demand forecasting.

Regarding software development, in order to implement this system we would strongly recommend to create a three layers based system, Data layer, Modelling layer and Explainability layer.

The Data layer would be the responsible of cleaning the data, being able to split the data into train, test and validate. Also, we recommend to add to this layer, filtering and grouping options to make easier the executions in the other layers. As we explained in 3.7, we would implement the functions **getIsocountry**, **getValuesfilter** , **getindexfilter** and **groupData**, to filter and group, and also, we would prepare the data using the functions **tagCustomer** and **getValue**.
We would use as a front-end a web application, to visualize the data, and micro-services system, some of them would be responsible for the web application and the others would be responsible of all data processes.

In the Modelling layer, we would use the **Hyperparameter optimization** procedure and the functionality called **removeFeatures** from 3.7 to select the best modelling

system. Looking at the results from 4.3, the **Hybrid approach** (section 3.4.6), would be the model to implement. Also, we can test different models using the **Error score** procedure explained in 3.7.
We recommend a cloud service based to implement this layer.

In the Explainability layer we would have two sub-layers, interval prediction and interpretability. In the first one, the user would be able to see the earning prediction given a specific input data during a certain number of days, using the **Distributor Estimator** procedure explained in 3.7. As before, in this case we recommend to have a web application to see the graphics and micro-services system containing the distributor estimator and the services for the web application. In the second one, the user would be able to see the variable importance given for a specific input data. All the function explained in 3.7 would be perfect for this purpose. As we aforementioned, we recommend to have a web application to see the results and micro-services to process the requests.

Finally, we suggest to use cloud web hosting services to implement the system mentioned before. However, they are sometimes a bit expensive, you will always have the assertion that you will never lose information, it is secure and there is always user assistance.

5.1 Possible Extensions

In terms of possible extensions, although we created a system able to categorise what customer should be more susceptible to buy a new product, we did not explore all the option that there are in the literature. One of the most famous is the method called **Bass diffusion** or **Bass model** [3], which is able to describe the process of how new products can be categorised in a population. Future works could explore this model and compare the results with the current system to determine if it adds value.

In this project, we have been exploring different modelling methods to create an expert system capable of predicting the earnings given an input data. The methods explored were meticulously chosen from the literature, but there was one which we could not exploit, **Gaussian Process Regression**. Even though we explained the reasons why this model was not suitable for this case of study (section [3.4.5]), we found different articles that adapted the algorithm to be able to work with Big Data. We strongly recommend to read the articles [10] and [7] where the authors introduce the approaches, and finally we invite you to explore them in this case of study.

In section 3.6.6, we explained how to achieve the algorithm exposed in the paper [4], we coded it in a sequential composition, so it led to fit the models sequentially, spending the double of the time. Even though the algorithm does not have a high computational cost, the speed resides in how long the models take to run, so we strongly recommend to parallelize the fitting process. This would reduce the execution time to at least half.

6 Appendix A: Datasets

The main purpose of this appendix is to document the data sources used in this project. All the data is provided by Camper. We cannot share the content of the datasets because we have signed a non disclosure agreement.
The data have been recollected during a period of three years (2018, 2019 and 2020) from different sources, intern databases and web scraping.

6.1 Sales' dataset

This is the main dataset used in the project, we can find information regarding to the online sales that Camper has had during the period of time. It contains different information related to the customer and features of the products.

Feature	Description
FACTURA_ID	Number of invoice
FACTURA_POSICION_ID	Invoice position number
CUSTOMER_ID	Identification code of the customer
FACTURA_CLASE_DOCUMENTO_ID	Type of the invoice
ANO_MES_FACTURA	Year and month concatenated
ANO_FACTURA	Year
MES_FACTURA	Month
FECHA_FACTURA	Full date
IMP_VENTA_NETO_EUR	Sale amount €
CANAL_VENTA_ID	Channel of sale
CANAL_VENTA_DESC	Description of sales channel
TEMPORADA_COMERCIAL_ID	Commercial season
TEMPORADA_COMERCIAL_DESC	Description of commercial season
PRODUCTO_ID	Code of the product, SKU without size
TALLA	Size
MATERIAL_ID	Full SKU (Product id + Size)
ESFUERZO_VENTA_ID	Who/What shop sales
ESFUERZO_VENTA_DESC	Description of the sale effort
NUMERO_DEUDOR	Identification code of who pay
NUMERO_DEUDOR_PAIS_ID	Sale Country
NUMERO_DEUDOR_PAIS_DESC	Description of the sale country
VENTA_DEVOLUCION	Sale or return
JERARQUIA_PROD_ID	Hierarchical code of the product
GRUPO_ARTICULO_PRODUCTO_ID	Group code item
GRUPO_ARTICULO	Description of item code group
CONCEPTO	Concept
LINEA	Line
GENERO_PRODUCTO	Gender of the product

Table 15: Features dataset (Part I)

Feature	Description
CATEGORIA	Category
TIPOLOGIA	Tipology
COLOR	Intern colour
CONSUMER_COLOR	Consumer colour
CREMALLERA	Zipper (Yes/No)
CORDONES	Laces (Yes/No)
OUTSOLE_SUELA_TIPO	Indicates the type of the sole based on the height
OUTSOLE_SUELA_SUBTIPO	Indicates the degree of height
PLANTILLA_EXTRAIBLE	Indicates whether the shoe insole is removable or not
CONTACTO_SN	Indicates if there's contact or not
EDAD_SN	Indicates if there's specified the age of the customer
GENERO_CONTACTO	Gender of the customer
EDAD_COMPRA	Age of the customer
EDAD_RANGO_COMPRA	Range of age
PAIS_CONTACTO	Customer's country
PAIS_CONTACTO_DESC	Description of customer country
CIUDAD_CONTACTO	Customer's city
IDIOMA_CONTACTO	Customer's language

Table 16: Features dataset (Part II)

6.2 Structure Sales' dataset

This dataset contains the information related to each feature we will find in the previous dataset.

6.3 Product hierarchy's dataset

This dataset contains the information related to the product itself, it adds valuable information to understand the data from the main dataset.

6.4 Interpretability input data

	Spain input 1	Spain input 2
ANO_FACTURA	2020	2020
MES_FACTURA	1	1
FECHA_FACTURA	19	20
IMP_VENTA_NETO_EUR	107.44	152.89
TEMPORADA_COMERCIAL_ID	89	89
PRODUCTO_ID	K100360-006	16002-194
TALLA	40	41
ESFUERZO_VENTA_ID	2381974.0	2381974.0
NUMERO_DEUDOR_PAIS_ID	ES	ES
JERARQUIA_PROD_ID	101PIXPIX0HK100360	101PELPELAH16002
GRUPO_ARTICULO_PRODUCTO_ID	1.0	1.0
GENERO_PRODUCTO	MEN	MEN
CATEGORIA	Men Shoe	Men Shoe
TIPOLOGIA	Lace Up Shoe	Oxford
CONSUMER_COLOR	Multicolour	Brown
CREMALLERA	NO	NO
CORDONES	YES	YES
OUTSOLE_SUELA_TIPO	FLAT	FLAT
OUTSOLE_SUELA_SUBTIPO	FLAT	FLAT
PLANTILLA_EXTRAIBLE	NO	NO
CONTACTO_SN	YES	YES
EDAD_SN	YES	YES
GENERO_CONTACTO	NV	MAN
EDAD_COMPRA	31	50
EDAD_RANGO_COMPRA	30-39	50-59
CIUDAD_CONTACTO	Barcelona	Zaragoza
IDIOMA_CONTACTO	EN	ES

Table 17: Input values Spain (1 and 2)

	Spain input 3	Spain input 4
ANO_FACTURA	2020	2020
MES_FACTURA	1	1
FECHA_FACTURA	20	21
IMP_VENTA_NETO_EUR	118.8	118.54
TEMPORADA_COMERCIAL_ID	89	89
PRODUCTO_ID	K300072-014	K400381-001
TALLA	41	38
ESFUERZO_VENTA_ID	2381974.0	2381974.0
NUMERO_DEUDOR_PAIS_ID	ES	ES
JERARQUIA_PROD_ID	101DUBDUB0HK300072	101WEEWEE0MK400381
GRUPO_ARTICULO_PRODUCTO_ID	1.0	1.0
GENERO_PRODUCTO	MEN	WOMEN
CATEGORIA	Men Sneaker Boot	Women Ankle Boot
TIPOLOGIA	Sneaker bootie	Lace Up Bootie
CONSUMER_COLOR	Multicolour	Black
CREMALLERA	NO	YES
CORDONES	NO	YES
OUTSOLE_SUELA_TIPO	FLAT	HEEL
OUTSOLE_SUELA_SUBTIPO	FLAT	HIGH
PLANTILLA_EXTRAIBLE	YES	NO
CONTACTO_SN	YES	YES
EDAD_SN	YES	YES
GENERO_CONTACTO	MAN	NV
EDAD_COMPRA	44	50
EDAD_RANGO_COMPRA	40-49	50-59
CIUDAD_CONTACTO	Zamora	Madrid
IDIOMA_CONTACTO	ES	ES

Table 18: Input values Spain (3 and 4)

	US input 1	US input 2
ANO_FACTURA	2020	2020
MES_FACTURA	1	1
FECHA_FACTURA	19	19
IMP_VENTA_NETO_EUR	96.46	80.38
TEMPORADA_COMERCIAL_ID	89	89
PRODUCTO_ID	K100356-010	K100467-001
TALLA	45	43
ESFUERZO_VENTA_ID	2381974.0	2381974.0
NUMERO_DEUDOR_PAIS_ID	US	US
JERARQUIA_PROD_ID	101BILBIL0HK100356	101TWSKTI0HK100467
GRUPO_ARTICULO_PRODUCTO_ID	1.0	1.0
GENERO_PRODUCTO	MEN	MEN
CATEGORIA	Men Shoe	Men Sneaker
TIPOLOGIA	Basket Shoe	Sneaker
CONSUMER_COLOR	Grey	Black
CREMALLERA	NO	SI
CORDONES	YES	NO
OUTSOLE_SUELA_TIPO	FLAT	FLAT
OUTSOLE_SUELA_SUBTIPO	FLAT	FLAT
PLANTILLA_EXTRAIBLE	YES	NO
CONTACTO_SN	YES	YES
EDAD_SN	YES	NO
GENERO_CONTACTO	MAN	NV
EDAD_COMPRA	62	0
EDAD_RANGO_COMPRA	60-69	NV
CIUDAD_CONTACTO	Sant Louis	San Jose
IDIOMA_CONTACTO	EN	EN

Table 19: Input values US (1 and 2)

	US input 3	US input 4
ANO_FACTURA	2020	2020
MES_FACTURA	1	1
FECHA_FACTURA	19	20
IMP_VENTA_NETO_EUR	110.31	88.69
TEMPORADA_COMERCIAL_ID	89	89
PRODUCTO_ID	K300072-014	K100397-018
TALLA	41	43
ESFUERZO_VENTA_ID	2381974.0	2381974.0
NUMERO_DEUDOR_PAIS_ID	US	US
JERARQUIA_PROD_ID	101IMNIMN0MK400299	101PELPELXHK100397
GRUPO_ARTICULO_PRODUCTO_ID	1.0	1.0
GENERO_PRODUCTO	WOMEN	MEN
CATEGORIA	Women Ankle Boot	Men Shoe
TIPOLOGIA	Chelsea Bootie	Basket
CONSUMER_COLOR	Brown	Black
CREMALLERA	NO	NO
CORDONES	NO	YES
OUTSOLE_SUELA_TIPO	FLAT	FLAT
OUTSOLE_SUELA_SUBTIPO	FLAT	FLAT
PLANTILLA_EXTRAIBLE	NO	YES
CONTACTO_SN	YES	YES
EDAD_SN	YES	YES
GENERO_CONTACTO	NV	NV
EDAD_COMPRA	72	48
EDAD_RANGO_COMPRA	72++	40-49
CIUDAD_CONTACTO	Kansas City	Modesto
IDIOMA_CONTACTO	EN	EN

Table 20: Input values US (3 and 4)

References

[1] J. Armstrong and Kesten Green. "Demand Forecasting: Evidence-Based Methods". In: *SSRN Electronic Journal* (Oct. 2005). DOI: 10.2139/ssrn.3063308.

[2] C. K. I. Williams C. E. Rasmussen. *Gaussian Processes for Machine Learning*. Massachusetts Institute of Technology. the MIT Press, 2006. ISBN: 026218253X.

[3] Bass's Basement Research Institute. *Bass's Basement Research Institute*. 2010. URL: http://www.bassbasement.org/BassModel/.

[4] Erik Strumbelj and Igor Kononenko. "An Efficient Explanation of Individual Classifications Using Game Theory". In: *J. Mach. Learn. Res.* 11 (Mar. 2010), 1–18. ISSN: 1532-4435.

[5] F. Pedregosa et al. "Scikit-learn: Machine Learning in Python". In: *Journal of Machine Learning Research* 12 (2011), pp. 2825–2830.

[6] Hakyeon Lee et al. "Pre-launch new product demand forecasting using the Bass model: A statistical and machine learning-based approach". In: *Technological Forecasting and Social Change* 86 (2014), pp. 49–64. ISSN: 0040-1625. DOI: https://doi.org/10.1016/j.techfore.2013.08.020. URL: https://www.sciencedirect.com/science/article/pii/S0040162513001881.

[7] Sourish Das, Sasanka Roy, and Rajiv Sambasivan. "Fast Gaussian Process Regression for Big Data". In: *Big Data Research* 14 (Sept. 2015). DOI: 10.1016/j.bdr.2018.06.002.

[8] Kenneth Jensen. *CRISP-DM Process Diagram*. 2016. URL: https://upload.wikimedia.org/wikipedia/commons/thumb/b/b9/CRISP-DM_Process_Diagram.png/512px-CRISP-DM_Process_Diagram.png.

[9] Marco Tulio Ribeiro, Sameer Singh, and Carlos Guestrin. ""Why Should I Trust You?": Explaining the Predictions of Any Classifier". In: *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, San Francisco, CA, USA, August 13-17, 2016*. 2016, pp. 1135–1144.

[10] Maziar Raissi. "Parametric Gaussian Process Regression for Big Data". In: (Apr. 2017).

[11] Chan HL. Ram P. Ren S. "A Comparative Study on Fashion Demand Forecasting Models with Multiple Sources of Uncertainty". In: *Ann Oper Res* 257 (2017), pp. 335–355. DOI: https://doi.org/10.1007/s10479-016-2204-6.

[12] R.J. Hyndman and G. Athanasopoulos. *Forecasting: principles and practice*. 2018. URL: https://otexts.com/fpp2/counts.html.

[13] Ji S. Liu G. Li M. "Forecasting of Chinese E-Commerce Sales: An Empirical Comparison of ARIMA, Nonlinear Autoregressive Neural Network, and a Combined ARIMA-NARNN Model". In: *Mathematical Problems in Engineering* (Nov. 2018). ISSN: 1024-123X. DOI: https://doi.org/10.1155/2018/6924960.

[14] Tim Royston-Webb. "Propensity Modelling for Business". In: *Data Science Foundation* (2018), pp. 7–8.

[15] DeepGio. *MLP nn*. 2019. URL: https://github.com/deepGio/ANNFromScratch/blob/master/images/nn.png.

[16] Adam Paszke et al. "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., 2019, pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

[17] Andrew J Tiffin. "Machine Learning and Causality: The Impact of Financial Crises on Growth". In: *IMF Working Papers* (2019).

[18] Hossein Abbasimehr, Mostafa Shabani, and Mohsen Yousefi. "An optimized model using LSTM network for demand forecasting". In: *Computers Industrial Engineering* 143 (2020), p. 106435. ISSN: 0360-8352. DOI: https://doi.org/10.1016/j.cie.2020.106435. URL: https://www.sciencedirect.com/science/article/pii/S0360835220301698.

[19] Hubert Baniecki et al. "dalex: Responsible Machine Learning with Interactive Explainability and Fairness in Python". In: (2020). arXiv: 2012.14406. URL: https://arxiv.org/abs/2012.14406.

[20] Grisoni F. Schneider G. Jiménez-Luna J. "Drug discovery with explainable artificial intelligence". In: *Nat Mach Intell* 2 (2020), pp. 573–584. DOI: https://doi.org/10.1038/s42256-020-00236-4.

[21] Narine Kokhlikyan et al. *Captum: A unified and generic model interpretability library for PyTorch*. 2020. arXiv: 2009.07896 [cs.LG].

[22] Raul Lorenzo Villagrasa. *Demand Forecasting*. 2020. URL: https://github.com/rauls19/DemandForecasting.

[23] Xie G. Zhao W. Gu Y. Huang Y Huang L. "Regional logistics demand forecasting: a BP neural network approach". In: *Complex Intelligent Systems* (Mar. 2021). ISSN: 2198-6053. DOI: https://doi.org/10.1007/s40747-021-00297-x.

[24] Camper. URL: https://www.camper.com.