NOVA IMS Information Management School

MDSAA

Master Degree Program in

Data Science and Advanced Analytics

Machine Learning in Digital Retail

Demand Forecasting for Inventory Management in a Sportswear Company

Andreia da Silva Tavares Bastos

Internship Report

presented as partial requirement for obtaining a Master's Degree in Data Science and Advanced Analytics

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação
Universidade Nova de Lisboa

NOVA Information Management School

Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa

Machine Learning in Digital Retail

Demand Forecasting for Inventory Management in a Sportswear Company

by

Andreia da Silva Tavares Bastos

Internship Report presented as partial requirement for obtaining the Master's degree in Data Science and Advanced Analytics, with a specialization in Business Analytics

Supervised by

Mauro Castelli, PhD, Nova University

May, 2024

STATEMENT OF INTEGRITY

I hereby declare having conducted this academic work with integrity. I confirm that I have not used plagiarism or any form of undue use of information or falsification of results along the process leading to its elaboration. I further declare that I have fully acknowledged the Rules of Conduct and Code of Honor from the NOVA Information Management School.

Portugal, 28-05-2024

DEDICATION

This project is dedicated to my parents and brother, my inspirations in life. To my boyfriend as well, without whom I could not have succeeded. Your love and support meant the world to me. And last but not least, to my dear friends, who were always there for me. Thank you for everything.

ACKNOWLEDGEMENTS

I would like to thank my supervisor Mauro Castelli for the support and encouragement throughout this process. I also wish to extend my gratitude to my team, I couldn't have done without you.

ABSTRACT

Several domains benefit from accurate forecasts, especially within the retail sector, as precise demand forecasting plays a key role in making informed decisions, such as for inventory management. The focus of this project is to improve the safety stock process for Always Available (AA) products in the digital channel in a Sportswear company, by implementing a machine learning solution to forecast demand units. By providing an accurate forecast, the project aims to optimize inventory levels, ensuring that sufficient stock is maintained to meet customer demand without overstocking. This approach focused on the implementation of tree-based models, specifically Random Forest, XGBoost, and LightGBM, which were evaluated using two performance metrics: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE). Various factors influenced the models' performance, including promotions, inventory levels, website traffic, conversion rates, etc. All three algorithms demonstrated similar MAE scores, highlighting their effectiveness in handling complex and large datasets and showing that tree-based models can provide highly effective solutions in a retail context. However, LightGBM emerged as the most robust method, achieving the lowest RMSE and thereby indicating a slightly superior performance. This project demonstrates the significant benefit that machine learning can bring to the retail sector, driving business growth.

KEYWORDS

Digital Retail; Machine Learning; Tree-Based Regression Algorithms; Inventory Management.

Sustainable Development Goals (SDG):

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS AND ACRONYMS

B&M	Brick & Mortar
AA	Always Available
SKU	Stock Keeping Unit
ES	Exponential Smoothing
ARIMA	Autoregressive Integrated Moving Average
HWES	Holt Winter Exponential Smoothing
ARIMAX	Autoregressive Integrated Moving Average with external variables
SARIMA	Seasonal Autoregressive Integrated Moving Average
SARIMAX	Seasonal Autoregressive Integrated Moving Average with external variables
ARMAX	Autoregressive Moving Average with external variables
LR	Linear Regression
MLR	Multiple Linear Regression
DT	Decision Tree
RF	Random Forest
SVM	Support Vector Machine
K-NN	K-Nearest Neighbors
ETR	Extra Ree Regression
LightGBM	Light Gradient Boosting Machine
XGBoost	eXtreme Gradient Boosting
LSTM	Long-Short Term Memory
NN	Neural Network
MAPE	Mean Absolute Percentage Error
MAE	Mean Absolute Error
RMSE	Root Mean Squared Error
CRISP-DM	Cross Industry Standard Process for Data Mining

SQL	Structured Query Language
RRP	Recommended Retail Price
CRP	Current Retail Price
OHE	One Hot Encoding
CSV	Comma Separated Value
POS	Point of Sale

1. INTRODUCTION

1.1. CONTEXT

Multiple domains benefit from accurate forecasts since precise predictions not only facilitate decision-making but also mitigate the uncertainties associated with it (Punia et al., 2020). Makridakis and Hilbon (2000) said it best: “Predictions remain the foundation of all science”. Especially within the retail sector, precise demand forecasting plays a pivotal role in making informed decisions in purchasing, inventory management, scheduling, capacity management, assortment planning, etc. (Punia et al., 2020).

By 2024, global retail e-commerce sales are projected to surpass 6.3 trillion U.S. dollars, with further growth anticipated in the subsequent years (Statista, 2024). Moreover, in this era of big data and technology, where e-commerce stands as the primary channel and continues to grow rapidly, businesses are encouraged to rethink their forecasting methods to make accurate data-driven decisions, since an intuition-based methodology is no longer sufficient (Lalou et al., 2020). In today's digital world, retail demand forecasting systems must be fast, reliable, and more efficient. Especially when it comes to inventory management, an accurate forecast can avoid overstocking or understocking issues (Falatouri et al., 2022). However, despite the importance of the digital channel, existing literature on retail demand forecasting is still mainly focused on offline retail scenarios. This project aims to address the gap in the existing literature by developing a machine learning approach to forecast demand in a digital retail setting, exploring its challenges and learning opportunities.

Historically, time-series techniques stand out as the most frequently used methods for retail demand forecasting, such as Exponential Smoothing (ES), ARIMA (Autoregressive Integrated Moving Average), etc. Whilst satisfactory in many cases, these approaches face challenges when faced with nonlinear relationships between variables, which are common in real-life retail scenarios (Punia et al., 2020). To improve accuracy, recent studies focus on more advanced forecasting techniques, from regression-based to deep learning models, and even propose hybrid approaches.

1.2. MOTIVATION AND OBJECTIVES

This study was conducted within a Sportswear Retail company, where the focus of the business shifted after the COVID-19 epidemic, as the digital channel experienced a boom and Brick & Mortar (term designated for a business with physical presence, i.e., offline channel) slowed down. Year after year, digital has outpaced Brick & Mortar (B&M), growing at a double-digit pace. Thus, there was a need to update and improve its processes, implementing machine learning solutions, especially for inventory management.

For specific products designated as AA (Always Available), there is a unique safety stock allocated for all channels retailing these items, with priority given to the first entity that books

the units. Hence, it's crucial to predict future demand for these items, but more importantly, to do it as accurately and as fast as possible to avoid stock-out situations, among other issues.

The past demand forecast process for the AA products in the digital channel was solely based on the intuition and instincts of the employees, built upon their experience and market expertise. As mentioned before, this technique is no longer sufficient, since today's businesses need faster and more efficient methodologies for decision-making, especially to be able to deal with high volumes of data. Therefore, the goal is to develop and maintain a forecasting model for AA products sold online, projecting sales for the next 28 days at a daily x SKU (Stock Keeping Unit) level. This aims to anticipate future sales and determine the need and quantity for additional units. This allows the business to optimize inventory levels so shortages can be avoided by booking the necessary units in advance.

2. LITERATURE REVIEW

2.1. INTRODUCTION

Forecasting demand is fundamental within supply chain management, particularly for maintaining healthy inventory levels and providing key insights to numerous functional areas. This process can be done on several levels of granularity (e.g., SKU, size), and the objective of the forecast can be in units (product demand) or monetary units (sales demand). Inaccurate predictions can lead to overstocking or understocking. The first results in product spoilage, storage space constraints, and an unnecessary planning effort, in addition to being unsustainable, while the latter triggers out-of-stock situations, which results in product unavailability and reduced customer satisfaction (Falatouri et al., 2022). Thus, it becomes crucial to deliver accurate forecasts to ensure efficiency on an operational and strategic level (Lalou et al., 2020).

Traditional forecasting machine learning methods found in the literature can be divided into three categories: (1) time series-based methods, (2) regression-based approaches, and (3) supervised and unsupervised methods (Falatouri et al., 2022). Furthermore, it's also important to acknowledge and understand the factors that impact retail sales, whether in an online or offline scenario, which will be discussed further on.

2.2. FORECASTING METHODS IN RETAIL

2.2.1. Time Series-Based Models

Traditionally, time series-based methods are the most used techniques for demand forecasting, since they succeed in dealing with trends and seasonality, common traits of retail data (Marques, 2020 & Falatouri et al., 2022). One of the most famous is the ARIMA method, popularized by Box & Jenkins, with several studies in retail. Gilbert (2005) developed an ARIMA forecasting model and used it to analyze the causes of the bullwhip effect and Shukla and Jharkharia (2013) used ARIMA to predict daily sales of onions and potatoes in an Indian market. Another popular method is the Holt Winter Exponential Smoothing (HWES). Da Veiga et al. (2014), compared this technique to ARIMA when predicting sales of perishable dairy products and discovered that the HWES performed better, capturing the linearity of the series.

Furthermore, it is important to acknowledge that in numerous cases the dependent variable is influenced by external factors, such as weather conditions, holidays, etc., which are not being accounted for in the previously mentioned univariate methods. Thus, new methods were introduced to incorporate external variables, such as ARIMAX (ARIMA with external variables), SARIMAX (Seasonal ARIMA with external variables), etc. Dellino et al. (2015) studied the ARIMA, ARIMAX, and transferred function models to predict sales of fresh

products, including variables like price but highlighted that the effectiveness of each method is closely linked to the dataset's quality. Lee and Hamzah (2010) on the other hand, compared the ARIMAX with calendar effect during Ramadan with SARIMA (Seasonal ARIMA) and Neural Networks, to forecast demand for boy's clothes in Indonesia, and concluded that ARIMAX presented better results in out-of-sample data. Bratina and Faganel (2008) applied ARMAX (ARMA with external variables) to predict beer sales in Slovenia on a daily level, incorporating temperature, holidays, and price-related external features to obtain a more accurate forecast. Furthermore, Arunraj et al. (2016) analyzed the SARIMAX model to predict the demand for bananas in Germany, considering the effects of price reductions, holidays, and months, and concluded that the proposed SARIMAX method outpaces the SARIMA and the benchmark model (a Seasonal Random Walk model) in out-of-sample data.

2.2.2. Regression-Based Approaches

Regression-based methods hold the advantage of being able to incorporate both dependent and independent variables (Falatouri et al., 2022). One of the most popular methods overall is Linear Regression (LR) or Multiple Linear Regression (MLR) (Ulrich et al., 2021). This technique is also commonly utilized for demand forecasting in retail, since it is simple and fast to fit, usually serving as a benchmark model (Evers et al., 2018). However, this method is subject to several assumptions such as homoscedasticity and linearity of the predictor, hence, if not implemented properly it can lead to misleading results (Ulrich et al., 2021). Quantile regression, however, offers an alternative approach by focusing on forecasting specific quantiles. Taylor (2007) developed an Exponentially Weighted Quantile Regression to forecast daily demand intervals for a supermarket, based on high-volatile and skewed series.

Despite not being wildly studied in the recent literature as neural networks, tree-based methods have shown impressive results in retail demand forecasting. Several Kaggle competitions and the M5 competition, which had the goal of forecasting revenue for over 3,000 items sold by Walmart in the USA, brought a new light to these methods, where they dominated the leaderboard, especially gradient-boosted trees methods such as the LightGBM. Januschowski et al. (2022) studied the M5 competition and highlighted that the winning solutions, besides being tree-based methods, were implemented as black boxes (no alteration to the method). This constitutes a glaring difference from the M4 competition, where the best deep learning solution was heavily modified. Furthermore, these methods allow the user to compute feature importance, effectively handle sparse data and missing values without the need for scaling, as it does not influence their learning process, unlike deep learning methods. Additionally, tree-based methods are easier to understand and interpret and provide fast training, unlike neural networks (NN) which can be computationally exhaustive. Raizada and Saini (2021) intended to forecast sales for Walmart's 45 outlets spread across the US, developing, and comparing several models such as Linear Regression (LR), Random Forest (RF), Support Vector Machine (SVM), K-Nearest Neighbours (K-NN), and Extra Tree Regression (ETR). The study achieved the highest accuracy with ETR and RF, incorporating external

variables such as consumer price index, temperature, and unemployment rate. Moreover, Evers et al. (2018) aimed to forecast bread sales per customer for an online supermarket and achieved the highest accuracy with tree models, Decision Tree (DT) and RF, when compared to its baseline model and LR, with the DT and RF capturing the non-linear component of the data. On the other hand, Saha et al. (2022) studied the performance of a Long-short Term Memory (LSTM) and LightGBM to forecast demand for an American retail company, using price, sales, and date features and inferred that the LightGBM outperformed the deep learning approach.

2.2.3. Supervised and Unsupervised Methods

This third group encompasses methods like LSTM and NN, known for their ability to capture the non-linear relationships in the data (Falatouri et al., 2022). Alon et al. (2001) evaluated an NN against ARIMA, HWES, and multivariate regression in a retail setting. The HWES performed well under stable economic conditions but overall, the NN outperformed the traditional methods, effectively capturing the non-linear characteristics of the data and its seasonality. Marques (2020) focused on developing an LSTM approach and compared it to SARIMA, HWES, and Fourier Seasonal models to forecast the demand for fresh fish. In this study, the LSTM demonstrated higher accuracy; nonetheless, the author emphasizes the significant effort required in the data preprocessing phase compared to other methods. Falatouri et al. (2022) also developed an LSTM model to forecast demand for an Austrian retailer, comparing it to SARIMA and SARIMAX. The three methods achieved good results under different settings: SARIMA performed better for products with seasonality and SARIMAX outperformed when the products were also under the effect of promotions, while LSTM yielded great results for products characterized by stable demand.

Inspired by the differences and advantages of ARIMA and NN, Aburto and Weber (2003) proposed a hybrid model using the two techniques to forecast sales for the top 50 selling products in a Chilean supermarket, with the purpose of developing a replenishment system. Using date, price, and calendar features, the authors compared the two models and their combination, concluding that the hybrid approach achieved the lowest percentage errors, followed by the NN. He and Yu (2020) also developed a hybrid model, combining a LightGBM and LSTM to predict vegetable sales, using its sales and characteristics as features. The two models performed similarly well but their combination lowered the MAPE (Mean Absolute Percentage Error) across all products. In this hybrid approach, the LightGBM was utilized to deal with the high volumes of data while the LSTM was employed to discern the non-linearity components of the data. Similarly, Punia et al. (2020) combined RF with LSTM to forecast demand in a multi-channel retail setting. The hybrid approach was evaluated against LR, ARIMAX, NN, and the two initial models, in terms of variance, bias, and accuracy, incorporating product details, transactional details, and store details (in case of offline sales) as features. The evaluation was done on different window forecasts (one day, one week, and one month),

on an offline and online setting, with the LSTM-RF approach achieving better results across all scenarios.

2.3. Factors Influencing Demand

As studied by Fildes et al. (2022) numerous factors can influence product demand. Some fall within the control of retailers (such as promotions and secondary effects like cannibalization) while others are uncontrollable, but retailers know when they are taking place (holidays, seasonal variations, etc.). Additionally, sales can also be influenced by weather patterns, economic conditions, etc.

2.3.1. Calendar Events

Calendar events exert a considerable impact on retail sales with numerous researchers including it as a dummy variable (Lee and Hamzah, 2010). Hirche et al. (2021) analysed the sales of alcoholic beverages and concluded that public holidays such as Christmas, Easter, and New Year lead to significantly higher sales across all beverages. Holidays such as Christmas and New Years can also be characterized as seasonality and modeled as such while holidays such as Easter cannot, since the date changes every year (Marques, 2020).

2.3.2. Weather

Product demand is also greatly impacted by the weather, especially by sunlight. Lazo et al. (2011) conducted a research in the US on the economic impact of weather conditions, finding that consumers are more prone to purchase and increase their spending when exposed to sunlight. Incorporating weather as a feature can be challenging to its multifaced nature (temperature, rain, wind, etc). However, in several cases it has demonstrated efficacy: Hirche et al. (2021) also included temperature variables and discovered that certain beverages such as beer, red and white wine exhibit increased sales when the temperature is high.

2.3.3. Seasonality

Retail sales data displays a seasonality component since it deals with multiple seasonal cycles. Usually, sales increase during the weekends and decrease during the weekdays, with a peak in summer and across the holidays such as Christmas. Hence, it is crucial to choose the right forecasting technique that can support seasonality (Fildes et al., 2022).

2.3.4. Pricing Strategies and Promotions

Including pricing variables (regular price & discounted price) is extremely important in forecasting product demand especially on a short-term window, since promotions, no matter how temporary, impact sales across both online and offline channels. Additionally, promotions can lead to cannibalization, where a reduction in the price of one product affects the sales of other items. Yet, modeling cannibalization is highly complex, requiring the identification of its drivers and affected products (Fildes et al., 2022).

2.3.5. Other factors

Numerous other factors can impact retail sales, both positively and negatively. For instance, Out-of-stock situations lead to product unavailability and potential loss of sales. This leads to inaccurate demand forecasts since the models depend on the data captured by POS (Point of Sale) transactions and these scenarios might not reflect the actual customer demand (Fildes et al., 2022). Moreover, in this technology era, it's also important to account for social media and online reviews. Through social media platforms, retailers can engage with customers, build brand awareness, and showcase products. Social media influencers, user-generated content and online reviews can also influence purchasing decisions, driving traffic to their websites and/or stores. Archak et al. (2011) and Chen et al. (2011) identified a direct relationship between online reviews and product sales, highlighting however, that the influence is dependent on the category. Hence, including online reviews in the forecasting models, by employing techniques such as text mining or sentiment analysis could improve the model's performance and results.

2.4. SUMMARY

As observed, an accurate demand forecast benefits retailers on an operational, strategic, and tactical level, moving the business forward. As observed in the literature, demand is not the only key factor to be modeled, as product characteristics, price and markdown information, calendar events, temperature, etc., have proven to improve the accuracy in multiple studies. Moreover, following extensive research on the literature concerning retail forecasting, it becomes evident that there is no universally optimal method, due to its complexity. Each method reviewed possesses its own set of strengths and weaknesses, and the choice of method should ultimately be guided by the specific characteristics of the available data, the model, and the purpose of the problem at hand.

3. METHODOLOGY

The present study followed the CRISP-DM Methodology (Cross Industry Standard Process for Data Mining), providing a structured and efficient approach throughout its six phases: Business Understanding, Data Understanding, Data Preparation, Modeling, Evaluation, and Deployment. This method allows a continuous evaluation of the progress of the project, keeping business goals in focus.

Figure 1 - CRISP-DM Framework

A brief overview of the CRISP-DM framework:

- Business Understanding: In the first phase, the main goal is to comprehend the project's objectives and business requirements. This understanding is then translated into a concise data mining problem definition and an initial plan devised to meet the end goal.
- Data Understanding: This phase starts with data collection and its analysis, encompassing tasks such as identifying data quality issues and deriving initial insights.
- Data Preparation: The data preparation phase involves a series of tasks aimed at transforming the initial raw data into the finalized dataset, ready to be used in the modeling tools.
- Modeling: As the name implies, this is the stage where the models are chosen and implemented, fine-tuning their parameters. Additionally, as certain algorithms may

have specific data format requirements, it might be necessary to return to the previous phase to ensure compatibility.

- Evaluation: After selecting and implementing the models, a comprehensive evaluation is conducted, also focused on verifying their alignment with the business goals.
- Deployment: Typically, the development of the model does not mark the conclusion of the project. Even if the model's objective is to improve the understanding of the data, the insights acquired must be structured and communicated in a manner that is accessible and actionable for the end-user (Introduction to CRISP-DM, n.d.).

Moreover, the success of the project heavily depended on the utilization of two programming languages, SQL (Structure Query Language) and Python. SQL was used to retrieve, manipulate, and join the data from Snowflake, the designated data storage platform of the company. Meanwhile, Python was employed for data exploration, preparation, model development, and subsequent evaluation, by leveraging various libraries available, which will be explained further on. Additionally, the project was implemented in a Jupyter Notebook, a web-based environment that allows users to write and execute Python code (Jupyter, n.d.).

3.1. BUSINESS UNDERSTANDING

This study was conducted during a 12-month internship in a Multinational Sportswear Retail company. Initially, the business operated solely through physical stores. However, with the changing landscape characterized by the rising popularity of online shopping, the necessity of establishing an online platform became evident. Especially after the COVID-19 pandemic, the digital channel experienced a boom while Brick & Mortar sales slowed down, bringing new challenges to the business.

Despite the rapid growth of e-commerce, the company continued to rely heavily on manual processes for this side of the business, particularly for demand forecasting. As stated previously, accurate demand forecasts enable informed decisions across several operational areas, including inventory management, which plays a vital role in the organization.

This project focuses on one such process within the digital channel: the management of AA safety stock bookings. These products, which as the name indicates are always available, represent a high share of the business with the sales continuously growing throughout the years. Thus, it’s important to ensure that customer demand is met at all times. As outlined in the introduction, a dedicated safety stock is specifically allocated to these products for all retail channels where they are offered, with priority given to the first entity that books the units. Hence, it’s crucial to predict future demand as accurately and efficiently as possible to ensure the necessary units are reserved far in advance and consequently avoid stock-outs.

Previously, the demand forecasting process for these products in the digital channel relied solely on the intuition and expertise of the planners, based on their experience and historical

sales data. However, given the increasing online demand witnessed in recent years, the high volumes of data available (since the forecast is run weekly), and the additional factors that can influence demand, as discussed in the literature review, there was an evident need to improve and automate this process, implementing a machine learning solution. This solution aims to forecast daily product demand for the upcoming 28 days, aggregating it on a weekly level. Afterward, considering the estimated inventory levels, it will assess the need to book additional units and the quantity, if needed.

3.2. DATA UNDERSTANDING

3.2.1. Data Collection

Data collection was one of the most challenging steps in the project, starting with understanding what data was needed, its availability, and the structure of its data sources to combine it into the final dataset.

As discussed in the literature review, numerous factors beyond historical demand can influence demand. Hence, information such as markdown values, product characteristics, inventory levels, website visits and conversion, calendar features, and holiday promotions were deemed significant and consequently incorporated into the analysis. As expected, the majority of these variables came from different tables in Snowflake, which had different levels of granularity. Therefore, it was important to understand how they could be joined.

Product characteristics such as gender (men, women, or kids), segment (lifestyle or performance), and division (footwear, apparel, or equipment), among others, were extracted from the *attribution* table, which was organized at a SKU level. The AA variable (column indicating whether a product was AA or not) was also found in *attribution* and before joining to the other tables, all the non-AA products were filtered out.

Considering calendar features, both the Gregorian calendar and the company's retail calendar were taken into account, as they operate on different timeframes. These variables, including day, week, retail week, etc., were available in the *calendar* table, structured at a daily level.

The *traffic*, *inventory*, *promotion,* and *price* tables displayed the same level of granularity, on a daily and SKU basis. Website visits and conversion data were sourced from *traffic*, the recommended retail price and current retail price were retrieved from *price,* stock levels were pulled from *inventory* and *t*he promotion indicator was queried from the *promotion* table. The latter indicates, for each day and product, if there was a promotional campaign on the website or not. These promotions can either be directly incorporated in the product as a discount or be in the form of a discount code to input before the payment.

Lastly, historical demand (units and monetary units) and markdown information (discount amount) were pulled from the *sales* table. This table was organized at the highest level of

granularity of all, per country, day, SKU, and size. Although sizing and country information would have been helpful, it was not available in the remaining tables. Consequently, to be joined with *attribution*, *inventory*, *calendar*, *promotion*, *traffic,* and *price*, *sales* had to be aggregated to a daily and SKU level. Following that, left joins were utilized to merge *sales* with *promotion*, *traffic, inventory,* and *price*, based on the product code and date. Afterwards, another left join was implemented with the *calendar* table, on the date only. Finally, a right join was performed with *attribution* on product code, to only retain the AA products.

Table 1 - Final Dataset

Variable	Description	Data Source
transaction_dt	Date of the transaction. Format: dd-mm-yyyy.	sales
product_cd	SKU, the unique identifier of each product. Format: xxxxxxx-xxx. The first 7 characters identify the product while the last 3 identify its main color.	sales
demand	Total sales in dollars.	sales
units_sold	Total sales in units.	sales
discount_amount	Reduction in the selling price of a product from its original price. Displayed in dollars.	sales
stock	Current inventory available in the company's warehouse.	inventory
item_name	Product name displayed on the website.	attribution
color_cd	3-digit code, an indicator of the product's main color.	attribution
gender	The target consumer of the product, Men, Women or Kids.	attribution
division	Characteristic of the product based on its primary function. Division is split between Footwear, Apparel and Equipment.	attribution
category	Classification of the product based on its intended use, such as Basketball, Running, etc.	attribution
silhouette	Overall shape or style of the product, i.e., pants, t-shirts, hats, etc.	attribution
segment	Another characteristic of the product, split between lifestyle and performance.	attribution
family	Product classification based on its similarities. There are over 60 families in the dataset.	attribution

model	Sub-section of family. The dataset encompasses around 175 models.	attribution
rrp	Recommended retail price, the first price selling price of the product on the website.	price
crp	Current retail price, the selling price of the product on the website at the moment it was sold.	price
visits	Number of interactions that the users have with the product on the website.	traffic
conversion	Percentage of website visits that completed a purchase. $Conversion = \frac{Units\ Sold}{Website\ Visits}$	traffic
day	Day number.	calendar
day_of_week	Day of the week, such as Monday.	calendar
Week	Week number.	calendar
month	Month number.	calendar
year	Year number.	calendar
retail_week	Retail week number.	calendar
retail _season	Retail season number.	calendar
retail_year	Retail year number.	calendar
promo	Identifier of promotions. If the product was included in a promotion on the website when it was sold (such as Cyberweek, End of Season Sales, Valentine's Day, etc.) it will show 1, otherwise 0.	promotion

3.2.2. Data Analysis and Visualization

The dataset encompassed 2 years of data, from 23-08-2021 to 26-08-2023, with over 400,000 transactions, covering 1226 different AA products. The data included 26 independent variables and the target variable, units_sold. After an initial analysis, no duplicates were identified in the data. However, 8 columns displayed missing values: category, family, model, rrp (recommended retail price), crp (current retail price), stock, visits, and conversion. In some cases, the missing values could be attributed to occasional failures in the company's technological infrastructure, resulting in incomplete data input. Furthermore, it's worth noting that certain data sources may not record entries when the value is zero. Given that each table has a different owner and management protocols, both assumptions regarding missing values are considered valid. Additionally, it's important to highlight that the features

displayed a high positive skewness. This means that the majority of the data points are concentrated on the left side of the distribution, with a long tail extending to the right, indicating the presence of unusual higher values in the dataset, outliers. This was discovered by analyzing the descriptive statistics available in pandas - for all variables the mean was greater than the median, indicating that the data was skewed to the right. Furthermore, the difference from the 75% percentile to the maximum value was significant for most features, indicating the presence of outliers. Both these assumptions were then confirmed with the help of histograms and boxplots.

Figure 2 illustrates the evolution of the target variable over the 2 years, the demand in units, with noticeable peaks occurring in November and a clear decrease after it. These spikes in sales can be attributed to CyberWeek, a highly promotional week at the end of November that includes Black Friday and Cyber Monday, where retailers offer exceptional discounts and deals. This assumption is confirmed in Figure 3, where it is noticeable that the sales peaks in November were driven by promotional campaigns, both in 2021 and 2022, validating the importance of considering promotions when forecasting demand in a retail setting.

Furthermore, it is important to mention the peaks observed in May of both 2021 and 2022 and in September 2021. These peaks align with promotional activities as seen in Figure 3, specifically special spring campaigns in May and the Back to School promo in September, concluding that the data exhibits seasonal patterns. Moreover, it was also discovered that Mondays are the best-selling days for AA products, followed by Sundays.

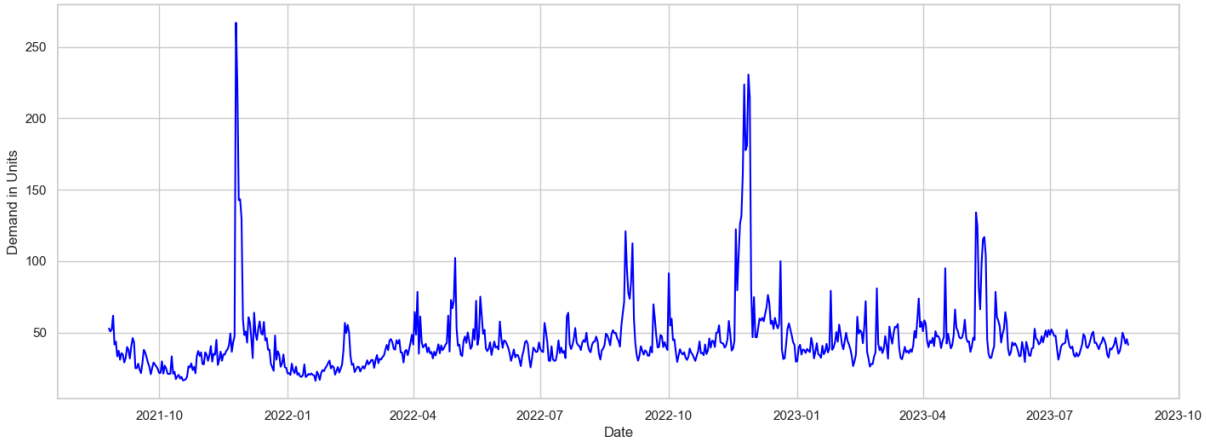

Figure 2 - Evolution of Sold Units

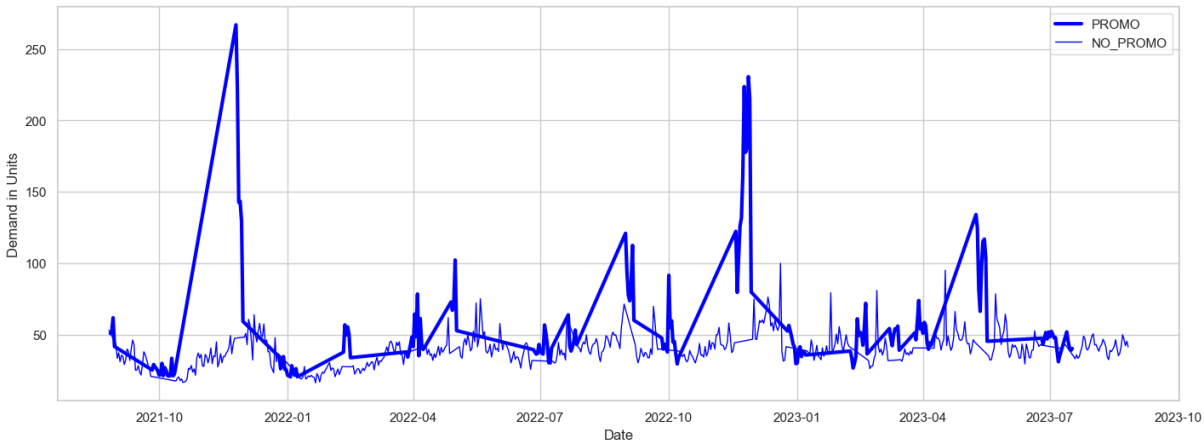

Figure 3 - Evolution of Sold Units, split by the Promotional Indicator

Additionally, as seen in Figure 4, the company has 3 major selling categories: Men's, Footwear, and Lifestyle products. Within the genders, Men's products drive the sales, accounting for over 50% of the company's share of business. Regarding division, footwear emerges as the preferred choice among consumers, particularly for everyday wear rather than for a sport-specific activity. It is worth mentioning that these product classifications are based on what the product was designed for, and not what the consumer uses it for.

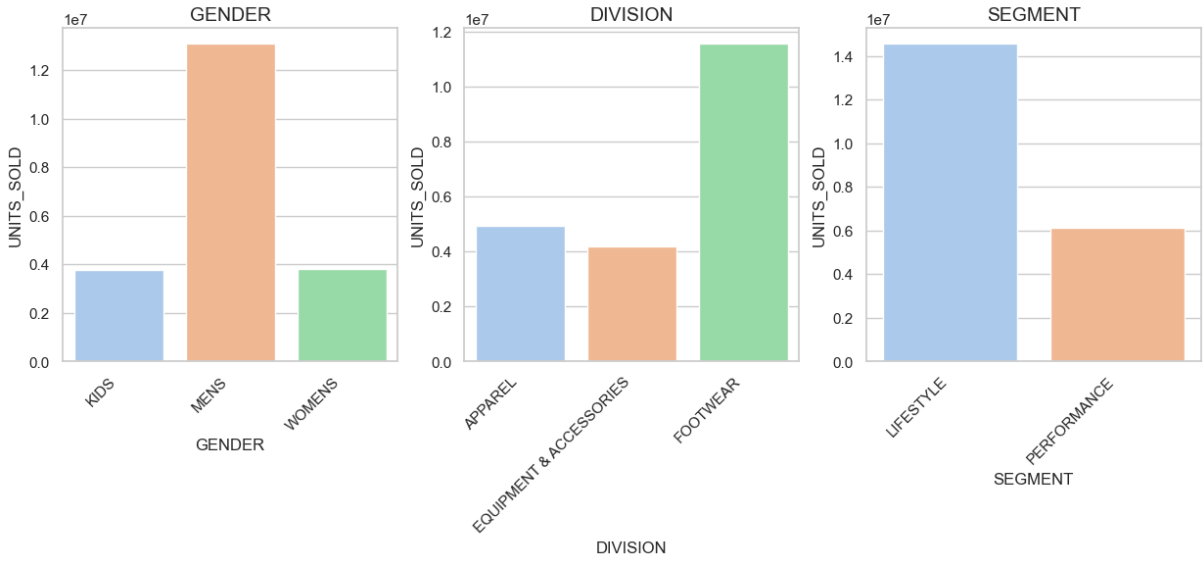

Figure 4 - Quantity of Sold Units by Gender, Division and Segment

3.3. DATA PREPARATION

Data preparation is a crucial step in the methodology since it ensures that the data used for modeling is clean, relevant, and properly formatted.

3.3.1. Data Cleaning

This phase started by addressing the missing values, using the pandas and numpy libraries. As mentioned before, 8 columns displayed missing entries: category, family, model, rrp (recommended retail price), crp (current retail price), stock, visits, and conversion. Due to the uniqueness of the data, the columns were dealt with differently.

In the dataset, the product code follows a 10-character format (xxxxxxx-xxx), where the first 7 digits identify the product, and the last 3 digits denote the color. To facilitate this step, a new column named "product" was created containing only the first 7 digits of the product code. This aided in filling missing attributes in the category, family, and model columns by replacing them with the mode of each product, with any remaining missing values marked as NA.

The rrp (recommended retail price) remains constant per material code regardless of the transaction date since it is the first selling price of the product. Moreover, it is important to consider the potential price variations across different colors of the same product. Given this, missing values were inputted using the mean of each product code, which filled all the null values. As for the crp (current selling price), missing entries were replaced with the result of dividing the dollar sales by the units sold, representing the selling price of each observation.

Regarding stock, conversion, and visits, it was discovered that the tables *inventory* and *traffic* do not record entries when the values are zero, so all the missing values were replaced with zero.

Furthermore, outliers were not excluded from the dataset since they provide important information to the research. For instance, during high promotional periods such as CyberWeek, there is a substantial increase in demand, website traffic, and conversion rates, with inventory levels being exceptionally high. Since these data points provide valuable insights into the behavior of the business during abnormal periods it's important to include them, ensuring that the model captures all the possibilities including fluctuations and anomalies.

3.3.2. Data Transformation

This phase started by creating additional variables. For time-dependent data, there are four classes of features that can be created to improve model accuracy: time-based features, lag features, window features, and their combinations.

Time-based features such as day, month, etc., were mostly all available in Snowflake, within the *calendar* table, so there wasn't a need to create additional features.

Lag and window features play an important role in feature engineering, capturing temporal dependencies and patterns, since the behavior of the past values may repeat in the future. (Ribeiro et al., 2011). Lagged features are created by incorporating the value of a feature from a previous time point into the model at the current time point, shifting the data by a specific number of time steps known as the lag, using the *shift* command available in pandas. Rolling window features, on the other hand, compute summary statistics, in this case, the mean, across a sliding window of a specific number of previous values. This was calculated using the *rolling* function and then aggregated using the *mean* function. Given the weekly nature of the safety stock process, these features were calculated on intervals of 7, 14, 21, and 28 days.

Additional features were created to capture a wider range of patterns and relationships within the data. These features include averages for daily and monthly sales (*average units per day*, *average units per month*) as well as product-related metrics (*average stock per product*, *average visits per product*, *average unit per product,* and *average of units per product and month*), using the *transform* function available in pandas.

The final task in this phase involved encoding the categorical variables. However, due to the extensive number of categories, implementing one-hot encoding was unfeasible. For instance, the "product-code" column alone featured 1226 distinct items distributed across 65 families, 175 models, and other product characteristics, resulting in an excessively large dataset. As an alternative, the categorical features were encoded using the LabelEncoder technique from Scikit-learn library. This approach assigns numerical labels to categories ranging from 0 to n-1, where n represents the number of categories. While this method avoids increasing dimensionality, it can potentially cause algorithms to infer an ordinal relationship among the categories.

3.3.3. Feature Selection

As it will be explained in more detail in the next phase, the models selected for this project were tree-based. These algorithms support feature importance, allowing users to identify and select the most relevant variables for each use case. Therefore, the focus of this section was primarily on identifying and removing highly correlated features to avoid redundancy and multicollinearity, consequently improving computational efficiency and model interpretability and performance.

Given the non-linear relationships of the data, the Spearman correlation was used to measure the monotonicity of the relationships between the features. As a result, features with correlation coefficients over 0.90 were removed. This led to the removal of *crp* (current retail price), *demand*, all the rolling window features, and *average visits per product*. In the end, the final dataset encompassed 34 independent variables and the target, *units_sold*.

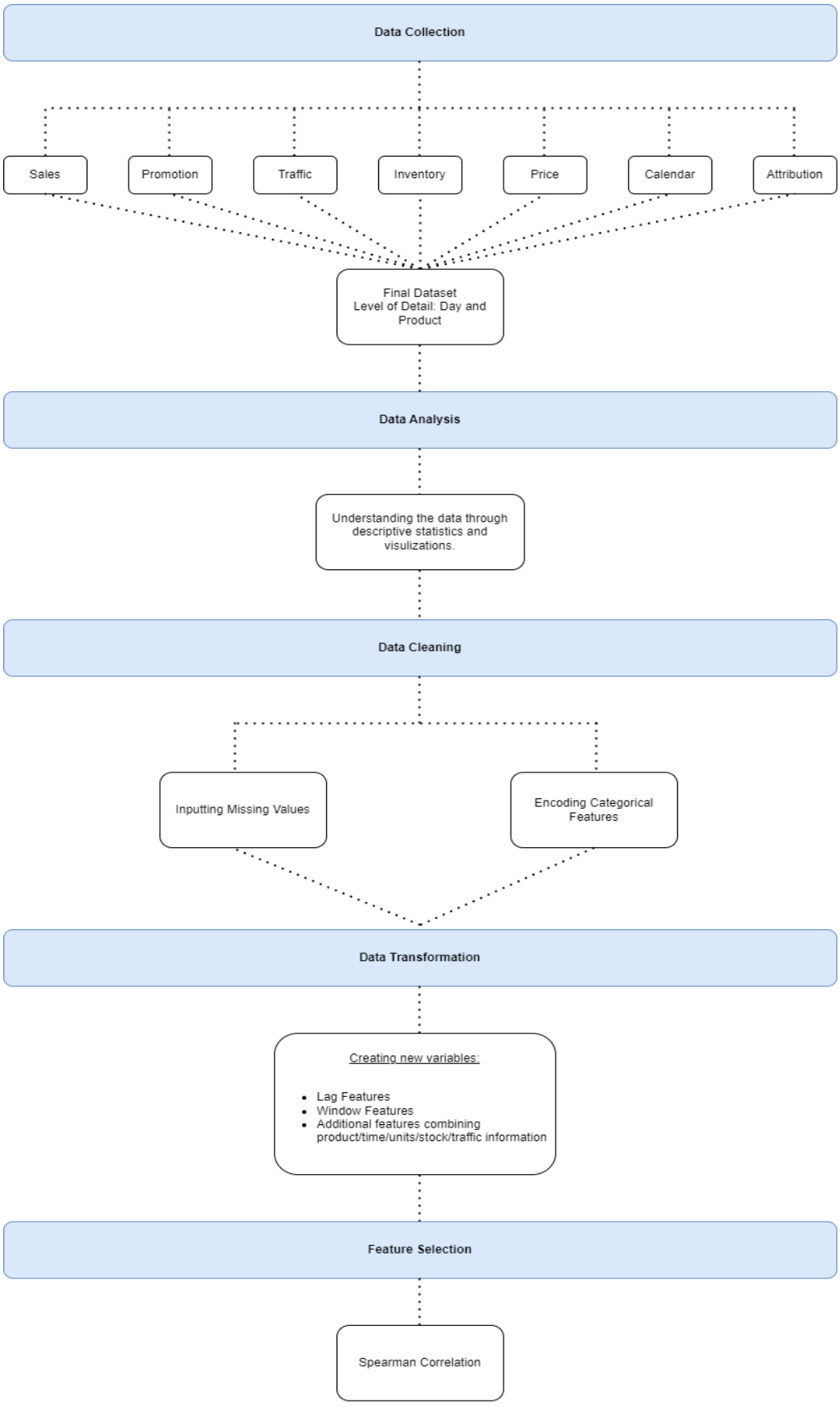

Figure 5 - Workflow of the Data Collection, Analysis, Preparation, Cleaning, Transformation and Feature Selection

3.4. MODELING

This phase focused on implementing and fine-tuning the machine learning algorithms to improve their performance.

Statistical methods such as ARIMA, SARIMA, etc., have demonstrated effectiveness in handling trends and seasonality in demand data. However, due to the large size of the dataset, containing 1226 different products, implementing these methods would have been impractical. Most techniques don't support multiple time series, so each product would require its own model, leading to a time-consuming and exhaustive process. Thus, a more efficient method was needed, capable of handling large datasets and multiple products simultaneously.

This project focused on implementing three tree-based regression models—Random Forest (RF), LightGBM, and XGBoost. The decision to opt for tree-based methods was influenced by several factors related to the data: the extensiveness of the dataset, non-linear relationships, strong positive skewness, and the presence of outliers made tree-based methods an attractive choice. These methods are characterized as non-parametric and non-linear, i.e., they don't make assumptions about the distribution of the data and are more robust to outliers. Furthermore, they can handle large datasets without the need to perform data normalization. Additionally, they also support feature importance calculation, which allows the executioner to identify and select the most relevant features, contributing to more accurate results and increased efficiency (Januschowski et al., 2022).

Among tree-based models, LightGBM was chosen for its success in the M5 competition, leading the scoreboard (Januschowski et al., 2022). Random Forest was selected based on its proven success in the studied literature (Evers et al., 2018 & Raizada and Saini, 2021), while XGBoost was chosen for its successful impact across various machine learning and data mining problems (Chen and Guestrin, 2016).

Tree-based models use decision trees as their foundation. Decision trees work by recursively partitioning the feature space into smaller regions and predicting the target variable within each region (Loh, 2011). They split the data based on feature values, and at each split, aim to maximize the homogeneity of the target variable, which makes it suitable for skewed data. Afterward, predictions are made by averaging (regression) or taking the majority class of the target variable values (classification) within each leaf node.

Furthermore, these three methods are characterized as ensembles. This technique combines the outputs of several machine learning models to achieve better results than any single algorithm could achieve alone. In essence, an ensemble model is a collection of numerous models – in this case, decision trees.

3.4.1. Random Forest

Random Forest is a versatile machine learning technique capable of handling both regression and classification tasks. As an ensemble learning method, it combines multiple weak models to create a stronger predictive model (Breiman, 2001). Instead of relying on a single decision tree, Random Forest constructs multiple trees.

For each decision tree in the forest, the algorithm begins by creating a bootstrap sample from the training dataset. This involves randomly sampling the data with replacement. At each tree node, a random subset of features is selected to determine the best split. The algorithm recursively splits the nodes until a stopping criterion is met. The predictions from all the individual trees are then aggregated to produce the final output: the mode of the tree predictions for classification tasks or the mean for regression tasks (Random Forest, 2022). By implementing multiple decision trees, Random Forest reduces the variance and mitigates overfitting compared to a single decision tree.

This algorithm was implemented using the Random Forest Regressor, imported from the Sckit-learn library in Python. An essential hyperparameter within this implementation is n_estimators, which defines the total number of decision trees. Increasing the number of trees may improve model performance, but it also raises computational demands. The max_features parameter controls the number of features considered for the best split. Increasing this parameter may enhance the performance of the model but also risks overfitting. As for the max_depth parameter, it limits the depth of the decision trees. Greater depth allows the model to capture more complex patterns in the data, enhancing performance but also increases the potential for overfitting. Furthermore, the min_samples_leaf parameter sets the minimum number of samples needed to form a leaf node. Higher values typically produce a more conservative model, reducing the likelihood of overfitting, but may cause underfitting if the value is excessively high. Finally, min_samples_split indicates the minimum number of samples needed to perform a split at a node. Setting a higher value generally results in a more robust model by preventing splits based on small data subsets, though it may lead to underfitting if set too high (Random Forest, 2022).

3.4.2. XGBoost

XGBoost or eXtreme Gradient Boosting, is a successful machine learning algorithm, known for its effectiveness in both regression and classification tasks.

The model is a part of the ensemble learning category and operates by combining the predictions of multiple weak learners, decision trees, to create a robust predictive model. XGBoost utilizes a gradient boosting framework, which sequentially adds decision trees to the ensemble, with each new tree aiming to improve upon the errors of its predecessors. This iterative process continues until a stopping criterion is met (XGBoost, 2023). Furthermore, XGBoost offers extensive support for handling various real-world data characteristics,

including handling missing values and supporting categorical features (Januschowski et al., 2022). Lastly, it incorporates regularization techniques to help control the complexity of the model and prevent overfitting.

The XGBRegressor was implemented by levering the xgboost library, making use of its various hyperparameters for optimization. The n_estimators parameter determines the number of trees to be constructed. While a higher value might improve model performance, it increases the computational time and model complexity. The learning_rate regulates the step size for each iteration. A smaller value results in a slower process but may yield more resilient models with reduced risk of overfitting. On the other hand, a higher value accelerates training but may lead to overfitting. Regarding the max_depth parameter, it indicates the maximum depth for each tree. Deeper trees possess the ability to capture more complex relationships in the data, but they are also at a higher risk of overfitting. The subsample parameter represents the fraction of observations randomly sampled for each tree during training. A lower subsample outputs a more conservative model, reducing the risk of overfitting. The min_child_weight parameter sets the minimum sum of weights of all observations required in a child node during tree construction. Setting higher values for this parameter can help prevent overfitting, but excessively high values may risk underfitting. Lastly, col_sample_bytree determines the ratio of columns randomly chosen for each tree construction. Lower values promote diversity by requiring trees to learn from different subsets of features, thus mitigating the risk of overfitting (XGBoost, 2023).

3.4.3. LightGBM

LightGBM or Light Gradient Boosting Machine, is a high-performance ensemble technique proposed by Microsoft.

The algorithm constructs decision trees sequentially to minimize residual errors, adapting a leaf-wise tree growth strategy. Furthermore, it leverages histogram-based algorithms to construct the decision trees, which discretize continuous features into bins. This method allows the algorithm to efficiently identify optimal split points, significantly speeding up the computation process. Moreover, LightGBM possesses other characteristics that improve the computational process – Exclusive Feature Bundling (EFB) and Gradient-Based One Side Sampling (GOSS). In other words, the algorithm combines similar features together to reduce the complexity of the model (EFB) and focuses on the most important data points during training by prioritizing instances with larger changes (gradients) in prediction error (GOSS). This helps it learn faster and more efficiently (LightGBM, 2024). Moreover, similar to XGBoost, LightGBM includes regularization techniques to manage the model's complexity and mitigate overfitting.

This algorithm was implemented using the LightGBM regressor, available in the lightgbm library. The hyperparameters were a key factor in improving the performance of the model, starting with the n_estimators, indicating the number of decision trees to be constructed

during training. Although a higher value can enhance the model's performance, it also increases computational time and complexity. The learning_rate parameter sets the step size for each iteration. Smaller values lead to slower processes but can produce more robust models. Regarding the num_leaves parameter, it determines the maximum number of leaves per tree. A higher value enables the model to capture more complicated patterns in the data but also increases the risk of overfitting. The min_data_in_leaf parameter defines the minimum number of data points needed to create a leaf node. Setting a higher value reduces the risk of overfitting, but if the value is too high it might lead to underfitting. Lastly, the subsample parameter indicates the ratio of observations to be randomly selected for each tree while colsample_by_tree sets the fraction of columns subsampled for each tree. For both parameters, lower values generally lead to more conservative models, reducing the risk of overfitting (LightGBM, 2024).

As mentioned previously, these methods support feature importance calculation, so this process was conducted for each model. The table below displays the final subset used to train each algorithm.

Table 2 - Final Dataset used for training each model

Model	Dataset
Random Forest	product_cd, color_cd, gender, family, rrp, discount_amount, stock, visits, conversion, retail_week, day, day_of_week, average_product_units, average_product_units_per_month, units_lag_7, units_lag_14, units_lag_21, units_lag_28, promo
XGBoost	product_cd, color_cd, gender, category, family, model, rrp, discount_amount, stock, visits, conversion, retail_week, day, day_of_week, average_product_units, average_product_stock, average_product_units_per_month, units_lag_7, units_lag_14, units_lag_21, units_lag_28, promo
LightGBM	product_cd, color_cd, gender, family, model, rrp, discount_amount, stock, visits, conversion, day, retail_week, average_product_units, avg_daily_units, average_product_units_per_month,

	average_product_stock, units_lag_7, units_lag_14, units_lag_21, units_lag_28, promo

Analysing the table above, one can conclude that stock, price, traffic, and the newly added variables (especially the lagged features) are considered significant. Among the attribution variables, only gender, color and family are selected for the three models, with model being selected for two. As for time-dependent variables, only day, day_of_week, and retail_week are observed to have an influence on the target variable.

3.5. EVALUATION

This phase focuses on evaluating the chosen models with the correct metrics, ensuring alignment with the business goals.

3.5.1. Evaluation Metrics

To evaluate the implemented models, two metrics were chosen: Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE).

The MAE is calculated as the average of the absolute errors (difference between predicted and actual values). It provides a clear and intuitive understanding of the overall performance of models, as it's not influenced by outliers.

$$\mathrm{MAE} = \frac{1}{n} \sum_{i=1}^{n} |yi - pi|$$

Figure 6 - Equation of MAE

The RMSE is calculated as the square root of the average of the squared errors (difference between predicted and actual values). By squaring the errors, this metric gives greater weight to large errors, which is important given the presence of outliers in the data.

$$\mathrm{RMSE} = \sqrt{\frac{1}{n} \sum_{i=1}^{n} (yi - pi)^2}$$

Figure 7 - Equation of RMSE

In the formulas, n indicates the total number of observations, yi refers to the actual value and pi denotes the predictive value.

Essentially, MAE provides information on the general performance and RMSE serves the purpose of prioritizing outliers, assessing how the model performs with extreme values. Moreover, another frequently used accuracy metric in retail scenarios is the Mean Absolute Percentage Error (MAPE), due to its simplicity in interpretation and the ease of comparing forecasts across series with varying scales. However, MAPE becomes undefined when there are zero values in the output, which can happen in this project, so it was not implemented.

3.5.2. Data Split and Hyperparameter Tuning

Before starting the modeling phase, the data was split into a train and test set. Due to the data's temporal dependence, it should not be split randomly to avoid information leakage. Furthermore, aligning with the project's objective of forecasting AA unit quantities for the following 28 days, the split was performed on 29-09-2023, taking the last 28 days as the test set and the remaining observations for training.

After partitioning the data into training and test sets and implementing the models, the next step focused on fine-tuning their hyperparameters, with Optuna proving to be the most efficient tool for this task. Optuna is a hyperparameter optimization library in Python that automates the hyperparameter tuning process for machine learning models (Aggarwal, 2023).

For this project, Optuna was implemented for each model for 50 trials, searching for the hyperparameters that minimize MAE. The table below displays the best set of hyperparameters for each model found with Optuna.

Table 3 - Final Hyperparameters to be used in each model

Model	Final Hyperparameters
Random Forest	max_depth = 10 n_estimators = 300 min_samples_split = 4 min_samples_leaf = 4 max_features = 'sqrt'
XGBoost	n_estimators = 512 learning_rate = 0.04 colsample_bytree = 0.934 max_depth = 7 subsample = 0.78 min_child_weight = 10

LightGBM	n_estimators = 288 learning_rate = 0.005 num_leaves = 376 subsample = 0.86 colsample_bytree = 0.859 min_data_in_leaf = 37

Moreover, Section 4 will provide a thorough explanation of the results.

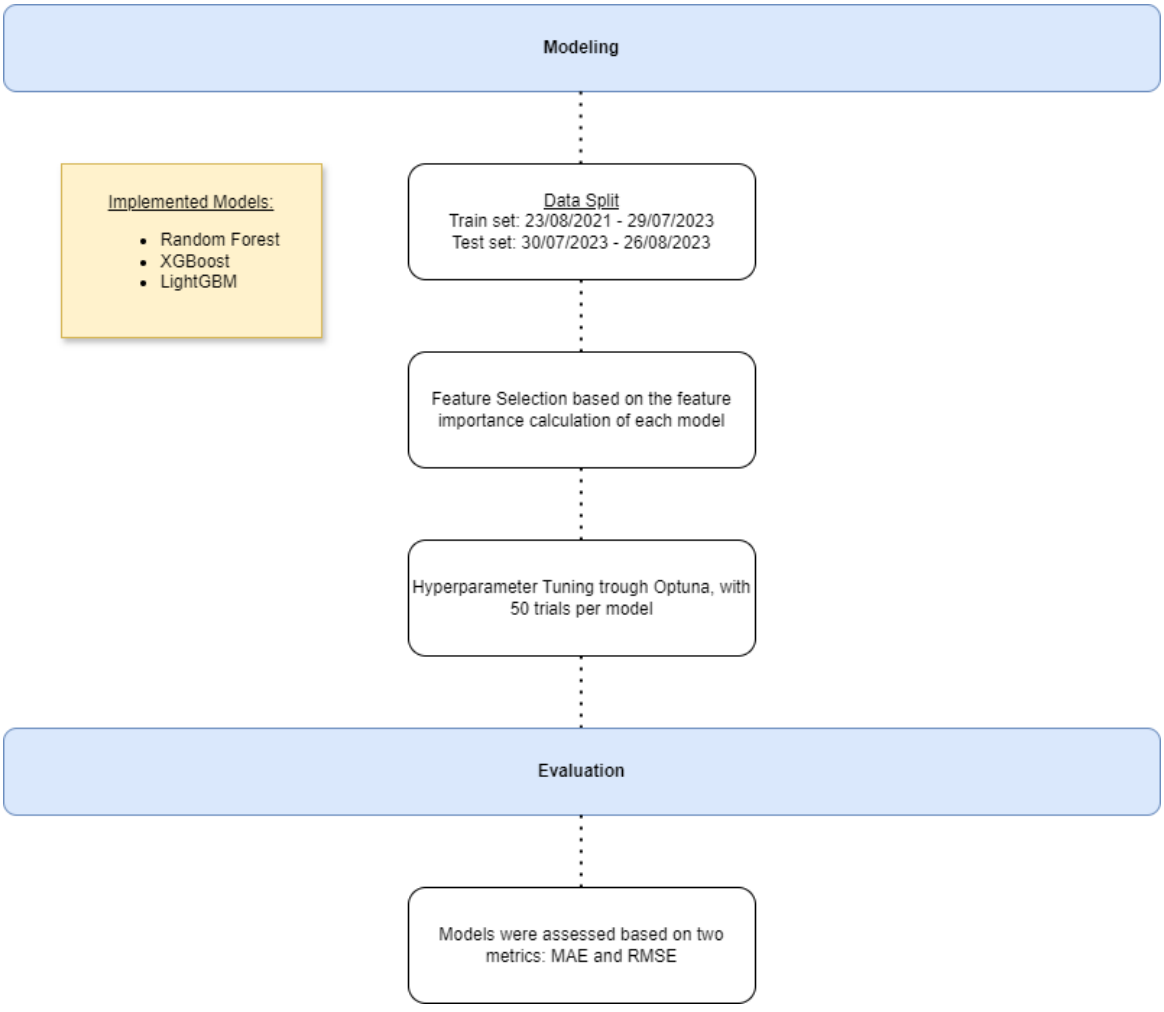

Figure 8 - Workflow of Modeling and Evaluation

3.6. DEPLOYMENT

This project also aims for simplicity and efficiency, so that anyone can run it, independently of their business function. For that purpose, the process will run in one single Jupyter notebook and output a CSV (comma-separated value) file with the expected units to be sold per product and week, along with the indication of the number of units to buy (if necessary).

In the same notebook the model was created, another query was implemented, to generate the new data for predictions. This query begins by defining a temporary table, using the with clause. This temporary table takes the upcoming 28 days after the current date from the *calendar* table (along with the respective day and retail week) and performs a cross-join with the AA product codes, present in *attribution,* also adding the needed product characteristics, to follow the structure of the test set. This ensures that there will be an observation for each day and product combination.

Next, the query extracts the date features, the product code, and product characteristics from the temporary table. Subsequently, a left join is performed with the promotion table to pull the promo indicator and the estimated discount, joined on the date and product code. The promotion table already contains future information (in the short term) as well as the intended discount percentage (which may change). Moreover, the remaining columns (stock, visits, conversion, etc.) were added as null. Lastly, a right join is performed with a new table, *lifecyle_engine*, on the product code, to filter only the active products. This table is updated every day and is structured at a product level, containing the indication if a product is active on the website or not. This reduces the output, decreasing the computational time.

After, an initial check is done, comparing the new dataset with the initial one to understand if there are new material codes. For further understanding, once a product is attributed as AA, it can never be non-AA again, only discontinued, but old products can be converted to AA. This does not occur often, but it can happen.

Then, the two datasets (new data and initial dataset) are merged, and the null columns are populated. For stock, visits, conversion, average_product_units, avg_daily_units, average_product_units_per_month, average_product_stock, units_lag_7, units_lag_14, units_lag_21, units_lag_28, and rrp, an assumption was made to take the values of the same date in the previous year, using the shift function available in pandas. Afterward, the discount_amount feature is computed as the multiplication of the discount percentage column and the rrp, discarding the discount percentage column after.

With the new dataset structured appropriately, the categorical features are encoded, and the data is fed into the model to generate the predictions. If there are new products, the solution found is to take the average of the product's gender, family, and model predictions. Finally, the encoding is reversed.

The resulting values are aggregated per product code and retail week and then compared to the stock levels at the beginning of each retail week (once again, these values are taken from the same date in the previous year), to ensure that there is enough stock for the following week. A threshold of 5 units was set to always ensure stock. Hence, another column was created to output the business result per week and product based on the following calculation:

$$stock - (predicted\ units + 5) = n$$

- If n < 0 then "Action need. Book |n| units from the safety stock."
- If n ≥ 0 then "No action needed."

Table 4 - Example of the Output

Retail_week	Product_cd	Predicted Units	Action
9	aaaaaaa-010	20	No action needed
9	bbbbbbb-020	43	Action needed. Book 20 units from the safety stock.
…			
12	zzzzzzz-100	65	Action needed. Book 5 units from the safety stock.

Finally, the output is exported as a CSV file, for the team to analyze.

Moreover, at the end of each retail season, the model will be revisited to fine-tune the hyperparameters and understand if new materials were/are being converted to AA.

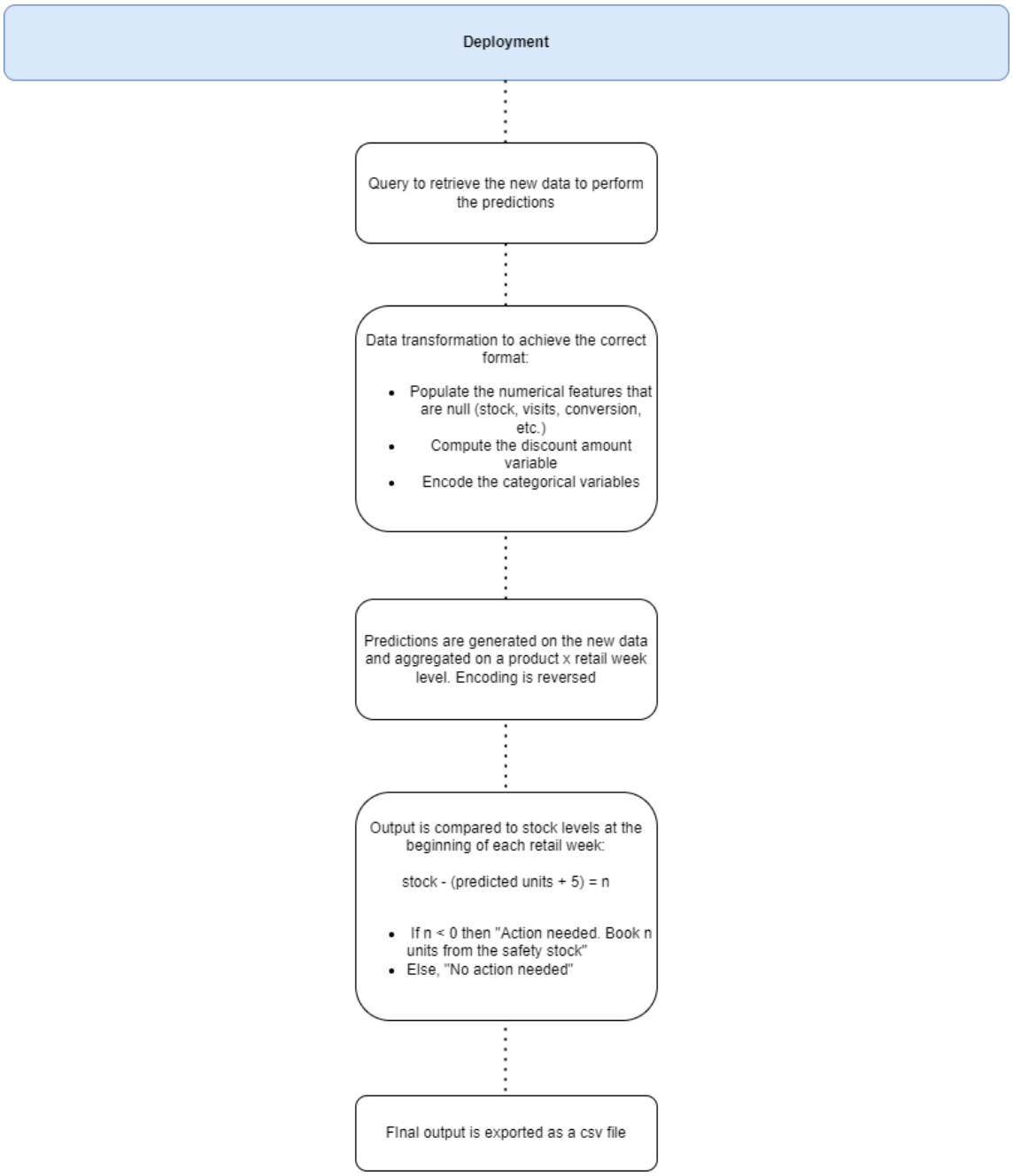

Figure 9 - Workflow of the Project Deployment

4. RESULTS AND DISCUSSION

The primary goal of this chapter is to evaluate the performance of the models and examine their effectiveness in addressing the problem at hand. Three algorithms were evaluated: Random Forest, XGBoost, and LightGBM. Furthermore, this evaluation was centered on two key performance metrics, Root Mean Squared Error (RMSE) and Mean Absolute Error (MAE), calculated on the test set, to understand how well the models capture the underlying patterns and make accurate predictions.

4.1. RANDOM FOREST RESULTS

The Random Forest Regressor, an ensemble method recognized for its successful implementations in major frameworks, underwent a rigorous evaluation in this project, with the results shown in table 5.

The model was first executed with the default parameters and yielded satisfactory results, with no signs of overfitting. While the overall error was not high (6.4), the RMSE was not ideal, with a score of 24.2, indicating the presence of high errors in the evaluation of the test set. Despite the good performance for footwear material codes, especially in adults, the model struggled in the equipment and apparel divisions, mainly with kids' items, significantly overestimating the values. Given the goal of this project, to forecast the demand units as accurately as possible to optimally manage the inventory levels, it became obvious that performing hyperparameter tuning was necessary.

As explained previously, the five hyperparameters were optimized with the Optuna library in Python, with 50 iterations. The high number of decision trees, 300 (n_estimators) increased the computational time significantly, but also improved the RMSE score. As for the maximum depth of the decision trees, this parameter was set at 10. The min_samples_split (minimum samples needed to split a node) and min_samples_leaf (minimum samples needed at each leaf) were set at 4. The optimal max_features was found to be 'sqrt', meaning that the model will take the square root of the total of features as the number of features to account for when searching for the best split.

The performance of the model improved with the optimal hyperparameters, especially the RMSE score (18.1). The algorithm continued to exhibit very good predictions in footwear, for all genders. Furthermore, there was a substantial improvement in kids' equipment and apparel. Although the predictions were still overstated, they were closer to the truth. While the algorithm produced favorable results, its extensive computational time proved to be a significant drawback. This high processing demand made the model less suitable for practical use, as it was not the most efficient choice compared to other options.

Table 5 - Random Forest Results

	RMSE		MAE	
	Train	**Test**	**Train**	**Test**
Before hyperparameter tuning	23.7	24.2	6.1	6.4
After hyperparameter tuning	17.8	18.1	4.4	4.8

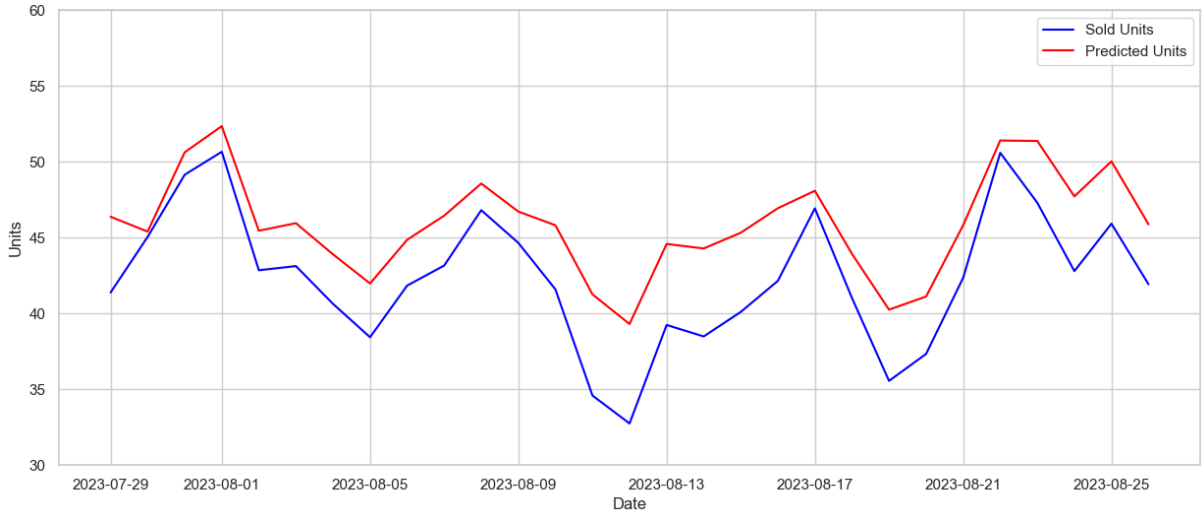

Figure 10 - Random Forest Results: Predictions vs Actuals

4.2. XGBoost Results

The second method evaluated was XGBoost Regressor, a technique that shines in many applications (Chen and Guestrin, 2016). The model was initially implemented and evaluated using default parameters, with the results of the evaluation showcased in table 6. Upon analysis, a notable difference was seen between the test and training scores, especially for RMSE (5.9 difference between train and test evaluation), suggesting a potential overfitting. This highlights the need for hyperparameter optimization to improve the model's performance and ensure it generalizes well to unseen data. By fine-tuning these parameters, a more balanced and effective predictive model can be achieved. Additionally, while analyzing the test predictions, it was discovered that the algorithm generated favorable outcomes in the footwear division, predominantly for adults. However, it overestimated substantially the predictions for equipment across all genders and apparel for women.

As mentioned earlier, the optimization of the six hyperparameters was conducted using Optuna with 50 iterations. Ultimately, the model was implemented with 512 decision trees

(n_estimators) and a maximum depth of 7, providing sufficient granularity for each tree. The optimal ratio of columns subsampled for each tree construction (colsample_by_tree) was determined to be 0.934, while the proportion of observations randomly sampled for each tree (subsample) was set at 0.78. This setup was designed to enhance model performance by ensuring a balanced and representative subset of data for each tree. Additionally, a learning rate of 0.04 was chosen to mitigate overfitting, as the default parameter (0.3) was deemed too high for this context. Finally, the min_child_weight parameter was established as 10 (minimum sum of instance weights required in a child node).

The model saw great improvements after tuning the hyperparameters, as seen in table 6, especially for RMSE (0.5 difference between train and test evaluation) effectively addressing the overfitting problem. While exhibiting good results in the apparel and footwear divisions, this model still encountered challenges in performance (RMSE of 19.3), particularly evident in equipment for kids, despite the improvement in equipment for adults.

Table 6 - XGBoost Results

	RMSE		MAE	
	Train	**Test**	**Train**	**Test**
Before hyperparameter tuning	21.3	27.2	5.2	7.1
After hyperparameter tuning	18.8	19.3	4.0	4.5

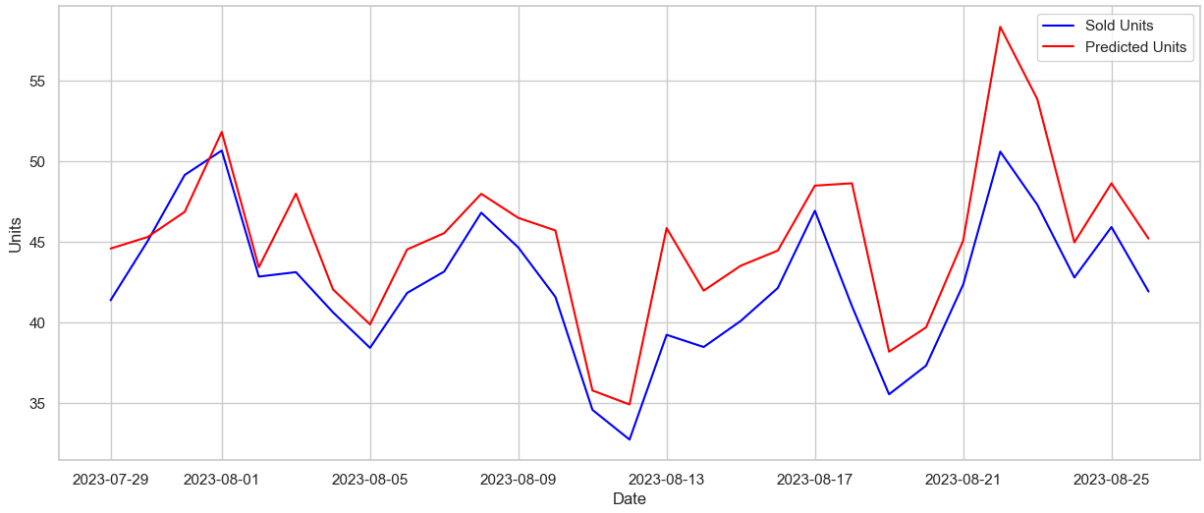

Figure 11 - XGBoost Results: Predictions vs Actuals

4.3. LIGHTGBM RESULTS

The last method examined in the project was the LightGBM Regressor, renowned for its efficient gradient-boosting tree implementation (Januschowski et al., 2022). Once again, the algorithm was initially implemented and assessed using default parameters, with the results presented in table 7. An initial RMSE score of 28.1 on the test set highlighted the need for hyperparameter optimization, as this model encountered difficulties with predicting kids' items across all divisions, particularly in apparel, and female equipment, where it consistently overestimated the number of units. And, as mentioned previously, the primary objective of this project was to accurately forecast demand units to manage inventory levels effectively.

Once again, the optimization process was implemented using Optuna, over 50 iterations. This iterative approach allowed for a thorough exploration of the hyperparameter space to identify the most effective configurations for the model. The algorithm was built using 288 decision trees (n_estimators) and configured with 376 leaves (num_leaves), specifying the maximum number of leaves each tree could have. This setup aimed to balance model complexity and computational efficiency, ensuring a robust performance. The optimal setting for the ratio of columns subsampled for each tree construction (colsample_by_tree) was determined to be 0.859, indicating that approximately 85.9% of the features were used when building each tree. Additionally, the proportion of observations randomly sampled for each tree (subsample) was set at 0.86, meaning that 86% of the training data was utilized for constructing each tree. Finally, the min_data_in_leaf parameter was specified as 37, representing the minimum number of data points required to be present in a leaf node. Additionally, the learning rate was set to 0.005.

After fine-tuning the hyperparameters, significant improvements were observed in the model's performance, as illustrated in Table 7. LightGBM achieved great results overall (MAE of 4.3), with an incredible performance in footwear. The RMSE score also decreased significantly (16.4), with improvements in the overall performance for kids. However, the model continued to slightly struggle to capture the trend in women’s equipment. Finally, it’s noteworthy to mention that this model stood out with its exceptional execution time, promptly delivering results.

Table 7 - LightGBM Results

	RMSE		**MAE**	
	Train	**Test**	**Train**	**Test**
Before hyperparameter tuning	27.5	28.1	6.5	6.9
After hyperparameter tuning	16.1	16.4	4.1	4.3

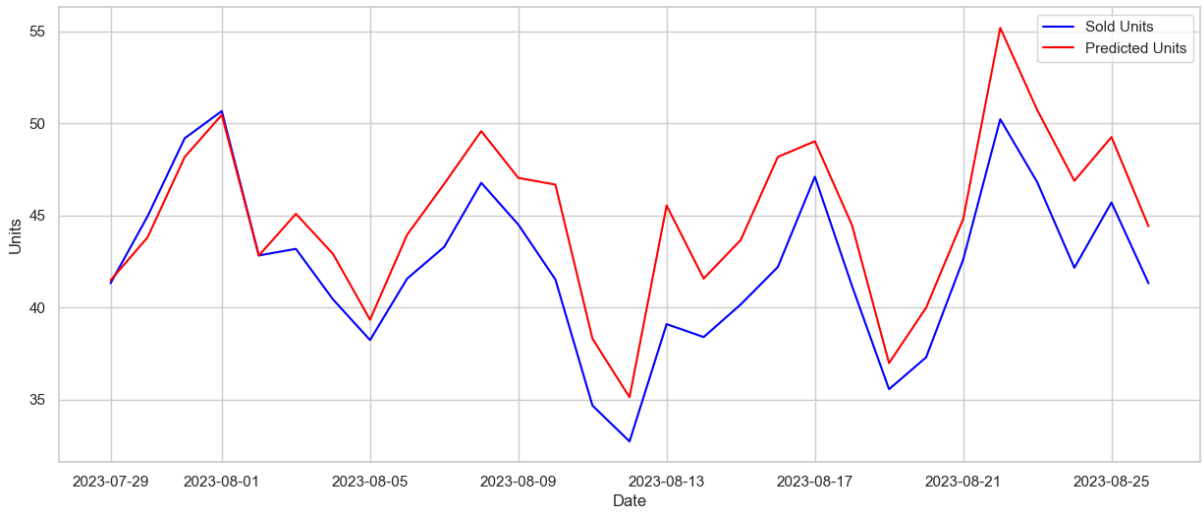

Figure 12 - LightGBM Results: Predictions vs Actuals

4.4. SUMMARY

Upon the first iteration, with the default hyperparameters, Random Forest outperformed LightGBM and XGBoost. While the latter exhibited the lowest RMSE and MAE scores on the training set, it displayed the highest scores on the test set, suggesting a potential issue with overfitting. This discrepancy confirms the importance of optimizing hyperparameters to achieve better performance. Despite this, the overall initial performance of the algorithms was satisfactory, although with challenges, such as with the Equipment division and Kids. After tuning the hyperparameters, there was an improvement in performance for all models, especially for XGBoost, handling the overfitting issue seen previously. The MAE score was very similar among the three outputs, but LightGBM exhibited the lowest RMSE and MAE scores on the test set. As observed in table 8, there is a difference in magnitude between the two metrics, which led to further investigation. As mentioned previously, RMSE penalizes large errors, which were found to happen in some of the lowest-selling material codes, mostly in Equipment and Kids. Random Forest encountered challenges in performance, particularly with Kid's equipment and apparel. XGBoost similarly faced difficulties with Kid's equipment, while LightGBM struggled specifically with women's equipment. Overall, the models yielded generally good results, but it was expected that they would not perform optimally for all 1226 AA products.

Despite the similar MAE scores, LightGBM stands out as the most stable method, displaying the smallest RMSE score. This observation suggests that while the models perform similarly in terms of average error, LightGBM demonstrates a slightly superior accuracy in predicting

individual data points. Additionally, it showed superior computational efficiency, surpassing the other models in processing time. As a final observation, it is strongly posited that all three algorithms have succeeded with the task at hand, demonstrating adeptness in capturing different patterns and trends. This success not only validates the robustness of tree-based models but also reinforces their capability to handle large and complex datasets, proving that these methods offer very attractive solutions in a retail setting.

Table 8 - Results of the models with the final hyperparameters

Model	RMSE		MAE	
	Train	Test	Train	Test
Random Forest	17.8	18.1	4.4	4.8
XGBoost	18.8	19.3	4.0	4.5
LightGBM	16.1	16.4	4.1	4.3

4.5. LIMITATIONS

The primary challenges faced during this project were related to the data. A significant issue was the lack of one single source of truth, resulting in the existence of multiple data sources for each field. This increased the difficulty of the data collection, as navigating through various sources to understand the best one was complex and labor-intensive. Additionally, it was discovered that many tables in Snowflake were still being updated manually. This manual update process is not ideal, as it introduces the potential for human error and inconsistencies, and it undermines the efficiency and reliability of the data management system. Automating these updates would improve data accuracy and streamline the workflow. Moreover, the project was also impacted by the lack of transparency regarding data issues. Issues such as delays in data refreshes and the decommissioning of tables are not adequately communicated to users. This lack of timely information affects the ability to prepare effectively for changes, resulting in disruptions to the business flow. Furthermore, the imbalance among division classes significantly impacted the models' performance. The business has a clear footwear focus and equipment is not as representative, causing the models to struggle with performance in the latter.

5. CONCLUSIONS AND FUTURE WORKS

The COVID-19 pandemic brought significant disruptions to numerous industries, and the sportswear retail sector was no exception. With restrictions limiting in-person shopping, consumers increasingly turned to online channels, leading to an increase in digital sales. Despite this rise, many companies such as this one continued to rely on outdated manual processes to manage their digital operations. Recognizing the need for innovation and efficiency, this project was initiated to improve the safety stock booking process for AA products within the digital channel. Leveraging machine learning techniques, the aim was to streamline and optimize inventory management practices, ensuring adequate stock levels while minimizing costs and maximizing customer satisfaction. This approach not only helps in maintaining a balance between supply and demand but also minimizes the risks associated with stockouts and excess inventory. Furthermore, the process was constructed using LightGBM, chosen for its robust performance in handling the dataset's skewed distribution and the presence of outliers.

Moreover, it is important to emphasize the importance of adding factors such as promotions, website traffic, conversion, and inventory, as they contribute to the accuracy of the models. During the feature selection process for each algorithm, these features showcased great importance, unlike some product characteristics such as silhouette and division. Additionally, considering the timeframe covered by the dataset, it's reasonable to assume that the data is still influenced by the COVID-19 pandemic. During the pandemic, the digital channel experienced unprecedented growth, driven not only by the necessity of consumers to stay indoors but also by their focus on health and home-based exercise, contributing to increased sales in the sportswear sector (Cho et al., 2023). Subsequently, a decline in business was observed following this period, having impacted the business for longer than expected.

Regarding future work, there are various directions the project could follow. The next step could involve optimizing the queries and migrating the project to Databricks, creating a Tableau dashboard to showcase the results, which would streamline and automate the process. This automation would eliminate the need for manual intervention and refreshment of the process, improving efficiency. Furthermore, the imbalance in the division classes could be exploited. Incorporating non-AA equipment items into the dataset, thereby increasing the representation of this class, could improve the models' performance.

As for data improvements, if it becomes available, incorporating size information would be a great addition, bringing an extra layer to the process. Additionally, incorporating country-specific data could provide valuable insights into regional variations in consumer behavior. For instance, analyzing differences between purchasing patterns in northern and southern countries, influenced by factors like climate, economic disparities, and cultural norms, could offer opportunities to refine forecasting models. Lastly, a potential future extension of this project would involve advancing from managing the safety stock to actively planning

inventory. Given that seasonal planning typically occurs months in advance, implementing a machine learning solution to forecast and plan inventory ahead of time could prove highly beneficial.

BIBLIOGRAPHICAL REFERENCES

Aburto, L., & Weber, R. (2007). Improved supply chain management based on hybrid demand forecasts. Applied Soft Computing, 7 (1), 136–44. https://doi.org/10.1016/j.asoc.2005.06.001

Alon, I., Qi, M., & Sadowski, R. J. (2001). Forecasting aggregate retail sales: Journal of Retailing and Consumer Services, 8(3), 147–156. https://doi.org/10.1016/s0969-6989(00)00011-4

Aggarwal, T. (2023). Master the Power of Optuna: A Step-by-Step Guide. Retrieved from Medium: https://medium.com/data-and-beyond/master-the-power-of-optuna-a-step-by-step-guide-ed43500e9b95

Archak, N., Ghose, A., & Ipeirotis, P. G. (2011). Deriving the pricing power of product features by mining consumer reviews. Management Science, 57(8), 1485–1509. https://doi.org/10.1287/mnsc.1110.1370

Arunraj, N. S., Ahrens, D., & Fernandes, M. (2016). Application of SARIMAX model to forecast daily sales in food retail industry. International Journal of Operations Research and Information Systems, 7(2), 1–21. https://doi.org/10.4018/ijoris.2016040101

Bratina, D., & Faganel, A. (2008). Forecasting the primary demand for a beer brand using time series analysis. Organizacija, Journal of Management, Information Systems and Human Resources, 41(3), 116–124. https://doi.org/10.2478/v10051-008-0013-7

Breiman, L. (2001). Random Forests. Machine Learning, 45, 5–32. https://doi.org/10.1023/A:1010933404324

Chen, T.Q. & Guestrin, C. (2016) Xgboost: A Scalable Tree Boosting System. Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 785-794. https://doi.org/10.1145/2939672.2939785

Chen, Y., Wang, Q., & Xie, J. (2011). Online Social Interactions: A Natural Experiment on Word of Mouth versus Observational Learning. Journal of Marketing Research, 48(2), 238–254. https://doi.org/10.1509/jmkr.48.2.238

Cho, H., Wang, J. C. K., Kim, S., & Chiu, W. (2023). Increasing exercise participation during the COVID-19 pandemic: the buffering role of nostalgia. Frontiers in Psychology, 14. https://doi.org/10.3389/fpsyg.2023.1285204

Introduction to CRISP-DM. (n.d.). Retrieved from https://zllu2.github.io/python-business/08-crisp/

Da Veiga, C., Da Veiga, C. R., Catapan, A., Tortato, U. & Da Silva, W. (2014). Demand forecasting in food retail: A comparison between the Holt-Winters and ARIMA models. WSEAS

Transactions on Business and Economics. 11. 608-614. https://wseas.com/journals/articles.php?id=5004

Dellino, G., Laudadio, T., Mari, R., Mastronardi, N. & Meloni, C. (2015). Sales Forecasting Models in the Fresh Food Supply Chain. In Proceedings of the International Conference on Operations Research and Enterprise Systems, 419-426. https://doi.org/10.5220/0005293204190426

Evers, J. M., Tavasszy, L., Van Duin, R., Schott, D., & Gorte, F. (2018). Demand forecast models for online supermarkets. In E-groceries, digitalization and sustainability: Which governance, planning and regulation mix do our cities need? Molde University. https://research.tudelft.nl/en/publications/demand-forecast-models-for-online-supermarkets

Falatouri, T., Darbanian, F., Brandtner, P., & Udokwu, C. (2022). Predictive Analytics for Demand Forecasting – A comparison of SARIMA and LSTM in Retail SCM. Procedia Computer Science, 200, 993–1003. https://doi.org/10.1016/j.procs.2022.01.298

Fildes, R., Ma, S., & Kolassa, S. (2022). Retail forecasting: Research and practice. International Journal of Forecasting, 38(4), 1283–1318. https://doi.org/10.1016/j.ijforecast.2019.06.004

Gelder, K. v. (2024). E-commerce worldwide - statistics & facts. Retrieved from Statista: https://www.statista.com/topics/871/online-shopping/#topicOverview

Gilbert, K. (2005). An ARIMA supply chain model. Management Science, 51(2), 305–310. https://doi.org/10.1287/mnsc.1040.0308

He, Z., & Yu, S. (2020). Application of LightGBM and LSTM combined model in vegetable sales forecast. Journal of Physics. Conference Series, 1693(1), 012110. https://doi.org/10.1088/1742-6596/1693/1/012110

Hirche, M., Haensch, J., & Lockshin, L. (2021). Comparing the day temperature and holiday effects on retail sales of alcoholic beverages – a time-series analysis. International Journal of Wine Business Research, 33(3), 432–455. https://doi.org/10.1108/ijwbr-07-2020-0035

Januschowski, T., Wang, Y., Torkkola, K., Erkkilä, T., Hasson, H., & Gasthaus, J. (2022). Forecasting with trees. International Journal of Forecasting, 38(4), 1473–1481. https://doi.org/10.1016/j.ijforecast.2021.10.004

Jupyter. (n.d.). Retrieved from https://jupyter.org/

Lalou, P., Ponis, S. T., & Efthymiou, O. K. (2020). Demand forecasting of retail sales using data analytics and statistical programming. Management & Marketing, 15(2), 186–202. https://doi.org/10.2478/mmcks-2020-0012

Lazo, J. K., Lawson, M., Larsen, P. H., & Waldman, D. M. (2011). U.S. economic sensitivity to weather variability. Bulletin of the American Meteorological Society, 92(6), 709–720. https://doi.org/10.1175/2011bams2928.1

Lee, M., & Hamzah, N.A. (2010). Calendar variation model based on ARIMAX for forecasting sales data with Ramadhan effect. In Proceedings of the Regional Conference on Statistical Sciences, 349-361. https://api.semanticscholar.org/CorpusID:131281093

LightGBM. (2024). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/lightgbm-light-gradient-boosting-machine/

Loh, W. (2011). Classification and regression trees. Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery/Wiley Interdisciplinary Reviews. Data Mining and Knowledge Discovery, 1(1), 14–23. https://doi.org/10.1002/widm.8

Makridakis, S., & Hibon, M. (2000). The M3-Competition: results, conclusions and implications. International Journal of Forecasting, 16(4), 451–476. https://doi.org/10.1016/s0169-2070(00)00057-1

Marques, R. A. D. F. (2020). A comparison on statistical methods and long short term memory network forecasting the demand of fresh fish products. Master dissertation in Faculty of Engineering of the University of Porto. https://hdl.handle.net/10216/126819

Punia, S., Nikolopoulos, K., Singh, S. P., Madaan, J. K., & Litsiou, K. (2020). Deep learning with long short-term memory networks and random forests for demand forecasting in multi-channel retail. International Journal of Production Research, 58(16), 4964–4979. https://doi.org/10.1080/00207543.2020.1735666

Raizada, S., & Saini, J. R. (2021). Comparative analysis of supervised machine learning techniques for sales forecasting. International Journal of Advanced Computer Science and Applications, 12(11). https://doi.org/10.14569/ijacsa.2021.0121112

Random Forest. (2022). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/random-forest-hyperparameter-tuning-in-python/

Saha, P., Gudheniya, N., Mitra, R., Das, D., Narayana, S., & Tiwari, M. K. (2022). Demand Forecasting of a Multinational Retail Company using Deep Learning Frameworks. IFAC-PapersOnLine, 55(10), 395–399. https://doi.org/10.1016/j.ifacol.2022.09.425

Shukla, M., & Jharkharia, S. (2013). Applicability of ARIMA models in wholesale vegetable market. International Journal of Information Systems and Supply Chain Management, 6(3), 105–119. https://doi.org/10.4018/ijisscm.2013070105

Taylor, J. W. (2007). Forecasting daily supermarket sales using exponentially weighted quantile regression. European Journal of Operational Research, 178(1), 154–167. https://doi.org/10.1016/j.ejor.2006.02.006

Ulrich, M., Jahnke, H., Langrock, R., Pesch, R., & Senge, R. (2021). Distributional regression for demand forecasting in e-grocery. European Journal of Operational Research, 294(3), 831–842. https://doi.org/10.1016/j.ejor.2019.11.029

XGBoost. (2023). Retrieved from Geeksforgeeks: https://www.geeksforgeeks.org/ml-xgboost-extreme-gradient-boosting/

NOVA Information Management School
Instituto Superior de Estatística e Gestão de Informação

Universidade Nova de Lisboa