International Journal of Basic and Applied Sciences, 14 (5) (2025) 455-471

International Journal of Basic and Applied Sciences

Website: www.sciencepubco.com/index.php/IJBAS
https://doi.org/10.14419/b9w7t018
Research paper

Optimization of Food Supply Chain Management Using Machine Learning

Pujan Shailesh Kakkad *

Senior Systems Development Engineer, Amazon Robotics, Business Applications and Solutions Engineering Team, Austin, Texas
**Corresponding author E-mail: pujankakkad@gmail.com*

Received: July 28, 2025, Accepted: September 4, 2025, Published: September 14, 2025

Abstract

This research work aims to improve supply chain management using predictive modeling techniques. First, we gained a comprehensive understanding of the company's operations, internal policies, product categories, customer base, and procurement processes. Through a literature review, we identified machine learning algorithms, including multivariate linear regression, support vector regression (SVR), regression decision trees, and neural networks, and implemented these algorithms using libraries such as Scikit-learn and TensorFlow. After data preparation and normalization, we trained and evaluated the models using various performance metrics such as R2, MAE, MAPE, MSE, and RMSE. Among all the models, the SVR algorithm had the highest predictive performance. After applying the predictive model, supply chain metrics improved: monthly revenue per supermarket decreased by 36.36%, monthly revenue per category decreased by 39.74%, and total revenue decreased by 25.95%. These findings confirm the effectiveness of machine learning in improving demand forecasting and supply chain efficiency. Recommendations include creating a data-driven culture within the company, exploring other regression algorithms, integrating historical data for new products, and continuously improving the predictive model. In addition, there is a need to improve evaluation metrics and continuously monitor model performance to improve the process.

Keywords: *Supply Chain Optimization; Predictive Modeling; Machine Learning; Support Vector Regression (SVR); Demand Forecasting.*

1. Introduction

In today's competitive, data-driven market, effective supply chain management is critical for a product development business. Accurate demand forecasting and inventory optimization are key drivers of profitability, customer satisfaction, and operational stability. This is important for businesses that distribute fast-moving consumer goods, as reducing waste transportation and storage costs while maintaining product quality is a constant challenge [1 - 3].

The focus of this study is on a medium-sized distributor company in Pune, which has been supplying local supermarkets under the brand name "Union Distributors" for more than 15 years. The company uses a just-in-time (JIT) replenishment model and seeks to maintain inventory at minimum levels to avoid losses due to product expiration and reduce storage costs. Weekly inventory decisions are based on a combination of digital data and manual inventory tracking. However, the lack of advanced forecasting tools and reliance on partially manual systems has led to frequent returns of perishable items and reduced inventory levels [4 - 6].

To address these operational challenges, this study proposes the use of machine learning (ML) techniques for inventory forecasting and management. Using historical inventory and sales data, the goal is to build a forecasting model that more accurately estimates product demand across the company's supermarkets. The work was built using the CRISP-DM approach, which starts with a deep understanding of the business, followed by data collection, cleaning, and mining using various regression-based machine learning algorithms [7 - 9].

The study evaluated the effectiveness of four major supervised learning models—Multiple Linear Regression (MLR), Support Vector Regression (SVR), Decision Tree Regression, and Artificial Neural Network (ANN). The models were trained and tested on well-processed data, and their performance was evaluated using evaluation metrics such as coefficient of determination (R2), mean absolute error (MAE), mean absolute percentage error (MAPE), and root mean absolute error (RMSE).

A recent study found that predictive models can help make better purchasing decisions, reduce inventory waste, and improve overall supply chain responsiveness. In addition to technical implementation, the study also recommends creating a data-driven culture within the organization, integrating predictive tools into day-to-day operations, and expanding data collection to support future optimization efforts.

2. Literature Review

Odimarha et al. (2024) [10] conducted a comprehensive analysis of how machine learning affects supply chain and logistics optimization in the oil and gas industry. The study focused on the use of predictive analytics and optimization algorithms to improve the efficiency of

asset tracking, demand forecasting, and route planning. Their study identified key challenges, including data fusion and real-time monitoring, while proposing some solutions, such as hybrid machine learning models that combine supervised and unsupervised learning techniques to handle complex data sets.

Pasupuleti et al. (2024) [11] explored how machine learning can help improve the agility and sustainability of supply chains, especially in logistics and inventory management. They highlighted optimization techniques such as reinforcement learning and tree gradient boosting to reduce delivery time, reduce material shortages, and improve adaptive planning. The study also examined the dual role of machine learning in environmental sustainability and cost optimization, emphasizing the importance of smart inventory systems and sustainable transportation planning.

Akbari and Do (2021) [12] conducted a systematic review that sheds light on current trends and future directions in machine learning in logistics and supply chain management. They classify machine learning applications into five areas: forecasting, inventory management, demand planning, transportation, and risk management. The review highlights that support vector regression (SVR), neural networks, and decision tree algorithms are the most used methods and show gaps in understanding and engaging with enterprise systems – key considerations for future research.

Yang et al. (2023) [13] focus on supply chain risk management (SCRM) and propose a literature-based framework that incorporates machine learning techniques for risk detection and mitigation. The study describes how anomaly detection algorithms, Bayesian networks, and ensemble models can be used to identify supplier and operational risks. The authors propose a combination of real-time data analytics, IoT sensors, and machine learning tools to improve threat response.

Dayos et al. (2025) [14] provide a broader perspective by examining the applications of artificial intelligence in supply chain management, including but not limited to machine learning. They are exploring the integration of natural language processing (NLP), computer vision, and deep learning into shopping automation, quality inspection, and customer behavior prediction. The paper highlights the importance of ethical AI, explainable models, and human-computer interaction systems to maintain transparency and accountability.

Sayad et al. (2024) [15] developed an innovative approach to improve the performance of machine learning models using domain-based adversarial neural networks (DANNs). Their research showed that incorporating domain adaptation techniques can significantly improve the prediction accuracy of supply chain models in different datasets and settings. This is particularly useful in multinational operations where these characteristics vary by region and time.

Day et al. (2024) [16] examined how artificial intelligence (especially machine learning) can improve the supply chain of small and medium enterprises (SMEs) in Vietnam. They stated that supervised learning was used to make predictions and unsupervised learning was used to detect inventory flow anomalies. Their results showed that machine learning can improve responsiveness and resilience to disruptions, especially in situations of tension and resource constraints.

A review of previous studies shows that machine learning has shown clear value in supply chain applications, particularly through deep learning techniques such as predictive analytics, ensemble models, forecasting, routing, risk detection, and inventory optimization [10], [141]. Efforts have also been made using hybrid approaches that combine supervised learning as well as domain adaptation techniques to address heterogeneous datasets and cross-region alignment [10], [15]. However, there are gaps in the existing literature. First, methods such as support vector regression (SVR) and different neural networks are highly sensitive to noise, bias, and sparse data, requiring careful kernel selection and scaling, which limits their reliability in real-world face-to-face and contextual applications [12]. Second, some studies strictly evaluate domain transfer issues, where models trained on one data or region perform poorly when transferred to another region, despite its prevalence in multinational operations [12], [15]. Third, anomaly detection and prediction algorithms are often proposed, but data fusion pipelines connecting IoT remain unexplored at the end [10], [13]. In addition, issues of explainability and ethics are not adequately addressed, as deep, ensemble models often lack transparency and interpretability, creating barriers to stakeholder adoption [14]. Another limitation is the lack of standard benchmarks for advertising demand, despite evidence that increases in advertising lead to some degree of stationarity in the forecast [7]. Finally, biased evaluation metrics and inconsistent datasets reduce reproducibility and hinder meaningful comparisons between studies [6].

This study directly addresses these drawbacks by implementing and evaluating different predictive modeling techniques—multivariate linear regression, support vector regression (SVR), regression decision trees, and neural networks using Scikit-learn networks. (R2, MAE, MAPE, MSE, and RMSE) for robust performance evaluation. Contrary to previous findings on SVR's sensitivity to noise [12], this study shows that SVR, properly tuned with the help of a systematic preprocessing pipeline, achieved the highest predictive accuracy on a given supply chain dataset. Measurable improvements were achieved in the supply chain using the selected prediction model, including an increase of 36.36% in monthly revenue per supermarket, 39.74% in monthly revenue per category, and a 25.95% increase in total revenue. These improvements validate the effectiveness of predictive modeling in demand forecasting and operational efficiency. In addition to the technical contributions, the study addresses gaps in past work by creating a data-driven culture, integrating historical data on new product categories, and continuously monitoring model performance with robust evaluation metrics. In this way, this study not only provides evidence for the use of machine learning in supply chain management but also provides practical strategies to overcome the methodological and operational limitations identified in previous studies.

A comparative table of ML techniques in cited studies is shown in Table 1.

Table 1: Comparative Table of ML Techniques in Cited Studies

Ref	Year	ML techniques	Application	Strengths reported	Limitations
Pal, S. Charles, Emrouznejad & Gherman	2023	AI integration for sustainability (various ML approaches)	Transparency & sustainable SCM	Framework for sustainability-aware AI	Largely conceptual; few implementation details
	2023	AI + Blockchain integration (ML for verification/analytics)	Traceability and decision support	Highlights the synergy of AI+blockchain	Integration complexity; not an ML performance study
Suwignjo et al. Odimarha, Ayodeji & Abaku	2023	Predictive analytics (classification/regression)	Inventory performance improvement (FMCG case)	Practical case improvements in inventory KPIs	May be context-specific to a single FMCG firm
	2024	Predictive analytics, hybrid ML (supervised + unsupervised)	Logistics & optimization in oil & gas	Hybrid models for data fusion; route/asset optimization	Real-time monitoring and fusion remain challenging
Pasupuleti et al.	2024	Reinforcement learning; gradient-boosted trees	Agility & sustainability in logistics/inventory	RL for adaptive planning; GBDT for strong supervised performance	RL needs careful reward shaping and simulation

Yang et al.	2023	Anomaly detection, Bayesian networks, ensemble models	Supply chain risk management (SCRM)	Combines IoT + ML for anomaly/risk detection	Real-time pipeline and deployment details are limited
Daios / Dayos et al.	2025	NLP, computer vision, deep learning, ethical AI	AI in SCM: shopping automation, quality inspection, customer prediction	Broad view; emphasizes explainability & HCI	Emerging area—need production studies
Sayyad, Attarde & Yilmaz	2024	Domain-adversarial neural networks (DANNs) / domain adaptation	Improve cross-domain prediction accuracy	Demonstrated gains in cross-dataset accuracy	Requires labeled source data and careful training
Dey et al. (Day)	2024	Supervised + unsupervised learning (anomaly detection)	AI for SME supply-chain resilience (Vietnam)	Practical benefits for SMEs: improved anomaly detection	Resource constraints for SMEs limit model completeness

3. Methodological Framework

The main objective of this work was to optimize the supply chain of "Union Distributors", a food distribution company based in Pune, and develop a weekly profit forecasting model for supermarkets. The focus was on improving procurement efficiency through machine learning and reducing costs through comprehensive open source tools.

The company's historical data from January 2020 to December 2021 was used for this study. While product prices and external supply chain information (such as supplier orders or customer inventory) were not incorporated, proportional prices and shelf life were used to simulate demand. Data collection included interviews, direct observations, and a review of supporting documents from relevant departments.

The dataset used in this study consisted of internal company records from January 2020 to December 2021, with two years of completed operations. Overall, the dataset contained a total [enter number, i.e., 24 months × N products × M supermarkets] observations, including product categories, proportional prices, seasonality, and the number of sales in each supermarket in Pune. These two years were chosen to ensure a representativeness of seasonal demand fluctuations such as festive seasons and monsoon-induced consumption changes, as well as disruptions caused by the COVID-19 pandemic. While this temporary number strengthens the data set's ability to capture a wide range of activity patterns, it also implies potential downsides: pandemic-related restrictions, panic buying, and supply disruptions can temporarily distort demand patterns, creating an anomaly that affects model learning and forecasts.

Another important limitation arises from extracting product prices and external supply chain information, such as supplier delivery times, customer inventory levels, and order histories. By relying on proportional prices and timing to simulate demand, the model captures product competition and compensation, but does not capture price mix and broader market dynamics. Therefore, the prediction model is tailored to the internal context of the "Union of Distributors" and may not generalize to markets where pricing strategies, supplier changes, and customer behavior play a more direct role in shaping demand in other firms and markets. Nevertheless, this approach is consistent with developing a practical forecasting tool that can be used internally, while at the same time highlighting opportunities for future work to integrate external data sources to improve their robustness and cross-context applicability.

This study adopted an experimental approach based on a modified CRISP-DM method. The main stages included business definition, data preparation, algorithm selection (multiple linear regression, support vector regression, decision tree, neural network), training and evaluation of the final model using standard regression metrics (R^2, MAE, and RMSE), and finally internal system performance evaluation.

A stratified sampling method was used, where 80% of the data was used for training/validation and 20% for testing. This applied and explanatory study provides a practical AI-based solution to improve inventory decisions and provides a reference for similar initiatives in the wholesale and retail industries.

4. Study Development - Phase 1: Business Understanding

This phase details the information gathered from the interview with the manager of the distribution company.

4.1. Company Description

This distribution company is based in Pune and has been distributing "Union Distributors" products in supermarkets in Pune for 15 years. The company's goal is to get expired products in front of customers in as many supermarkets as possible, thus avoiding losses. Product orders are placed through the supplier's online platform, which displays the product catalog and generates purchase orders. Replenishment orders are placed every Friday at various supermarkets.

Inventory control is carried out through an internally used web application that contains records of receipts and shipments from the warehouse. It is important to note that records of products that are not present in the physical inventory are maintained. Over the years, the company has continuously streamlined its business processes, and its main internal policy is to adopt "zero inventory", which means that during the procurement process, once the product arrives, it can be delivered to other supermarkets in Pune in a just-in-time (JIT) mode. This is because not all of these products have a long shelf life.

Table 2 shows all categories of products distributed by Union distributors in Pune, along with a brief description of the products available in each category. More notable are "bread" and "snack" products, with 21 products.

Table 2: Categories of Distributed Products

Name	Description	Expiration Margin	Number of Products
Beverages	Drinkable liquid product.	365 days	9
Pastries	Cakes and assorted buns made from flour and yeast.	15 days	6
Cereals	Product made from seeds enriched with vitamins and nutrients.	90 to 120 days	10
Breads	Products made from bread with nutrients and vitamins.	15 days	21
Snacks	Nutritious products to provide energy, such as cookies or snack foods.	150 days	21
Spreads	Products that can be spread, enriched with nutrients and vitamins.	90 to 120 days	4
Total			71

Table 3 lists all the local supermarkets to which products from the distribution company Union Distributors are delivered, along with their respective addresses. A total of 12 local supermarkets are included, with the most prominent chains being Big Bazaar, Reliance Fresh, D-Mart, and Spencer's.

Table 3: List of Local Supermarkets Supplied by Union Distributors with Corresponding Addresses

Speed (rpm)	Load (Ω)
Big Bazaar – Connaught Place	Rajiv Chowk, Block A, Connaught Place, New Delhi – 110001
Big Bazaar – Karol Bagh	Ajmal Khan Road, Karol Bagh, New Delhi – 110005
Reliance Fresh – Indiranagar	12th Main Road, Indiranagar, Bengaluru – 560038
Reliance Fresh – Koramangala	5th Block, Koramangala, Bengaluru – 560095
More Supermarket – T. Nagar	North Usman Road, T. Nagar, Chennai – 600017
More Supermarket – Velachery	100 Feet Bypass Road, Velachery, Chennai – 600042
D-Mart – Thane West	Ghodbunder Road, Thane West, Mumbai – 400607
D-Mart – Andheri East	JB Nagar, Andheri East, Mumbai – 400059
Spencer's – Ballygunge	Gariahat Road, Ballygunge, Kolkata – 700019
Spencer's – Salt Lake	Sector 1, Salt Lake, Kolkata – 700064
Vishal Mega Mart – Sector 14	Sector 14, Gurgaon, Haryana – 122001
Vishal Mega Mart – Sector 18	Sector 18 Market, Noida, Uttar Pradesh – 201301

4.2. Procurement Process

This process is carried out by the distribution company, in which two other entities participate: the supermarket, which plays the role of customer, and the supplier of joint products. Within the distribution company entity, three roles can be seen participating throughout the process; however, it is the store manager role that is involved in most activities. The process was modeled (Figure 1) using the Bizagi Modeler tool.

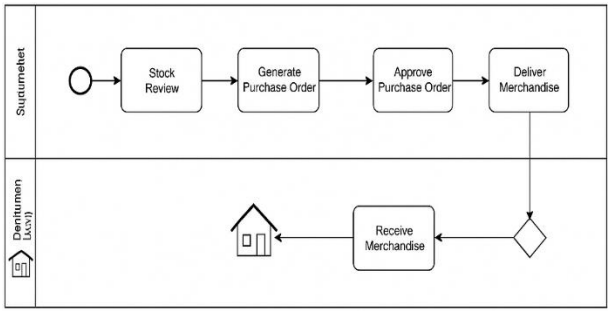

Fig. 1: Distribution Process Modeled in Bizagi.

As shown in Fig. 1, the food procurement process of a distribution company consists of six macro activities, which are carried out in the following order: receiving raw materials, creating a purchase order, confirming a purchase order, creating a purchase order, receiving goods, and delivering goods.

To better understand the workflow associated with the processing of Union distributors' products in local supermarkets, each of the macro activities is described below.

The first macro activity begins when the store manager, the logistics officer of the distributor, visits all the local supermarkets where the company's products are distributed. They then must go through the supermarket entrance control, where they have to show their ID and sign an internal supermarket security document to register their arrival. The inspector then visits each module of the supermarkets, observes the quantity of products of the "Union Distributors", and enters it on the physical sheet.

Once you have entered all the measurements, you need to enter the local supermarket warehouse with the help of the supermarket warehouse supervisor.

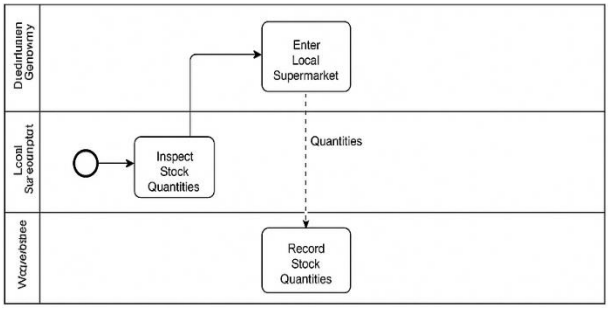

Fig. 2: Shows The Macro Activity of Stock Review, along with the Roles and Entities Involved.

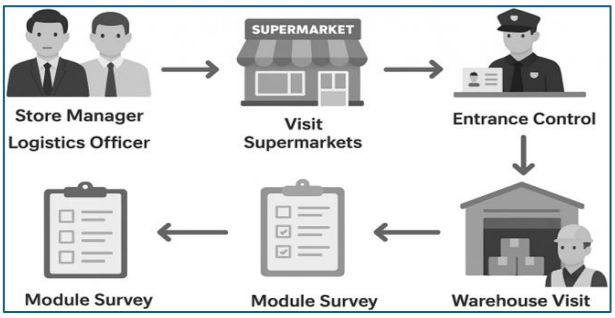

Fig. 2: Macroactivity: Stock Review.

After checking all the products in the supermarket, the store manager starts counting the products. A product of less than 36 units—the equivalent of 3 boxes, given that each box contains 12 different items from Union distributors—is considered a low-volume product. In these cases, the manager adds these products to the purchase order and calculates the quantity to be filled per batch. The order is then sent to the supermarket for review, and after the supermarket has received signature and approval, the order, signed by the store manager, is delivered to the distributor for final approval.

Upon receipt of the purchase order, the distributor's manager reviews it carefully to identify any errors. If discrepancies are found, they meet with the store manager to correct them. If the order is correct, it is approved and returned to the store manager. The next step is for the store manager to log into the supplier's online platform and enter the confirmed order details to formalize the purchase order.

Once the goods arrive at the distributor, the warehouse clerk checks the goods against the purchase order. If materials are not available, a letter of credit from the supplier is required. The selected products are then processed and prepared for delivery to local supermarkets.

Finally, the store manager issues an invoice to the supermarket when the product arrives. Even minor discrepancies, such as goods going missing during delivery, are the responsibility of the supermarket and will be investigated and resolved through a special process. Products are delivered in boxes with an invoice showing the completion of the purchase process and the necessary documents.

5. Study Development - Phase 2: Data Comprehension and Selection of Machine Learning Algorithms

In this phase, the available data were collected and used for analysis. First, the distribution company's data sources were analyzed: the inventory control system database and the physical files where missing product records are made. A description of the data was also provided, which was used to select the machine learning algorithms used in the work.

A request was made from the manager of the data supplier, who kindly agreed, noting that the prices of the products used for the analysis could not be disclosed.

The main source of data was the local supermarket information system, which records goods and commodities coming into the warehouse every week. Logistics staff printed reports from the inventory control system in Excel spreadsheets (.xlxs) on various physical documents containing logistics staff's signatures when checking stock of products in supermarkets and reporting on orders.

Reports of warehouse movements were made in 2020-2021 because the company had an inventory control system that they currently use. The data was collected in various Google Sheets files, which allowed data retrieval as the company's logistics staff had to remotely update the returned product information in the document. The data were then analyzed in Google Colab.

The inventory control system used by the distribution company was developed using MySQL, a popular relational database management system. The physical model of this system was created by MySQL Workbench using a database backup-based reverse engineering provided by the company.

This physical model consists of 10 relational tables, but only 6 tables were used for analysis and data extraction. These tables are described in detail in Table 4, which describes their names, their purposes, and their importance to the company's inventory and logistics.

The database schema includes typical elements such as:

- Primary keys and foreign key relationships between tables,
- Table fields capturing inventory transactions, product details, users, and stock movements,
- Proper normalization to reduce redundancy and ensure data consistency across the system.

This model supports the company's operations by maintaining accurate records of product inflows, outflows, and inventory levels across multiple supermarket locations.

Table 4: Description of Database Tables with Authorized Access

Name	Description
Article	Records data of all distributed products.
Category	Records the categories of the distributed products.
Entry	Records general data on product entries into the warehouse.
Entry_Detail	Records detailed data on product entries into the warehouse.
Sale	Records general data on product exits or sales from the warehouse.
Sale_Detail	Records detailed data on product exits or sales from the warehouse.

5.1. Data Exploration

The obtained data was organized in Google spreadsheets (Google Inc., 2020), divided into separate files for each local supermarket where the products are distributed. Since all files have the same structure, the same cleanup process can be performed for each. It was noted that there were many records with blank spaces.

Table 5: Fields Found in the Inventory Report

Field Name	Data Type	Description
ProductCode	Alphanu-meric	Field used to identify products with a unique code, which is also used to generate purchase orders on the supplier's platform.
Product	Alphanu-meric	Full name of the product; similar products are distinguished by their weight in grams.
Active	Boolean	An indicator showing whether products are currently active or valid.
F_expiry	Numeric	Approximate number of days between the product's delivery date and its actual expiration date.
P_purchase	Numeric	Purchase price of the product in soles.
P_sale	Numeric	Selling price of the product in soles to the local supermarket.
Margin	Numeric	The difference between the selling price and the purchase price of the product.
Category	Text	Identifier for the category to which the product belongs.
Quantities	Numeric	Multiple fields where weekly data was obtained for the distributed products.

Table 5 shows the fields in the requested inventory report from the company manager. The data type and description of each field are displayed. This information is very important in selecting the features used for appropriate training.
In examining the data, it was noted that the "Asset" field contained products that the distributor had not considered when placing orders. Many of these products were marked "MAP" because they were either discontinued or not allowed to be distributed in local supermarkets. In addition, dates were not included in the "Date" field; it was used to determine the number of days between the product delivery date and the expiration date.
In addition, it was noted that the quantities mentioned in the claims for each product were multiples of 12; therefore, it was assumed that the claims were packaged in boxes and that there were 12 units of the product described in the boxes. It was also observed that supermarkets had products that were in high demand but did not generate high profits; therefore, they did not represent a significant percentage of the distributor's monthly revenue. The bread category contains some of the most profitable products, but there are panettone products that only appear seasonally, along with others that show good profits for three months of the year. However, the bread category is currently experiencing short-term declines in profitability, largely influenced by shifts in consumer spending patterns.
Data from weekly product returns were not found in the records of the distributor's inventory control system because they were entered manually. However, during the data collection process, they were updated in a Google Sheets file, which revealed that products with shorter lead times were more frequently returned.

6. Selection of Machine Learning Algorithms

There is a wide variety of machine learning algorithms that can be used to find the appropriate model; however, by using labels for the data obtained to train the model, supervised learning algorithms can provide a better solution. Regression algorithms were selected because they output numerical values and were based on research by Seyedan & Fereshteh (2020) related to demand forecasting.

6.1. Multiple Linear Regression

According to Hayes (2022), multiple linear regression, also known as multiple regression, is a statistical technique that uses several explanatory variables to predict the outcome of a response variable. The purpose of multiple linear regression is to model the linear relationship between the explanatory (independent) variables and the response (dependent) variables. Basically, multiple regression is an extension of ordinary least squares regression because it includes more than one explanatory variable.
Multiple and simple linear regression have different use cases. In some cases, adding independent variables can worsen the situation, which is called overfitting. On the other hand, adding independent variables creates relationships between them; not only are the independent variables potentially related to the dependent variable, but they are also correlated with each other. The optimal scenario is for all independent variables to be correlated with the dependent variable, but not with each other. Equation 1 shows the formula for this algorithm according to Amat Rodrigo (2016), which is shown below:

$$y_i = \beta_0 + \beta_1 x_{i1} + \beta_2 x_{i2} + \cdots + \beta_p x_{ip} + \epsilon \quad (1)$$

Where: β_0: is the intercept, the value of the dependent variable when all predictors are zero. β_1: is the average effect of a one-unit increase in the predictor variable Xi on the dependent variable Y, holding all other variables constant. These are known as partial regression coefficients.

- ϵ: is the residual or error, the difference between the observed value and the value estimated by the model.

Figure 3 shows an example of multiple linear regression with three variables: sales, TV, and radio. Thus, the graph of the function takes the form of a plane where the red dots represent the actual value and the plane shows the predicted values.

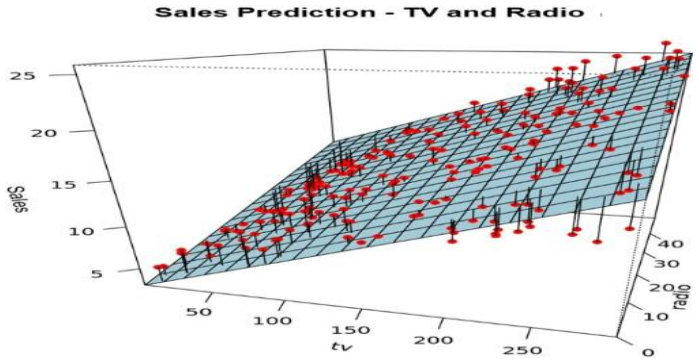

Fig. 3: Multiple Linear Regression of Sales on TV and Radio Advertising.

6.2. Support Vector Regression (SVR)

The Support Vector Regression algorithm is based on predicting numerical values; because the output is a real number, it becomes very difficult to predict the available information. However, the main idea is always the same: minimize the error, identify the hyperplane that maximizes the margin, considering that some error is tolerated.
Using Support Vector Regression in demand prediction can produce a lower mean square error than neural networks because the optimization function in SVR does not consider points beyond a certain distance from the training set. Therefore, this method leads to higher forecast accuracy.
According to [2], the formula for this algorithm is shown in Equation 2:

$$\min \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}(\varepsilon + \varepsilon *) \quad (2)$$

Where w is the magnitude of the vector or hyperplane, C is a constant and must be greater than 0. It determines the balance between the regularity of the function and the amount to which we tolerate deviations greater than the support bands. ξ and ξ* are the variables that control the error committed by the regression function when approximating the bands.
Literature [12] indicates that in most linear regression models, the objective is to minimize the sum of squared errors. He also concludes that SVR is a powerful algorithm that allows us to choose the degree of error tolerance through an acceptable error margin and by fine-tuning our tolerance to falling outside that acceptable error rate.

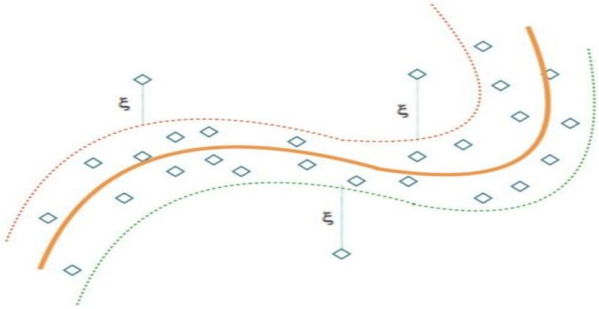

Fig. 4: Predictive Function with Support Vectors, Tolerance Margin, and Error Representation.

In Figure 4, one can see how the predictive function has a waveform on a plane with two variables X and Y, where the squares are the actual values. This explains why the data can be linear or nonlinear, since ultimately this model can be adjusted to the behavior of the data. One can also see the dotted lines, which indicate the tolerance margin assigned to variable C. Additionally, the variable ξ shows the error for each of the values that fall outside the support vectors. This algorithm allows for better tuning of its parameters to optimize the model for the required prediction.

6.3. Regression Decision Trees

Decision trees are a class of machine learning algorithms that create a map—actually, a tree—of questions to make a prediction. (Vandeput, 2021). Decision trees work by dividing the dataset into increasingly smaller subsets, and to obtain a prediction for a particular observation, the mean or mode of the responses from the training observations is used, within the partition to which the new observation belongs..

$$f(x) = \sum_{m=1}^{M} w_m \varphi(x; v_m) \quad (3)$$

Where Wm is the mean response in a particular region (Rm), Vm represents how each variable is partitioned at a particular threshold value; these partitions define how the feature space of R to the power of p is divided into M separate regions or hyperblocks.
Equation 3 shows the formula for this algorithm, whose objective is to minimize some type of error criterion. In this case, we want to minimize the residual sum of squares (RSS), an error measure also used in linear regression settings.
Figure 5 shows a subset containing the example data, where each partition of the domain is aligned with one of the feature axes, and the following is understood:
For a feature space of p, the space is divided into regions, in this case, 5 regions, and the tree generated from the sample of hyperblocks is located on the right, where comparisons are made to locate the data at the mean of the closest region.

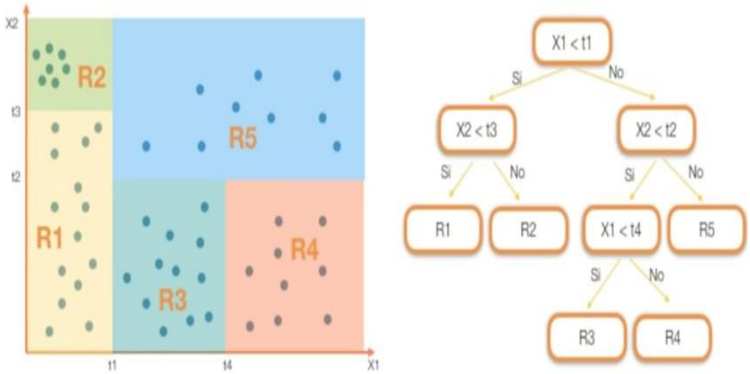

Fig. 5: Partitioned Feature Space and Corresponding Decision Tree.

6.4. Neural Networks

Artificial neural networks are a special type of machine learning algorithm inspired by the human brain. In other words, just as the neurons in our nervous system are capable of learning from prior information, artificial neural networks are capable of learning from information and providing answers in the form of predictions or classifications. here are two types of neurons in this network: detection neurons and processing neurons.
According to [11], the neurons in a network are divided into three groups:
1) Input neurons that receive information from the outside world.
2) Hidden neurons that process that information.
3) Output neurons that produce a conclusion.
Figure 6 explains how a neural network works, where X1 through Xn are the inputs it receives from the outputs of other neurons, W1 through Wn are the weights assigned to obtain one path as dominant over another, then a weighted sum is generated to subsequently give it an activation function, which transmits information generated by the linear combination of weights and inputs to the neuron's output.

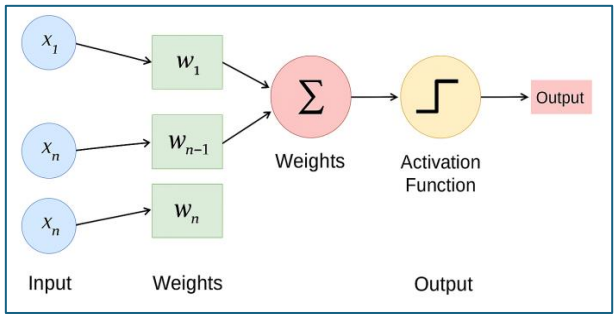

Fig. 6: Structure and Functioning of a Neural Network.

7. Study Development - Phase 3: Data Preparation

As shown in Table 2, the company's products are distributed in 12 local supermarkets, which requested different quantities for each product. Although they all have similar characteristics, it was decided to run a process for each supermarket and perform a predictive analysis for each one.
Data selection and cleaning were necessary to select the best algorithm for this dataset. The same steps were taken for all data obtained from each supermarket.
The data extraction and cleaning processes described in this section were conducted using Google Colaboratory, which provides free and fast computation. This required linking Google Sheets files showing the number of products requested and returned by the distributor's supermarket customers.
To begin the data extraction process, all available files were thoroughly analyzed. It was observed that the most important metric for determining product demand is the difference between the number of products delivered to supermarkets and the number of products returned to each supermarket. This difference represented the true demand and was collected into separate files for further processing.
To access these files, a connection was established between Google Colab and Google Drive. This required authorization to access Google Cloud Storage via Colab's drive.mount() function, which allows users to find a file path within their Drive and connect directly to files stored in the Colab environment.
Once the drive connection was established, another link to the Google Sheets was created with Google account credentials and appropriate access permissions. The auth and gspread libraries were used to manage authentication and facilitate access to spreadsheet content. After the connection was successfully made, the resulting authorization object variable was stored in gc, which allowed the data to be used.
After this alignment, the extracted data were imported into the working state. Using the Pandas library, Google Sheets reads the file contents and inserts them into a dataframe called df. This dataframe served as the basis for all subsequent data preparation and analysis steps.
After setting the connections correctly, the first 10 rows of the 'df' dataframe were displayed to check if the dataframe was not empty, as shown in Table 6.

Table 6: Printing the First 10 Rows of the 'Df' Dataframe

Supermarket	Product Code	Product	Category	Active	Expiry Days	Purchase Price	Sale Price
1	BEB003	Pineapple Pulp Drink 475 ml	Beverages	NO	NaN	NaN	NaN
1	BEB004	Peach Drink 300 ml	Beverages	NO	NaN	NaN	NaN
1	BEB005	Orange Pulp Drink 475 ml	Beverages	NO	NaN	NaN	NaN
1	BEB006	Apple Pulp Drink 475 ml	Beverages	NO	NaN	NaN	NaN
1	BEB011	Grape Juice 295 ml	Beverages	YES	365.0	4.50	6.00
1	BOLL002	Pre Pizza Union	Pastries	YES	15.0	15.00	17.00
1	BOLL003	Cinnamon Roll	Pastries	YES	15.0	1.30	2.50
1	CER001	Granola Lunchbox x 400gr.	Cereals	YES	120.0	4.50	6.00
1	CER002	Granola Lunchbox x 200gr.	Cereals	YES	120.0	4.50	6.00

Subsequently, to determine the data frame's dimensions, the '.shape' function from the pandas library was used, which displayed a total of 856 rows and 114 columns. It was also decided to obtain a concise summary of the data frame's columns using the '.info()' function, which showed that there were floats, integers, and objects.

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 856 entries, 0 to 855
Data columns (total 71 columns):  # Note: Only partial list shown
 #   Column          Non-Null Count  Dtype
---  ------          --------------  -----
 0   Supermarket     856 non-null    int64
 1   ProductCode     856 non-null    object
 2   Product         856 non-null    object
 3   Category        856 non-null    object
 4   Active          856 non-null    object
 5   ExpiryDate      486 non-null    float64
 6   PurchasePrice   486 non-null    object
 7   SalePrice       486 non-null    object
 8   Margin          486 non-null    object
 9   01-01-2020      486 non-null    float64
10   08-01-2020      486 non-null    float64
11   15-01-2020      486 non-null    float64
12   22-01-2020      486 non-null    float64
13   29-01-2020      486 non-null    float64
14   05-02-2020      486 non-null    float64
15   12-02-2020      486 non-null    float64
16   19-02-2020      486 non-null    float64
17   26-02-2020      486 non-null    float64
18   04-03-2020      486 non-null    float64
19   11-03-2020      486 non-null    float64
20   18-03-2020      486 non-null    float64
```

Fig. 7: Printing the Dimensions and Summary of the 'Df' Data Frame.

As can be seen in Figure 7, there are attributes or columns that contain different quantities than others; therefore, it was necessary to clean the 'df' data frame to work only with the products with complete data.

7.1. Data Cleaning

According to [10], data cleansing involves removing invalid data from a table or database by identifying incorrect, incomplete, or irrelevant entries. This process is necessary to maintain data integrity and ensure the reliability of data used for analysis. In the context of this work, data cleaning focused on verifying that there were no missing or empty values that could introduce anomalies during the training of the machine learning models. To perform the check, the .isnull(values.any() function from the Pandas library was used, which returns a True result, indicating the presence of incomplete values in the dataset.

In addition, it was important to confirm that the data frame did not contain any duplicate entries. However, since each product in the dataset is identified by a product code, it was not necessary to perform a double search. Due to the uniqueness of the identifier, some product records were missing from the data.

When the data were visualized in Figure 7, it was observed that the active products, as reported in Table 3, had all the data. Therefore, a data filter was applied to select all inactive or suspended products, as shown in Table 7.

Table 7: Dataframe Filter for Working with Active Data

Index	Super market	Product Code	Product Name	Category	Active	Expiry (Days)	Purchase Price	Sale Price
4	1	BEB011	Grape Juice 295 ml	Beverages	Yes	365	4.50	6.00
5	1	BOLL002	Pre Pizza Union	Pastries	Yes	15	15.00	17.00
6	1	BOLL003	Cinnamon Roll	Pastries	Yes	15	1.30	2.50
7	1	CER001	Granola Lunchbox x 400g	Cereals	Yes	120	4.50	6.00
8	1	CER002	Granola Lunchbox x 200g	Cereals	Yes	120	4.50	6.00
9	1	CER003	Granola Sport Union 50g	Cereals	Yes	120	4.50	6.00
10	1	CER004	Wheat Germ	Cereals	Yes	90	12.00	14.0

The data set was reassessed to confirm the absence of blank or incomplete values. After applying the filter to keep only active records, the .isnull().values.Any () function was used again, and the result is False, indicating that there were no missing values in the filtered data, called the "active" data frame.

Then, the .shape and .info() functions were used to assess the structure and description of the clean dataset. The results showed that the new dataframe, named "assets," consisted of 486 rows and 114 columns. This confirmed the modification of the measurements compared to the original data. In addition, data type classification revealed a dataset with 106 attributes with float values, 1 attribute with integer attributes, and 7 attributes with alphanumeric or text data. These details ensured a clear understanding of the structure of the data set before proceeding to further analysis or model training. Table 8 shows basic statistical details, such as percentile, mean, standard deviation, etc., for a given data frame. The '.describe()' function was used, which only uses numeric attributes for the purpose. In this case, it displays only columns containing numeric values and rejects those containing alphanumeric or Boolean values.

Table 8: Statistical Details of the 'Active' Data Frame

Statistic	Super market	Expiry Date	01-01-2020	08-01-2020	15-01-2020	22-01-2020	29-01-2020
Count	486.000	486.000	486.000	486.000	486.000	486.000	486.000
Mean	6.283	104.714	1.944	2.191	1.995	1.777	1.876
Std Dev	3.427	83.807	1.611	1.951	1.802	1.322	1.635
Min	1.000	15.000	0.000	0.000	0.000	0.000	0.000
25%	3.000	15.000	1.000	2.000	1.000	1.000	1.000
50%	6.000	120.000	2.000	2.000	2.000	2.000	1.000
75%	9.000	150.000	2.000	4.000	3.000	2.000	3.000
Max	12.000	365.000	8.000	8.000	8.000	6.000	5.000

Once measurements were obtained for each object in the 'active' data frame, columns containing values of 'object' types had to be checked to select only those containing numerical values.

The data types within the "active" data frame were examined, as shown in Figure 7, and it was determined that only quantitative data would be used to train the model. This decision was made since the selected machine learning algorithms were regression-based and required numerical input and output values for accurate predictions. The filter was applied to the "active" data frame to separate columns containing numeric data types, specifically those classified as int or float.

To do this, the .index function was used to extract column names, which helps to identify columns containing numeric values. Finally, the numeric columns specified in num_cols were removed from the "active" data frame and assigned to a new data frame named data. The final data frame, which contains only numerical values, serves as a clean and structured dataset used to train machine learning models.

8. Study Development -Phase 4: Training The Machine Learning Model

To train the data, the data universe had to be divided into three subsets: the training set, the validation set, and the testing set.
The total data was divided in an 80/20 ratio following the Pareto Law, as commonly used in supply chain management and warehousing and replenishment systems, according to [9].
Thus, 20% of the total data was assigned to the test set, and the remaining 80% was assigned to the training and validation data sets. These were again divided in an 80/20 ratio for each of them, as shown in Figure 8.

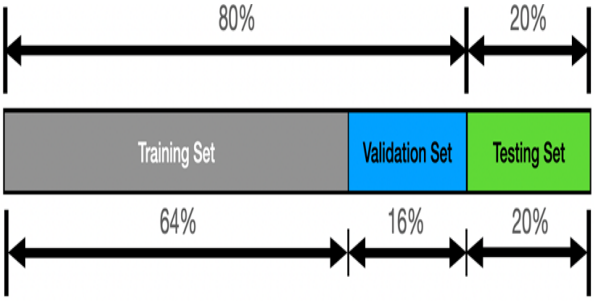

Fig. 8: Dataset Partitioning.

The training set was used to train the models with the selected algorithms for phase 4 of the work; the validation set was used to tune the parameters of the model with the best performance, as standard values were used during training; and finally, as mentioned, the test set was used to evaluate the predictions [7].
In this phase, each model was trained with the selected algorithms using the cleaned data. As shown in the previous phase, the numerical data in the first column are very different from the others. Therefore, data normalization was required to work with them in the same way. The 'StandardScaler' function from the Scikit-Learn library was used for this purpose. First, each variable had to be assigned attributes such that 'x' contained all the attributes for the study and 'y' contained the target values for each product.
After data normalization, the 'train_test_split()' function from the Scikit-Learn library was used to split the data into training and test sets. According to the Pareto principle, 80% of the data was selected for the training set (64% of the total) and the remaining 20% was selected for the validation set (16% of the total).

8.1. Training with The Multiple Linear Regression Algorithm

Training with this algorithm required the use of the Scikit-learn 'LinearRegression' library, where the training data subset was used to train the model. First, the imported multiple linear regression algorithm was assigned to the variable 'reg', and then, using the 'fit' function, the model was trained with the training variables 'X_train' and 'y_train'. Finally, the training score was 1.0, indicating that the training had 100% accuracy.

8.2. Training with The Support Vector Regression Algorithm

Training with this algorithm required the use of the SVR function from the Scikit-learn library. The training data subset was used to train the model, and the algorithm kernel was specified to be linear. The model width, epsilon, was set to 0.05 to allow for a smaller margin of error. The regularization parameter C, which allows weighting the importance of the error, was set to 1.
Training required a cross-sectional arrangement of the dataset in 'y_train' to avoid errors; therefore, the '.ravel()' function was necessary when training the model.
The training score obtained was 0.9992245296615283, a value that shows that the training has an accuracy of 99.9%.

8.3. Training with The Regression Decision Tree Algorithm

Training with this algorithm required the use of the 'DecisionTreeRegressor' function from Scikit-learn. For this purpose, the 'tree' library was imported, allowing the function to be used to access the regression decision tree algorithm.
After importing the necessary library, the algorithm was assigned to the variable 'dtree_reg' and then, using the 'fit()' function, the model was trained with the training data set found in the variables 'X_train' and 'y_train'.
The training score was 1.0, which shows that the trained model has 100% accuracy using the training data set.

8.4. Training with The Neural Network Algorithm

Training with this algorithm involved the use of the TensorFlow library (TensorFlow, 2022), which was imported under the alias tf. A subset of the prepared training data was used to fit the model. The neural network architecture consisted of three hidden neurons, each with three layers, designed to enhance training precision. The output layer, however, consisted of only a single neuron, as is standard for regression tasks. The model was compiled using the accuracy metric to monitor performance, and the mean_squared_error loss function was selected to evaluate how much accuracy was lost after each training iteration. After running the training process for 1000 iterations, the model achieved a training score of 0.529411792755127, indicating a relatively lower accuracy compared to other models used in the work. The Matplotlib library (Matplotlib, 2022) was used to generate the visualization in Figure 9. Despite 1000 iterations being used to train the model, the accuracy of the model is significantly lower than that of the other models trained. Starting with the 200th training session, the algorithm does not undergo significant changes to improve its accuracy. Therefore, time and computation could be saved in model training by reducing the number of iterations until the most suitable one is retained.

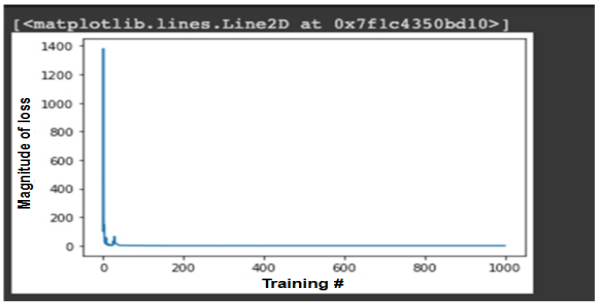

Fig. 9: Visualization of Training by Iterations According to the Loss Margin

9. Study Development - Phase 5: Evaluation of The Machine Learning Model

Regression metrics can help determine whether the regression model's predictions are accurate to a performance level, and it is necessary to evaluate these models to determine whether they are performing well [10,11,12,13].
In this phase, the 'r2_score', 'mean_absolute_error', and 'mean_squared_error' methods from the sklearn.metrics library and TensorFlow were used for the neural networks. The testing score was calculated using the coefficient of determination (R2), mean absolute error (MAE), mean square error (MSE), and root mean square error (RMSE) metrics to score the regression model trained with each algorithm.
Finally, the Matplotlib (Matplotlib, 2022) and Seaborn (Seaborn.org, 2012) libraries were used to generate the corresponding visualizations for each of the selected algorithms.

9.1. Multiple Linear Regression Algorithm Metrics

The score of the model trained by this algorithm gave a test score of 0.7821502808726812, calculated using the r2_score function. This score shows strong predictive performance on the test data, indicating that the model generalizes well to unseen data. In addition, the performance of different linear regression algorithms was evaluated using different key regression metrics. These were the mean absolute error (MAE), mean absolute percentage error (MAPE), and mean squared error (MSE). The results showed an overall MAE of 0.4276, MAPE of 0.2268, MSE of 0.2958, and a root mean square error (RMSE) of 0.5439. These values indicate that the model maintains the lowest prediction error, strengthening the suitability for the dataset used in this work. To graphically represent the similarity between the data used for testing and the data obtained from the prediction, the Matplotlib and Seaborn libraries were used to create a plot. The test prediction data are completely accurate with the test scores and line work shown in Figure 10.

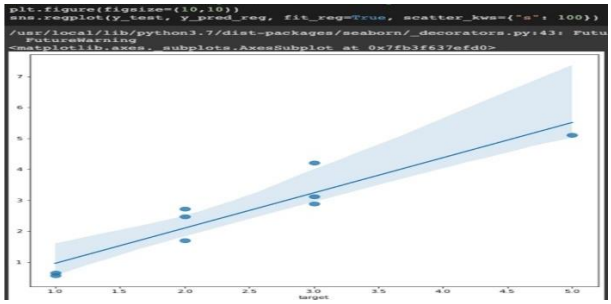

Fig. 10: Visualization of the Multiple Linear Regression Model Testing.

9.2. Support Vector Regression Algorithm Metrics

The evaluation of the model constructed using the support vector regression (SVR) algorithm yielded an output accuracy score of 0.8091766183956131, which is slightly higher than the score obtained by the previously evaluated model, indicating an improvement in the accuracy prediction performance. In addition, the regression metrics for the SVR model further support its effectiveness. The mean absolute error (MAE) was 0.4313, the mean absolute percentage error (MAPE) was 0.2301, the mean squared error (MSE) was 0.2591, and the root mean absolute error (RMSE) was 0.5091. These results show slight improvements in all metrics compared to the previous model, which confirms that the SVR algorithm showed better overall accuracy and lower prediction errors. Since the plot in Figure 11 shows a strong similarity between different linear regression algorithms, all metrics had to be directly compared to select the best algorithm among them. The Matplotlib and Seaborn libraries were used to create it; the projection is similar to the previous modeling algorithm.

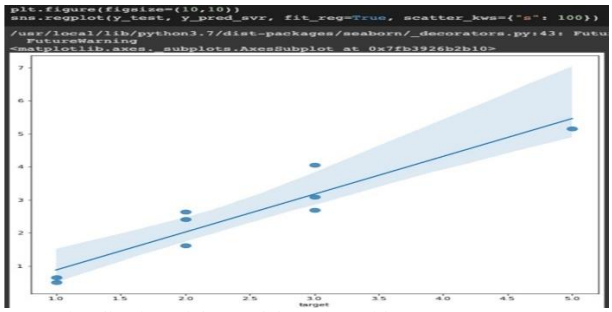

Fig. 11: Visualization of the Model Testing with Support Vector Regression.

9.3. Regression Decision Tree Algorithm Metrics

The model developed using the regression decision tree algorithm obtained a test score of 0.67, which is significantly lower than the scores of previously evaluated models, indicating poorer performance. The evaluation metrics further confirmed this observation. The model showed a mean absolute error (MAE) of 0.22, a mean absolute percentage error (MAPE) of 0.07, a mean squared error (MSE) of 0.44, and a mean absolute error of 0.66. Even when the MAE and MAPE values are low, the higher MSE and RMSE show a large difference in errors, indicating that the regression decision tree model did not perform as reliably as support vector regression and multiple linear regression models in predicting target values.
The graph in Figure 12 shows that the greater the number of predictions, the greater the margin of error, so its metrics are generally lower than the rest.

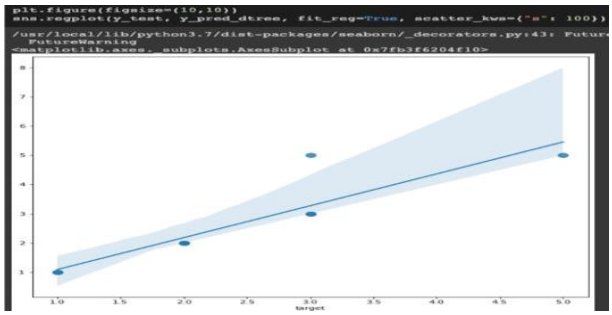

Fig. 12: Visualization of the Testing of the Model with Decision Trees.

9.4. Metrics of The Neural Network Algorithm

The evaluation of the model generated by the regression decision trees algorithm shows a Test Score of 0.7176519793867242, which is lower than the multiple linear regression and SVR models, but better than the regression decision tree algorithm.
Figure 13 shows the metrics derived from the neural network algorithm, which obtained an MAE of 0.5616, an MAPE of 0.3316, an MSE of 0.3834, and an RMSE of 0.6192. The graph in Figure 13 shows that this algorithm behaves in the same way as the various linear regression and SVR algorithms, but with the values further subtracted from the actual prediction.

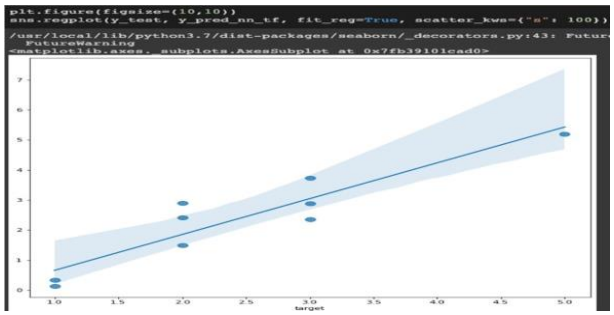

Fig. 13: Visualization of Model Testing with Neural Networks.

9.5. Analysis of Regression Metrics

As can be seen in Table 9, the regression metrics of the models generated from the selected algorithms had an optimal training score, and the testing score was also quite high, with the Multiple Linear Regression and SVR algorithms standing out. However, the latter performed better on average, and the parameters needed to train the model can be adapted according to the size of the data set. Because of this, the Support Vector Regression algorithm will be used to implement the predictive model.

Table 9: Comparative Table of Regression Metrics for Selected Algorithms

Metric	MLR	SVR	RDT	NN
Training Score	1.0000	0.9993	1.0000	0.5294
R^2 Score	0.7822	0.8092	0.6727	0.7177
MAE (Mean Absolute Error)	0.4276	0.4313	0.2222	0.5616
MAPE (Mean Absolute Percentage Error)	0.2268	0.2301	0.7408	0.3316
MSE (Mean Squared Error)	0.2958	0.2591	0.4444	0.3834
RMSE (Root Mean Squared Error)	0.5439	0.5091	0.6667	0.6192

Note: Values are rounded to four decimal places.

Based on the comparison metrics presented in Table 9, the Support Vector Regression (SVR) algorithm showed strong overall performance with a high R2 score (0.8092) and very low error rates in most evaluation metrics. The ability to model complex, nonlinear relationships while maintaining low forecast error makes “Union Distributors” an appropriate choice for a weekly earnings forecasting model. However, although SVR provides powerful prediction capabilities, several limitations should be noted when considering its application in real-world supply chains.
SVR is highly dependent on hyperparameter selection, such as kernel type, penalty parameter (C), and epsilon (ε). When there are outliers in the demand data – such as a sudden spike in purchases during promotional campaigns or panic buying during the COVID-19 pandemic – the model may overestimate these anomalies or become inappropriate if the epsilon threshold is too wide. In practice, this means that SVR predictions may be less reliable during periods of uncertain user behavior if strong preprocessing steps (such as outlier detection or smoothing) are not consistently applied.

Training SVR requires solving a quadratic optimization problem, which scales poorly with large datasets. For the data in this study (January 2020–April 2021), implementation was monitored; however, scaling up the model over the years, thousands of SKUs, or supermarket branches, increases computational costs dramatically. This may limit the feasibility of SVR in real-time prediction pipelines or in cases where frequent retraining is required.

Nonlinear kernels (such as radial basis function, RBF) help SVR capture complex patterns while simplifying interpretation, making it difficult for managers to predict outcomes by using input features, such as product life or proportional values. This "black box" behavior can hinder its use in business environments that require quantifiable forecasts to validate purchasing or inventory decisions. In real-world supermarket operations, SVR may fail due to (i) the introduction of new product categories without historical sales data, (ii) sudden market shocks such as supply disruptions or regulatory changes, and (iii) low-frequency data for low-frequency products. In such cases, the model may not be well-tuned and may produce predictions with large errors. In addition, since the current model does not include external supply chain variables such as supplier lead times or competitor pricing, SVR performs well in historical simulations, but there is a risk of not being able to generalize to a dynamic market.

The support vector regression (SVR) model showed superior performance over other regression algorithms tested in this study, especially in terms of accuracy and error reduction. Unlike hybrid machine learning approaches that combine multiple models to optimize supply chain and logistics services in complex sectors such as oil and gas (Odimarha et al. [10], the SVR model provides a streamlined but effective alternative. Its ability to capture nonlinear relationships without adding model coupling provides a significant advantage for medium-scale data such as supermarket demand forecasting. Similarly, Syed et al. [15], who use domain adversarial neural networks to improve forecast generality across different supply chain contexts, maintain the unique contribution of the SVR model by achieving robust forecasting performance even without the need for domain transfer algorithms. This positions the SVR as a practical, computationally efficient option in predictive supply architectures where interpretability and on-time results are equally important, in addition to forecast accuracy.

10. Study Development -Phase 6: Model Deployment

In this phase, a predictive model trained by the SVR algorithm was developed, which performed best with the training and validation datasets. The Mean Squared Error (MSE) metric was also used to select the best parameters for the model. Subsequently, the API was developed to generate the requested predictions using a module enabled in the graphical interface of the distribution company's inventory control system. The user with the "Manager" role can use it to forecast demand for each local supermarket.

The work was developed using Visual Studio Code, a free and customizable code editor from Microsoft. During setup, Python version 3.8.8 was identified, allowing the use of the virtualenv library to create a virtual environment tailored for the work. The main script, main.py, serves as the program's entry point. It reads data from the DatosDistribuidora.csv file, passes it to utils.py for conversion into a DataFrame, and proceeds to define features, target variables, and apply the Support Vector Regression (SVR) algorithm with optimized parameters for model training. The utils.py file includes helper functions for importing the validation dataset and exporting the trained model using the joblib library.

10.1. Optimization of Predictive Model Parameters

In Phase 4, the models were initially trained using standard parameter values to evaluate their baseline performance. However, to improve the Support Vector Regression (SVR) model and prevent issues like overfitting or underfitting, hyperparameter tuning was performed using GridSearchCV from the sklearn.model_selection module. This method systematically tests all possible parameter combinations to identify the most effective configuration for the dataset (Rodríguez, 2019; Müller & Guido, 2017). The optimal parameters found, improved—C = 1, epsilon = 0.3, gamma = 'auto', and kernel = 'linear'—resulted in a model accuracy of 85.6%.

10.2. Evaluation of The Predictive Model

Despite having a model with a high accuracy rate, it was necessary to evaluate its performance using data from the test set, which contains data not used to train the model. This ensures the integrity of the predictive model.

The cross-validation method was used to evaluate the model (Figure 14). This is a statistical method that helps evaluate generalization through more stable results than when using the train_test_split method (Müller & Guido, 2017).

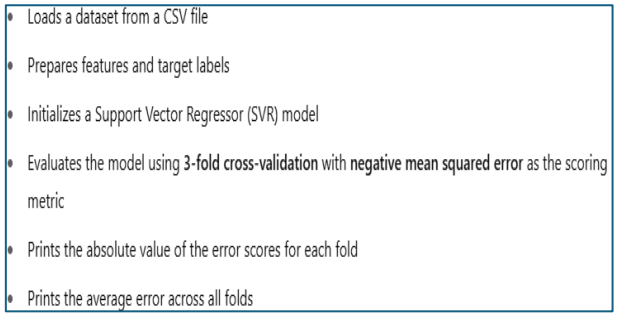

Fig. 14: Model Evaluation with Test Data Using Cross-Validation.

The model's accuracy with parameters optimized using cross-validation indicates an average success rate of 77.6%. Despite being lower than that obtained with the validation data, it demonstrates that the model is neither overfitting nor underfitting, as it has an acceptable accuracy level for export and use in the developed API.

10.3. API Development with The Predictive Model

The Python Flask framework was used for API development because it is simple, minimalist, and scalable. Additionally, most hosting providers work with WSGI, which facilitates the connection by returning one response at a time [3].

The inventory control system module that generates this report must be internally connected to the developed API. This API receives the data necessary to generate the prediction and returns the predictions for each product in JSON format to be displayed in a table, which would serve as a reference for store managers when generating purchase orders.

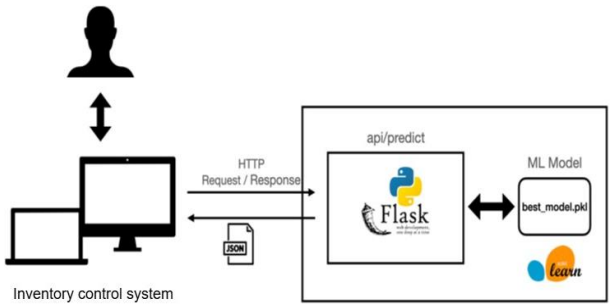

Fig. 15: Machine Learning Model Architecture.

As seen in Figure 15, the machine learning model architecture requires that the predictive model located in the 'models' folder, named 'best_model.pkl', be loaded at the beginning of the API execution.
To generate predictions, the system sends data to the /predict endpoint using the POST method. The server, built with the lightweight Flask framework, processes the incoming data and returns the prediction results in JSON format using the jsonify method. This setup allows the inventory control system to display suggested product quantities in a table. The model is loaded with the joblib library, and the input data—received as a list—is converted into a DataFrame to be compatible with the prediction model. The model then generates predictions for each local supermarket, returning them in a structured JSON response.

10.4. Predictive Module in The Inventory Control System

Once the predictive model was developed, it was necessary to connect it to the distribution company's inventory control system (Figure 16). This required adding a module within the user interface that would allow requests to be sent and values to be received in JSON format. In this module, it was necessary to import a .csv file with product demand data and also select the local supermarket for which the quantities suggested by the predictive model were required.

Fig. 16: Inventory Control System with the Predictive Module.

11. Results

This section shows a comparison of the initial state of the indicators and metrics used by the distribution company and how the values changed after the implementation of the predictive model. The indicators used were the following: monthly return per supermarket, monthly return by category for each supermarket, and monthly profit per supermarket.

11.1. Monthly Return Per Supermarket

The objective of this indicator is to obtain the percentage of overall monthly return volume. This indicator is calculated monthly and is calculated by the store manager, who sends a report to the company manager as part of their logistics duties.
The formula for calculating this indicator is shown in Equation 4:

$$\text{Monthly Return per Supermarket} = \left(\frac{\text{Total Returned products}}{\text{Total Distributed products}}\right) \times 100 \quad (4)$$

Where the 'Total returned products' is calculated by adding the quantities of all products returned due to expiration, the 'Total distributed products' is calculated by adding the quantities of all products distributed to local supermarkets, the 'Monthly return per supermarket' is calculated as a percentage, and the result is obtained by dividing the 'Total returned products' by the 'Total distributed products' and then multiplying by 100.

11.2. Monthly Returns by Category

This indicator is like the previous one; however, it is more specific because its objective is to obtain the percentage of monthly returns for each product category for each local supermarket. This indicator is calculated monthly, and the manager is responsible for calculating it. This indicator requires obtaining the monthly returns report from the store managers.
The formula for calculating this indicator is shown in Equation 5:

$$\text{Monthly Return by Category} = \sum \text{Monthly quantity of products from the same category} \quad (5)$$

Where the 'monthly return by category' is calculated for each of the 12 categories, the indicator is calculated by summing the monthly amounts of products returned in the same category by each local supermarket; this means that the total value of products in each category will be obtained.

11.3. Monthly Gross Profit Percentage

This indicator aims to calculate the monthly net profit for each supermarket. The frequency of this indicator is monthly, and the manager is responsible for performing the calculations to obtain the percentage value of this indicator. For this indicator, it is necessary to obtain monthly supermarket sales reports, supermarket return reports, and operating expenses. It should be noted that this indicator does not consider other operating expenses other than gross profit.
The first formula for calculating this indicator is shown in Equation 6, which first calculates the monthly gross profit per supermarket, and the result is used in Equation 7.

$$\text{Monthly gross profit per supermarket} = \text{Total revenue} - \text{Total costs} \quad (6)$$

Where 'Total revenue' is obtained from the accounting staff, 'Total costs' is obtained by adding all expenses incurred to obtain 'Union Distributors' products plus the monetary loss generated by product returns, 'Monthly gross profit per supermarket' is calculated by subtracting 'Total revenue' from 'Total costs'. Equation 7 is shown below:

$$\text{Gross Profit Percentage} = \left(\frac{\text{Monthlygross profit per supermarket}}{\text{Total monthly revenue}}\right) \times 100 \quad (7)$$

Where the 'Monthly gross profit per supermarket' is the result of the previous equation, the 'Total monthly revenue' is calculated by the distribution company's accounting staff. To find the 'Monthly profit per supermarket', divide the 'Monthly gross profit per supermarket' by the 'Total monthly revenue' and multiply by 100 to calculate the percentage.

11.4. Baseline - Monthly Return Per Supermarket Without A Predictive Model

The formula for this indicator was calculated for each of the local supermarkets to which the products are distributed. Figure 17 shows the result for the last 6 months without using the predictive model and an average semi-annual return per supermarket with a value of 11.05%, as shown in Figure 17.

11.5. Baseline - Monthly Return by Category without A Predictive Model

The formula for this indicator was calculated for each of the product categories, adding the totals for all the local supermarkets to which the products are distributed. Figure 18 shows the result for the last 6 months without using the predictive model. The sum of the semiannual return by category had a total value of 21,218 units, as shown in Figure 18.

11.6. Baseline - Monthly Percentage Gross Profit Margin without A Predictive Model

The percentage gross profit margin for the last 6 months without using the predictive model averaged 15.78%, as can be seen in Figure 19.

11.7. Status

After the implementation of the predictive model, notable improvements were observed. Firstly, the average monthly return per supermarket over six months decreased, resulting in a semi-annual average return of 7.03%, indicating better stock planning. Secondly, monthly returns by product category also declined, totaling 12,786 units in semiannual returns, reflecting improved inventory efficiency. Lastly, the average gross profit margin over the six months increased to 19.87%, demonstrating enhanced profitability under the predictive model. As seen in Table 10, the gross profit margin percentage for the last 6 months using the predictive model had a better average than the baseline, with a value of 19.87%.

Table 10: Comparative Table of Gross Profit Margin Percentage – with Predictive Model

Month 1	Month 1	Month 1	Month 1	Month 1	Month 1	Semiannual Average
19.88%	19.96%	19.99%	19.68%	19.76%	19.95%	19.87%

12. Analysis and Discussion

12.1. Analysis of Monthly Returns Per Supermarket

As seen in the current state results, after applying the formulas for each metric, we can conclude that there was an average optimization of 36.36% over the monthly return metric data per supermarket from the baseline.
Figure 17 shows the average return percentage per supermarket when the model was not used in red, and how the predictive model optimizes the average return percentage in blue.
Figure 17 shows the average return percentages per supermarket without a predictive model in red, compared to the percentages with the predictive model in blue. The percentages decreased after using the predictive model.

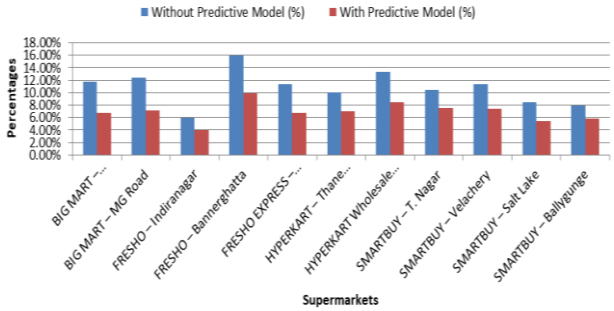

Fig. 17: Average Percentage Return Comparison by Supermarket.

12.2. Analysis of Monthly Returns by Category

The results show that the return percentage by product category has undergone some changes. Of the total returned products, the "bread" category had a much higher percentage than the rest of the categories. This is because the expiration margin for this category is much lower than for the other categories. Table 1 shows the expiration margin by category. Furthermore, the "bread" category sells in larger quantities, requiring a high turnover of merchandise, making it the most influential category in the monthly return volume per supermarket.
The predictive model has also helped to better balance these percentages, as 65.2% of the baseline results decreased to 55.6% in the current state results.

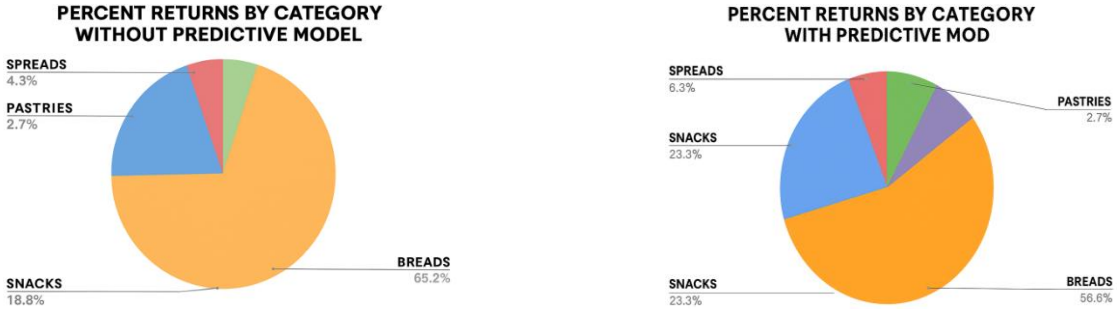

Fig. 18: Comparison of Return Percentage by Category.

Additionally, the comparison of category quantities shows a significant improvement in product returns; the "bread" category is the most clearly visible optimization, with a difference of 7,125 units from the semiannual average; that is, a 50.06% decrease from the total obtained before using the predictive model, as seen in Figure 18.
The other categories have also seen a significant reduction in the monthly return metrics by category, resulting in an overall optimization of this indicator, as the total returns per category decreased from 21,218 units to 12,786 units. That is, a difference of 8,432 units, representing a percentage difference of 39.74%.
As shown in Figure 19, the monthly return by category in quantities without a predictive model is shown in red and with the predictive model in blue.
This indicates that all categories have seen a decrease in returns, and the decrease in returns for the bread category is also very noticeable, thus validating the optimization of supply chain management with this indicator.

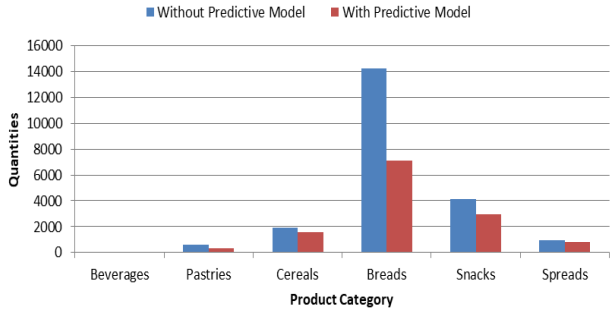

Fig. 19: Comparison of Monthly Returns by Category in Quantities.

12.3. Analysis of Monthly Gross Profit Percentage

The monthly gross profit percentage has also been optimized, with a better semi-annual average compared to that obtained before the predictive model was implemented, with an average value of 4.09%.
A better semi-annual average was obtained compared to the baseline gross profit percentage. The percentage difference indicates that there was a 25.95% optimization of the gross profit percentage after the implementation of the predictive model. Figure 20 shows the comparison between the monthly averages of gross profit percentage without the predictive model, which is colored red, and the results after implementing the predictive model, which is colored blue. There is an increase in all months due to the decrease in returns. That is, total costs decreased due to less expense generated by product returns, thus validating the optimization of supply chain management at the distribution company.

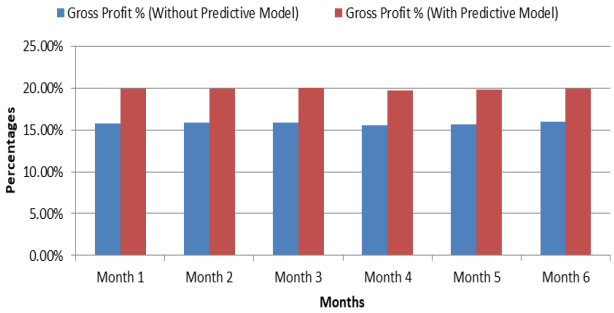

Fig. 20: Comparison of Average Gross Profit Percentages by Month.

13. Conclusions and Recommendations

The research work focused on optimizing a distribution company's supply chain through predictive modeling. A thorough understanding of the company's operations—including its internal policies, product categories, customer base, procurement processes, and inventory system—was established. Based on a review of relevant literature, the most used regression algorithms in supply chain optimization were identified: multiple linear regression, support vector regression (SVR), regression decision trees, and neural networks. The dataset was cleaned, normalized, and split into training (80%) and testing (20%) sets, and models were trained using libraries like Scikit-learn and TensorFlow. Evaluation metrics such as R^2, MAE, MAPE, MSE, and RMSE revealed that the SVR model performed best. After implementation, significant improvements were observed: returns per supermarket decreased by 36.36%, returns per category by 39.74%, and gross profit increased by 25.95% compared to the previous half-year, confirming the effectiveness of the model. Recommendations include fostering a data-driven culture, exploring new regression techniques, adding historical data on new products, refining predictive models for greater accuracy, improving evaluation metrics, and consistently tracking performance indicators to ensure continuous improvement in supply chain processes.

Future research could expand this work by adding real-time data, such as smart racks and RFID tags, to improve prediction responsiveness and accuracy. Another direction is to improve SVR when introducing new products with limited historical sales records. Exploring hybrid or ensemble models that combine SVR with techniques such as gradient boosting or neural networks could further improve prediction performance in highly dynamic supply chains. Furthermore, the use of domain adaptation methods allows models trained in one supermarket or region to be transferred to another with minimal retraining. Finally, future research could incorporate descriptive frameworks such as SHAP or LIME to make SVR predictions more transparent and actionable for decision makers, while expanding prediction goals to include sustainability metrics such as waste reduction and carbon efficiency.

References

[1] Nimmagadda, V. S. P. (2020). AI-powered predictive analytics for retail supply chain risk management: Advanced techniques, applications, and real-world case studies. *Distributed Learning and Broad Applications in Science Research*, 6, 152–194.

[2] Pal, S. (2023). Integrating AI in sustainable supply chain management: A new paradigm for enhanced transparency and sustainability. *International Journal of Research in Applied Science and Engineering Technology*, 11, 2979–2984. https://doi.org/10.22214/ijraset.2023.54139.

[3] Charles, V., Emrouznejad, A., & Gherman, T. (2023). A critical analysis of the integration of blockchain and artificial intelligence for supply chain. *Annals of Operations Research*, 327, 7–47. https://doi.org/10.1007/s10479-023-05169-w.

[4] Curcio, D., & Longo, F. (2009). Inventory and internal logistics management as critical factors affecting the supply chain performances. *International Journal of Simulation and Process Modelling*, 5, 278–288. https://doi.org/10.1504/IJSPM.2009.032591.

[5] Sharma, N., & Singhi, R. (2018). Logistics and supply chain management quality improvement of supply chain process through vendor managed inventory: A QFD approach. *Journal of Supply Chain Management Systems*, 7, 23–33.

[6] Mahraz, M. I., Benabbou, L., & Berrado, A. (2022). Machine learning in supply chain management: A systematic literature review. *International Journal of Supply and Operations Management*, 9, 398–416.

[7] Abolghasemi, M., Beh, E., Tarr, G., & Gerlach, R. (2020). Demand forecasting in supply chain: The impact of demand volatility in the presence of promotion. *Computers & Industrial Engineering*, 142, 106380. https://doi.org/10.1016/j.cie.2020.106380.

[8] Salamai, A. A., El-Kenawy, E. S. M., & Abdelhameed, I. (2021). Dynamic voting classifier for risk identification in supply chain 4.0. *Computers, Materials & Continua*, 69(3). https://doi.org/10.32604/cmc.2021.018179.

[9] Suwignjo, P., Panjaitan, L., Baihaqy, A., & Rusdiansyah, A. (2023). Predictive analytics to improve inventory performance: A case study of an FMCG company. *Operations and Supply Chain Management: An International Journal*, 16, 293–310. https://doi.org/10.31387/oscm0530390.

[10] Odimarha, A. C., Ayodeji, S. A., & Abaku, E. A. (2024). Machine learning's influence on supply chain and logistics optimization in the oil and gas sector: A comprehensive analysis. *Computer Science and IT Research Journal*, 5, 725–740. https://doi.org/10.51594/csitrj.v5i3.976.

[11] Pasupuleti, V., Thuraka, B., Kodete, C. S., & Malisetty, S. (2024). Enhancing supply chain agility and sustainability through machine learning: Optimization techniques for logistics and inventory management. *Logistics*, 8(3), 73. https://doi.org/10.3390/logistics8030073.

[12] Akbari, M., & Do, T. N. A. (2021). A systematic review of machine learning in logistics and supply chain management: Current trends and future directions. *Benchmarking: An International Journal*, 28, 2977–3005. https://doi.org/10.1108/BIJ-10-2020-0514.

[13] Yang, M., Lim, M. K., Qu, Y., Ni, D., & Xiao, Z. (2023). Supply chain risk management with machine learning technology: A literature review and future research directions. *Computers & Industrial Engineering*, 175, 108859. https://doi.org/10.1016/j.cie.2022.108859.

[14] Daios, A., Kladovasilakis, N., Kelemis, A., & Kostavelis, I. (2025). AI applications in supply chain management: A survey. *Applied Sciences*, 15(5), 2775. https://doi.org/10.3390/app15052775.

[15] Sayyad, J., Attarde, K., & Yilmaz, B. (2024). Improving machine learning predictive capacity for supply chain optimization through domain adversarial neural networks. *Big Data and Cognitive Computing*, 8(8), 81. https://doi.org/10.3390/bdcc8080081.

[16] Dey, P. K., Chowdhury, S., Abadie, A., Vann Yaroson, E., & Sarkar, S. (2024). Artificial intelligence-driven supply chain resilience in Vietnamese manufacturing small-and medium-sized enterprises. *International Journal of Production Research*, 62, 5417–5456. https://doi.org/10.1080/00207543.2023.2179859.