algorithms

MDPI

Article

Solving a Multi-Period Dynamic Pricing Problem Using Learning-Augmented Exact Methods and Learnheuristics

Angel A. Juan [1,2,*], Yangchongyi Men [1], Veronica Medina [1,3] and Marc Escoto [1]

[1] CIGIP—ValgrAI, Universitat Politècnica de València, Plaza Ferrandiz-Carbonell, 03801 Alcoy, Spain; myangch@upv.es (Y.M.); vmedrid@upv.edu.es (V.M.); mescgom@upv.es (M.E.)
[2] Euncet Business School, Universitat Politècnica de Catalunya, Mas Rubial 1, 08225 Terrassa, Spain
[3] ICSO Analytics SL, Avinguda Jacquard 87, Planta Entrante, 08222 Terrassa, Spain
* Correspondence: ajuanp@upv.es

Abstract

This paper addresses a dynamic multi-period pricing problem that incorporates time-varying contextual information and inventory constraints. Sales are modeled as a function of both price and a multidimensional context vector, which may include factors such as customer location, income, loyalty, competitor prices, and promotional activity. This formulation captures complex market dynamics over a finite selling horizon. The problem is formulated as a quadratic programming model, and two alternative solution approaches are proposed. The first uses a multivariate regression model to approximate the sales function linearly, allowing an exact quadratic programming solution that serves as a benchmark. The second is a 'learnheuristic' algorithm that combines a nonlinear sales learning model with metaheuristic optimization to generate high-quality pricing strategies under realistic operational constraints. Computational experiments demonstrate the effectiveness of the proposed learnheuristic approach.

Keywords: dynamic pricing; context-aware; multi-period; machine learning; metaheuristics

1. Introduction

Dynamic pricing refers to the strategy of adjusting prices in real time based on market conditions, customer behavior, and operational constraints [1]. It plays a crucial role in revenue management for various industries, including e-commerce, transportation, retail, and hospitality. Despite its widespread use, dynamic pricing remains a challenging problem due to several factors: sales uncertainty, market volatility, heterogeneous customer preferences, and practical limitations such as inventory or pricing rules. These challenges are compounded when prices need to be set over multiple time periods, where decisions at one time affect future outcomes [2]. A critical aspect often overlooked or simplified in many dynamic pricing models is the role of context information and inventory constraints [3]. Real-world sales is rarely driven by price alone. Instead, it depends on a wide range of external and dynamic factors, such as economic indicators, competitor actions, seasonality, or product attributes. Representing this information explicitly allows for a more realistic and flexible modeling of sales. Incorporating such multi-dimensional, time-varying context allows for a better modeling of the complex interactions that influence purchasing decisions. This paper addresses some of these challenges by formulating a dynamic multi-period pricing problem as a context-aware optimization task under inventory and pricing constraints. We propose a two-tiered approach that balances theoretical rigor and practical applicability.

Check for updates

Academic Editor: Grigorios Beligiannis

Received: 24 February 2026
Revised: 24 March 2026
Accepted: 2 April 2026
Published: 7 April 2026

Copyright: © 2026 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license.

First, we employ a statistical regression model to derive a linear approximation of the sales function based on price and context information. This approximation allows the problem to be cast as a mixed-integer quadratic program (MIQP), or even a quadratic program (QP) if prices are not forced to be integer. These programs can be solved exactly with modern optimization solvers. This step provides a solid benchmark and ensures that context and multi-period dependencies are rigorously accounted for. However, real-world sales relationships are often nonlinear and complex, and linear approximations may fall short in capturing these dynamics fully [4]. To overcome this, we introduce a novel 'learnheuristic' algorithm that integrates a nonlinear machine learning model with metaheuristic optimization. The term learnheuristic refers to a class of hybrid methods that integrate machine learning within metaheuristic frameworks to handle context-dependent or dynamically changing elements of an optimization problem [5]. In our approach, the machine learning model predicts richer sales patterns from historical data and context features, while the metaheuristic algorithm efficiently explores the solution space to produce high-quality pricing policies under realistic constraints [6]. This hybrid approach enables scalable and adaptive dynamic pricing solutions beyond what exact methods can handle, particularly in large markets or markets with non-linear sales relationships.

The main contributions of our work are described next: (i) it formalizes dynamic pricing as a context-sensitive, multi-period optimization problem, where demand is modeled via a linear regression that explicitly incorporates individual customer features and external factors while simultaneously enforcing inventory and price constraints; (ii) it solves the aforementioned problem using an exact solver (Gurobi) that relies on a linear predictive model; (iii) it proposes a learnheuristic algorithm that combines an XGBoost non-linear predictive model with a particle swarm optimization (PSO) metaheuristic to solve the problem; and (iv) it analyzes and compares the results provided by each approach. The results show that our learnheuristic approach provides a flexible framework that can accommodate context-dependent demand models and multi-period pricing, as illustrated on synthetic instances. The rest of the paper is structured as follows. Section 2 reviews related work on context-aware and multi-period dynamic pricing. Section 3 presents the formal problem statement and explains how it can be solved using an exact solver. Section 4 describes the process used to generate synthetic data for our numerical experiments. Section 5 explains how the linear and nonlinear machine learning models are developed with the goal of estimating sales. Section 6 introduces the learnheuristic algorithm for nonlinear models. An analysis of the results obtained in the computational experiments is provided in Section 7. Finally, Section 8 concludes with key findings and suggestions for future research.

2. Related Work on Context-Aware and Multi-Period Dynamic Pricing

Two relevant streams to our work are contextual dynamic pricing, which incorporates customer or market features into demand learning, and multi-period dynamic pricing, which integrates pricing decisions over time under operational constraints such as inventory or perishability. The following review summarizes key contributions in both areas and highlights how our work connects them.

Contextual dynamic pricing has emerged as a key area in revenue management, blending demand learning with operational constraints across time. Luo et al. [7] introduce a distribution-free approach. They assume customer valuation follows a linear function of context plus unknown noise. Their policy simultaneously learns both the context–demand relation and the noise distribution using a perturbed linear bandit framework. They prove matching regret bounds and validate performance on simulated and real auto-loan data, achieving robustness when noise is unknown. Bu et al. [8] explore semi-parametric demand models in a context-based pricing setting. They consider two frameworks: one additive in

price and context, the other combining a smooth price function and linear context. Their online algorithms achieve minimax-optimal regret rates, and they derive explicit bounds that clarify how model smoothness and context dimension drive learning complexity. Liu et al. [9] study dynamic pricing when buyers can strategically manipulate their context data. In a personalized pricing setting under strategic behavior, these authors design a policy that estimates both buyer valuation and manipulation cost online. They show that ignoring strategic manipulation leads to linear regret, whereas their algorithm achieves optimal regret under the setting. Wang et al. [10] address the often-neglected topic of uncertainty quantification in contextual dynamic pricing. They develop a debiased estimator for the demand function under sequential pricing data, enabling valid pointwise and uniform confidence intervals. This provides operationally valuable uncertainty estimates for decision-making under evolving demand. Miao et al. [11] address context-based dynamic pricing by combining online clustering with demand estimation. They propose an adaptive algorithm that groups similar customer contexts to improve price recommendations without prior demand models. Their method updates clusters and pricing decisions in real time, achieving performance guarantees in regret bounds. Numerical experiments illustrate its advantages in highly heterogeneous markets. Chen et al. [12] focus on fairness in contextual dynamic pricing, introducing the concept of 'utility fairness' where price decisions aim to balance revenue and fairness across customer segments. They integrate demand learning with fairness constraints into an online pricing framework. The model is shown to reduce disparity in utility among customer groups while maintaining competitive revenue performance. Nambiar et al. [13] investigate dynamic pricing when the assumed demand model is misspecified. They develop a learning framework that updates both the pricing policy and the demand model based on observed data, mitigating the adverse effects of initial misspecifications. Their results show that adaptive model correction significantly improves revenue compared to static or purely exploitative strategies.

The multi-period dynamic pricing literature addresses pricing decisions over time, often accounting for operational constraints such as inventory or perishability. Ullah et al. [14] study dynamic pricing decisions in a multi-period newsvendor framework where demand is stochastic and price-dependent. They develop an analytical model to jointly determine optimal prices and order quantities over multiple periods. The work examines how demand uncertainty and inventory replenishment interact, offering closed-form solutions in specific cases. Their findings highlight how price elasticity and remaining inventory shape optimal pricing paths across the horizon. Wang et al. [15] propose a multimodal dynamic pricing framework where demand can shift between different regimes, each with distinct price sensitivities. Using a stochastic model and machine learning techniques, they develop policies that adapt to observed demand modes over time. The approach balances exploration of unknown price–demand relationships with the exploitation of current demand knowledge, showing strong performance in both synthetic and real datasets. Chung et al. [16] analyze dynamic pricing in loyalty programs where customers can redeem points for purchases. They model the interaction between cash pricing and point redemption as a dynamic optimization problem. The solution accounts for customer redemption behavior, showing how optimal prices depend on point balances and redemption rules. Mumbower [17] review recent research on dynamic pricing of airline ancillary services, such as baggage fees and seat upgrades. The paper synthesizes offer management and revenue optimization literature, identifying emerging trends in personalized ancillary pricing and integration with core ticket pricing strategies. The review highlights the gap in combining ancillary and ticket pricing in a unified dynamic framework. Hu et al. [18] study dynamic pricing for perishable products under consumer preferences for low-carbon options. They develop a model where prices and product lifetimes jointly influence demand from environmentally

conscious consumers. The results show that incorporating low-carbon preferences alters optimal pricing trajectories and can increase both revenue and sustainability outcomes. Liang et al. [19] propose a distributed dynamic pricing strategy for presale mechanisms using deep reinforcement learning. Their approach coordinates multiple sellers to optimize prices while learning demand patterns in a decentralized way. The method improves revenue in environments where sellers face partial information and competition. Cho and Papavasiliou [20] examine pricing strategies under uncertainty in multi-interval real-time electricity markets. They develop a stochastic optimization model that accounts for demand fluctuations, generation constraints, and market rules. Their results provide insights into how uncertainty in supply and demand affects optimal pricing schedules. Chen et al. [21] address multi-period pricing and ordering for fresh produce under option contracts. They model decisions on preliminary and option orders alongside retail prices, considering perishability and price-dependent demand. The analysis shows how price differences between regular and concessionary products affect order allocation across periods.

Several recent studies further advance contextual dynamic pricing by addressing multi-product, multi-period, and adversarial demand settings. Perivier and Goyal [22] consider dynamic pricing and assortment under a contextual multinomial logit model, proposing an online Newton step algorithm that achieves near-optimal regret with adversarial context arrivals. Their work blends randomized pricing policies with contextual bandit methods and highlights the role of self-concordance in improving regret bounds. Hua et al. [23] study multi-period markdown pricing for perishable products in e-commerce fresh retail. They combine a semi-parametric structural demand model with counterfactual prediction to capture price elasticity and demand uncertainty. Their two-stage Markov decision process approach efficiently optimizes pricing policies over a finite horizon and demonstrates effectiveness in a real-world setting. Tullii et al. [24] propose an algorithm for contextual dynamic pricing that adapts between valuation approximation and price optimization under both linear and non-parametric valuation models. Their approach achieves order-optimal regret bounds in the linear case and improves results for smooth non-linear valuations. This work illustrates the benefits of integrating machine learning with adaptive pricing in high-dimensional context spaces.

While recent studies combine contextual sales learning with multi-period dynamic pricing, they do not explicitly address operational constraints such as inventory or perishability within this joint framework. Our work fills this gap by proposing a PSO-based learnheuristic approach that integrates context-aware sales volume learning with operationally realistic pricing decisions under inventory constraints in a multi-period setting. PSO-type algorithms have been widely studied for complex, high-dimensional, and constrained optimization problems. For example, Zhu et al. [25] combine multiple linear regression and PSO to optimize pricing and replenishment of short-shelf-life vegetables. More broadly, recent advances in PSO and its successful application to complex, non-linear and constrained optimization problems such as hybrid renewable energy system design [26] and heterogeneous comprehensive learning PSO variants for real-world engineering optimization problems [27] demonstrate the algorithm's suitability for handling operationally constrained decision spaces, which motivates its adoption in dynamic pricing settings.

3. Problem Modeling

Consider a company setting prices for a product over a finite time horizon with n periods, $t \in H = \{1, 2, \ldots, n\}$. At each period $t \in H$, the firm selects a discrete positive price, $p_t \in \mathbb{Z}^+$, from a set of integer price levels, $p_t \in \{p_{min}, p_{min} + 1, \ldots, p_{max}\}$, with $p_{max} > p_{min} > 0$. For a given period $t \in H$, the sales quantity in t, $s_t \in \mathbb{N}$ is a non-negative integer that depends on the chosen price, p_t, and on an m-dimensional con-

text vector for that period, $X_t = (x_{1t}, x_{2t}, \ldots, x_{mt})$, i.e., $s_t = f(p_t, X_t)$, $\forall t \in H$, where $f(\cdot)$ denotes a generic sales function. It is assumed that $\forall t \in H$, X_t is known at $t = 1$. The firm has a positive fixed inventory $q_{max} \in \mathbb{Z}^+$ for the entire horizon, so $\sum_{t \in H} s_t \leq q_{max}$. The goal is to choose the price p_t in each period t to maximize total revenue, $r = \sum_{t \in H} r_t = \sum_{t \in H} p_t s_t$, subject to inventory and price constraints. The corresponding mathematical programming model can be formulated as follows:

$$\text{maximize} \quad r = \sum_{t \in H} r_t = \sum_{t \in H} p_t s_t \tag{1}$$

$$\text{subject to} \quad s_t = f\left(p_t, X_t\right) \qquad \forall t \in H \tag{2}$$

$$\sum_{t \in H} s_t \leq q_{max} \tag{3}$$

$$p_{min} \leq p_t \leq p_{max} \qquad \forall t \in H \tag{4}$$

$$p_t \in \mathbb{Z}^+ \qquad \forall t \in H \tag{5}$$

The objective function in Equation (1) refers to the maximization of total revenue. Constraints (2) specify that the sales quantity for each period $t \in H$ is a function of price and context. Constraint (3) enforces that total sales do not exceed inventory. Constraints (4) impose price limits per period, and constraints (5) define variables domain. In general, the model is a mixed-integer nonlinear program because $f(\cdot)$, and consequently, the product $p_t s_t$ may be nonlinear.

To allow the model to be handled by an exact solver, the sales function $f(\cdot)$ can be approximated by a linear function of the price p_t and the context vector X_t, as indicated in Equation (6).

$$s_t = f(p_t, X_t) \approx \alpha + \sum_{j=1}^{m} \beta_j x_{jt} + \gamma p_t \quad \forall t \in H \tag{6}$$

where $X_t = (x_{1t}, x_{2t}, \ldots, x_{mt})$ are known $\forall t \in H$ and α, β_j, and γ are constants estimated from the data to capture observed sales patterns. Substituting this linear approximation into the revenue expression yields Equation (7).

$$r = \sum_{t \in H} p_t s_t \approx \sum_{t \in H} p_t \left(\alpha + \sum_{j=1}^{m} \beta_j x_{jt} + \gamma p_t \right) = \sum_{t \in H} \left(\alpha p_t + \sum_{j=1}^{m} \beta_j x_{jt} p_t + \gamma p_t^2 \right) \tag{7}$$

Since X_t and the coefficients α, β_j, γ are known constants, the revenue is a quadratic function in the decision variables p_t. If $\gamma < 0$, this quadratic function is concave, and the resulting maximization problem is a convex quadratic program. The full formulation is given in Equations (8)–(12):

$$\text{maximize} \quad \hat{r} = \sum_{t \in H} \left(\alpha p_t + \sum_{j=1}^{m} \beta_j x_{jt} p_t + \gamma p_t^2 \right) \tag{8}$$

$$\text{subject to} \quad \hat{s}_t = \alpha + \sum_{j=1}^{m} \beta_j x_{jt} + \gamma p_t \quad \forall t \in H \tag{9}$$

$$\sum_{t \in H} \hat{s}_t \leq q_{max} \tag{10}$$

$$p_{min} \leq p_t \leq p_{max} \quad \forall t \in H \tag{11}$$

$$p_t \in \mathbb{Z}^+ \quad \forall t \in H \tag{12}$$

where $\hat{r}$ and $\hat{s}_t$ are estimates of the original variables and s_t can be considered auxiliary variables linked to p_t. This new formulation is MIQP due to the quadratic term γp_t^2 in the objective and the integrality constraints on p_t. This formulation can be solved using MIQP solvers, such as Gurobi (https://www.gurobi.com, accessed on 3 April 2026) or CPLEX (https://www.ibm.com/products/ilog-cplex-optimization-studio, accessed on 23 March 2026), which handle quadratic objectives with integer variables.

Finally, if the price variables p_t are allowed to be continuous positive real numbers, i.e., $p_t \in \mathbb{R}^+$ with $0 < p_{\min} \leq p_t \leq p_{\max}$, the problem simplifies by removing the integrality constraints on prices. With this relaxation, we reduce the problem from an MIQP to a QP, which is generally easier to solve. The quadratic objective remains due to the term γp_t^2, but continuous variables allow for more efficient convex optimization methods. In the following, we will relax this integrality constraint by using continuous prices, which is suitable for a wide range of pricing problems that benefit from fine-grained optimization. The computed sales quantities were rounded to integers to determine the final revenue. Notice, however, that integer prices may be required to fit payment systems or operational constraints, especially in scenarios where payment by debit card is not always possible, such as public transit fares, vending machines, and similar cases.

4. Synthetic Data Generation

A synthetic dataset of 20,000 observations was generated to support the development and testing of machine learning models for estimating the sales function $f(\cdot)$ and for conducting computational experiments on the dynamic pricing problem. Each observation represents a sales event registered at a time period $t \in H$ with associated price (p_t), contextual variables (X_t), and sales quantity (s_t). As shown in Table 1, the dataset contains the variables p_t, customer zip code, customer $income$ (monthly household income), customer $loyalty$ status (binary indicator of customer loyalty program membership), competitor price at period t (cp_t), promotion flag at period t ($promo_t$, binary indicator), and s_t (integer). These variables were selected to capture key drivers of sales, including product price, spatial heterogeneity, socioeconomic characteristics, competitive conditions, and promotional effects while keeping the dataset compact enough for controlled experimentation.

Table 1. Sample of the synthetic dataset.

p_t	zip	$income$	$loyalty$	cp_t	$promo_t$	s_t
12.49	22	5700.70	0	9.72	0	42
24.01	2	5169.20	1	9.24	0	0
19.64	4	4469.30	1	12.26	0	0
16.97	13	4434.04	0	6.48	0	0
8.12	13	3745.73	0	16.21	0	63
⋮	⋮	⋮	⋮	⋮	⋮	⋮
22.54	15	5566.11	0	21.98	0	0
5.94	7	3840.37	1	16.94	0	84
11.07	11	3815.50	0	14.67	0	30
13.87	17	7028.86	0	6.74	1	258
8.45	5	4393.25	1	19.74	0	68

The synthetic dataset was generated to model the sales quantity (s_t) for a product as a nonlinear function of price (p_t) and context variables (X_t). The goal was to create data where a linear model achieves moderate predictive power while a nonlinear model might be able to capture the complexity more accurately, thus reflecting realistic sales behavior. The price variable at period t (p_t) is generated as a continuous variable uniformly distributed between

5 and 25, representing the sales price offered by the firm in each observation. The *zip* variable is a discrete integer randomly sampled from the set $\{1, 2, \ldots, 30\}$, simulating customer location across 30 geographic areas. Customer *income* was simulated using a lognormal distribution with $\mu = 8.474104$ and $\sigma = 0.293560$ (i.e., having an approximated mean of 5000 and standard deviation of 1500). This choice ensures all income values are positive while preserving the intended average and natural variability. It also captures the realistic right-skewness observed in income data. The *loyalty* variable is binary, sampled from a Bernoulli distribution with success probability 0.3, indicating whether a customer is loyal to the firm. The cp_t variable, representing competitor price at period t, is also uniformly sampled between 5 and 25, capturing competition pricing dynamics. Likewise, $promo_t$ is a binary indicator sampled from a Bernoulli distribution with probability 0.2, reflecting whether a promotional campaign is active at period t.

Finally, s_t is generated as a nonlinear function of the aforementioned predictors, capturing complex interactions and quadratic effects. Firstly, an auxiliary variable s'_t is generated as described in Equation (13); i.e., a baseline intercept is adjusted by negative linear and quadratic terms in p_t, a positive linear effect of *income*, and additive effects of $loyalty$, cp_t, and $promo_t$. Interaction terms between p_t and *income*, p_t and $loyalty$, as well as *income* and $promo_t$, introduce nonlinear dependencies. Gaussian noise ε_t, with mean 0 and standard deviation 1, is added to reflect random variation:

$$\begin{aligned} s'_t = {} & 100 - 1.2\ p_t + 0.00035\ income + 3.0\ loyalty \\ & - 0.7\ cp_t + 2.0\ promo_t - 0.8\ p_t^2 \\ & + 0.0012\ p_t \cdot income - 0.025\ p_t \cdot loyalty \\ & + 0.03\ income \cdot promo_t + \varepsilon_t \end{aligned} \tag{13}$$

Secondly, to ensure sales quantity remains strictly positive, a softplus transformation [28] is applied, as described in Equation (14):

$$s_t = \text{round}\left(\log\left(1 + e^{s'_t}\right)\right) \tag{14}$$

This transformation produces smoothly varying non-negative sales values suitable for modeling and prediction, ensuring numerical stability while preserving realistic zero-sales outcomes.

5. Machine Learning Models

Using Python 3.12.2, the synthetic dataset obtained in Section 4 was first divided into a training set (used for model estimation), and a test set (used for out-of-sample evaluation of predictive performance). The sales quantity variable, s_t, was modeled as a function of the selling price p_t and the selected context variables, with *zip* treated as a categorical variable and converted to a set of binary dummy variables. Two predictive models were developed. The first was a linear regression model, chosen for its interpretability and the ease with which a linear functional form can be directly incorporated into the exact mixed-integer optimization model, resulting in an MIQP formulation when combined with the quadratic revenue term. The second was a gradient boosting regression model implemented using the XGBoost algorithm [29], which can capture nonlinear relationships and higher-order interactions between price and contextual features. While such a model cannot be directly embedded into an exact optimization formulation, it can be easily integrated into a learnheuristic framework, where the machine learning model acts as a surrogate for $f(\cdot)$ inside a metaheuristic optimization procedure. Model quality was assessed for both training and test data using the coefficient of determination (R^2), the mean

absolute error (MAE), and the root mean squared error (RMSE), allowing for a quantitative comparison of the linear and nonlinear approaches in terms of accuracy and generalization ability. Table 2 summarizes the performance of both models on training and test sets using the R^2 value, MAE, and RMSE. On the test set, the linear model attains moderate R^2 values around 0.76, indicating that it explains a substantial but limited portion of the variance. Its MAE and RMSE values (about 24 and 34 units of sales, respectively) show that the average prediction error is significant and that some predictions deviate even more strongly from the true sales. In contrast, the XGBoost model achieves a near-perfect fit with R^2 close to 1, and very low MAE and RMSE values (about 1.6 and 3.1 units, respectively), indicating both small average errors and minimal occurrences of large deviations. Since the performance of the XGBoost model is nearly perfect, in our computational experiments, we will consider the estimated sales quantity provided by this model as a proxy for the actual sales quantity in real-life. However, it is important to emphasize that the purpose of this experimental design is not to compare predictive models but to analyze pricing decisions under different modeling and optimization approaches. In particular, there exists a trade-off between model expressiveness and optimization tractability: while flexible nonlinear models such as XGBoost can closely approximate the underlying sales patterns, they are not straightforward to embed within exact optimization formulations, whereas simpler linear models can be efficiently incorporated into exact solvers but may introduce approximation errors. The nonlinear data-generating process is intentionally constructed to create a controlled setting in which this trade-off can be clearly observed, allowing us to examine how these differences affect the resulting pricing decisions and revenue performance. This approach will allow us to compare the performance of the exact solver using the linear analytical predictive model against the learnheuristic algorithm employing the nonlinear, more accurate predictive model.

Table 2. Model performance metrics for train and test datasets.

Model	R^2	MAE	RMSE
Linear–Train	0.7619	24.36	34.86
Linear–Test	0.7630	24.65	34.57
XGBoost–Train	0.9993	1.22	1.90
XGBoost–Test	0.9981	1.63	3.09

The analytical expression of the linear regression model is given in Equation (15):

$$\begin{aligned} \hat{s}_t = 75.1828 - 7.3275\ p_t + 0.0131\ income + 2.1525\ loyalty \\ -0.3515\ cp_t + 100.4544\ promo_t + \sum_{h=2}^{30} \beta_h\ zip_h \end{aligned} \tag{15}$$

where: $\beta_2 = 1.8180$, $\beta_3 = 1.0572$, $\beta_4 = 0.3973$, $\beta_5 = 0.8575$, $\beta_6 = 2.0943$, $\beta_7 = 3.6073$, $\beta_8 = 2.1875$, $\beta_9 = 1.7626$, $\beta_{10} = 2.1520$, $\beta_{11} = 0.8667$, $\beta_{12} = 1.8058$, $\beta_{13} = 1.1033$, $\beta_{14} = 1.1326$, $\beta_{15} = 3.0868$, $\beta_{16} = -0.7558$, $\beta_{17} = 2.4145$, $\beta_{18} = 0.8580$, $\beta_{19} = 2.8138$, $\beta_{20} = 2.9100$, $\beta_{21} = 2.9521$, $\beta_{22} = 4.4999$, $\beta_{23} = -0.7742$, $\beta_{24} = 2.7212$, $\beta_{25} = 1.9329$, $\beta_{26} = 2.7416$, $\beta_{27} = 4.3564$, $\beta_{28} = 1.6214$, $\beta_{29} = 2.0974$, and $\beta_{30} = 2.2127$.

The estimated price coefficient is strongly negative ($\gamma = -7.3275 < 0$), which ensures that the revenue function derived from this linear sales approximation is concave in prices. Consequently, the corresponding maximization problem is a convex QP that can be solved using standard solvers. Building on the results of the predictive models, the subsequent computational experiments will compare the performance of the exact QP solver using the linear regression model described in Equation (15) against a learnheuristic

using the nonlinear XGBoost model. In particular, the experimental evaluation will proceed as follows:

1. First, we will generate a new synthetic dataset to simulate different clients, each characterized by specific attributes and purchasing behaviors. Each client is represented across multiple days, reflecting a multi-period setting. This new dataset provides the instances to be solved in the optimization problem.
2. The QP optimization problem will be solved for the selected instance using Gurobi, where the sales value is estimated by the linear regression model. This provides an optimal price vector $p' = (p'_1, \ldots, p'_n)$ together with the corresponding total revenue value p'_{value}. To explore the impact of capacity constraints, different values of the maximum quantity $q_{max} \in Q$ will be tested, and the resulting total revenues recorded, enabling the construction of a Pareto frontier with q_{max} on the x-axis and total revenue on the y-axis.
3. The same instance will be solved using a PSO-based learnheuristic approach, where the sales are predicted by the nonlinear XGBoost model. This heuristic will provide a near-optimal price vector $p'' = (p''_1, \ldots, p''_n)$ and total revenue p''_{value}, again evaluated over the same q_{max} values employed before.
4. To fairly assess the quality of the solutions, the true total revenue corresponding to the linear model solution p' will be recalculated using the nonlinear XGBoost sales predictions (which, as previously discussed, are assumed to represent the actual sales), with predicted sales quantities floored to integers, resulting in p'_{true_value}.
5. The Pareto curves for p'_{value}, p''_{value}, and p'_{true_value} will be plotted together to visually compare the trade-offs between capacity and revenue for the linear exact solver and the nonlinear metaheuristic.

The exact solution approach relies on the QP model introduced in Section 3, where Equation (12) is relaxed so that prices are allowed to take continuous values, i.e., $p_t \in \mathbb{R}^+$ for all $t \in H$. Before presenting the computational experiments, the following section provides a detailed description of the proposed learnheuristic approach.

6. Learnheuristic Solution Approach

Particle swarm optimization is a population-based metaheuristic inspired by the collective behavior of birds or fish [30]. It is widely used for continuous optimization because it can explore complex, nonlinear, and nonconvex search spaces without requiring gradient information. PSO is suitable for problems like the one in this paper, where the objective function is nonlinear, may be non-differentiable, and traditional gradient-based solvers may fail in the presence of constraints [31]. The selection of PSO over other metaheuristics was motivated by its simplicity, relatively small number of hyperparameters, and ease of configuration for continuous optimization tasks. Although alternative methods could also be employed for this type of problem, PSO has been extensively used in related real-valued and nonlinear optimization domains, as discussed in Section 2, which makes it a well-established and reliable choice for the dynamic pricing setting analyzed in this work. In our dynamic pricing problem, PSO is used to optimize a sequence of prices over a finite horizon $H = \{1, 2, \ldots, n\}$. We consider a price vector $p = (p_1, p_2, \ldots, p_n)$, where $p_t \in [p_{min}, p_{max}]$ for all $t \in H$. The expected sales in period t are predicted by a sales model $\hat{s}_t = f(p_t, X_t)$, where $f(\cdot)$ is learned from historical data using an XGBoost model described in Section 5, while X_t is a vector of contextual information available at the start of the horizon. Although the learnheuristic is validated empirically, its convergence properties can be related to existing PSO theory. Classical analyses show that, for standard parameter choices, inertia–weight PSO converges to a stable point because particle velocities diminish over time and the search domain is bounded [32,33]. In constrained settings, the projection

mechanism used in Algorithm 1 (clipping prices to $[p_{min}, p_{max}]$) is known to preserve feasibility and lead the swarm toward a boundary or interior stationary point [34,35]. While the XGBoost-based sales function is nonlinear and non-smooth, theoretical work indicates that PSO still converges almost surely to a stable solution in compact, black-box search spaces [36].

Algorithm 1 PSO-based Learnheuristic

Input: horizon length n, inventory limit q_{max}, price bounds $[p_{min}, p_{max}]$, context vectors $\{X_t\}_{t=1}^{n}$, sales model $f(p_t, X_t)$
Initialize: swarm of l particles $\{p^i = (p_1^i, \ldots, p_n^i)\}_{i=1}^{l}$ with random prices $p_t^i \in [p_{min}, p_{max}]$
Initialize l velocities $\{v^i = (v_1^i, \ldots, v_n^i)\}_{i=1}^{l}$ to zero, and initialize w and v_max
for $i = 1$ to l **do**
    fitness$(p^i) \leftarrow$ GETFITNESS(p^i)
end for
Store personal best positions and global best position based on fitness
while stopping criterion not met **do** ▷ Main loop
    Update inertia value w
    for particle $i = 1$ to l **do**
        for time period $t = 1$ to n **do**
            Sample $r_1, r_2 \sim U(0,1)$
            Update velocity: $v_t^i \leftarrow w v_t^i + c_1 r_1 (p_{t,best}^i - p_t^i) + c_2 r_2 (p_{t,best}^g - p_t^i)$
            Clip v_t^i to v_max
            Update position: $p_t^i \leftarrow p_t^i + v_t^i$
            Clip to bounds: $p_t^i \leftarrow \min\left(\max(p_t^i, p_{min}), p_{max}\right)$
        end for
        fitness$(p^i) \leftarrow$ GETFITNESS(p^i)
        if fitness$(p^i) >$ fitness(p_{best}^i) **then**
            $p_{best}^i \leftarrow p^i$
            if fitness$(p^i) >$ fitness(p_{best}^g) **then**
                $p_{best}^g \leftarrow p^i$
            end if
        end if
    end for
end while
Output: best price vector found: $p^* = p_{best}^g$

The PSO framework uses l particles (Algorithm 1). Each particle $i \in \{1, \ldots, l\}$ represents a candidate price vector, $p^i = (p_1^i, \ldots, p_n^i)$, and has an associated velocity vector $v^i = (v_1^i, \ldots, v_n^i)$, which guides its movement in the search space. Initially, particle positions are randomly sampled within the feasible price bounds, velocities are set to zero, and the fitness (total revenue) of each particle is evaluated using Algorithm 2. Personal best positions for each particle and the global best position are initialized accordingly. At each iteration k of the PSO algorithm, particle velocities and positions are updated using the standard PSO rules. For each particle $i \in \{1, \ldots, l\}$ and each time period $t \in H$,

$$v_t^{i,k+1} = w\, v_t^{i,k} + c_1\, r_1\, (p_{t,best}^{i,k} - p_t^{i,k}) + c_2\, r_2\, (g_{t,best}^{k} - p_t^{i,k}) \tag{16}$$

$$p_t^{i,k+1} = p_t^{i,k} + v_t^{i,k+1} \tag{17}$$

where w is the inertia weight, c_1 and c_2 are acceleration coefficients, and $r_1, r_2 \sim U(0,1)$. After updating, prices are projected back to the feasible interval $[p_{min}, p_{max}]$. The fitness of

each particle is recomputed after the position update using Algorithm 2, which calculates total expected revenue while enforcing the inventory constraint:

$$\text{fitness}(p^i) = \sum_{t=1}^{n} s_t^i \, p_t^i \tag{18}$$

$$s_t^i = \min\Big(\max(0, \lfloor \hat{s}_t^i \rfloor), \, q_{t-1}\Big) \tag{19}$$

$$q_t = q_{t-1} - s_t^i, \quad q_0 = q_{\max} \tag{20}$$

If a particle improves its personal best fitness, the personal best is updated. If the new personal best is better than the current global best, the global best is updated as well. This loop continues until a stopping criterion is met (e.g., maximum iterations or convergence). The algorithm outputs the global best price vector $p^* = p_{best}^g$, which maximizes total revenue under the inventory constraint. This approach allows the PSO-based learnheuristic to efficiently explore the space of price sequences, employing both the predictive power of the XGBoost model and the metaheuristic search over feasible price policies.

Algorithm 2 GETFITNESS(p)

revenue $\leftarrow 0$, $q \leftarrow q_{\max}$
for time period $t = 1$ to n **do**
  if $q = 0$ **then**
    $s_t \leftarrow 0$
  else
    $\hat{s}_t \leftarrow f(p_t, X_t)$
    $s_t \leftarrow \min\big(\max(0, \lfloor \hat{s}_t \rfloor), \, q\big)$
    $q \leftarrow q - s_t$
    revenue $\leftarrow$ *revenue* $+ s_t \cdot p_t$
  end if
end for
return *revenue*

7. Experiments and Analysis of Results

In order to test both approaches (the exact solver with a linear regressor and the metaheuristic with a non-linear regressor), several experiments were designed. In this numerical study, n was set to 10 as an example for considering multi-period sales. A total of 20 new customer profiles were created using a similar approach as in Section 4, in terms of some basic identifiers like *zip*, *income*, and *loyalty*. As shown in Table 3, for each customer and time period, competitor prices and promotion flags were generated. They were labeled as cp_1, $promo_1$, …, cp_{10}, $promo_{10}$. For each customer instance, the decision variable p_t was restricted to the range $[5, 25]$. The fixed inventory level $q_{\max}$ was selected from an arithmetic sequence with a common difference of 50 within the range $[100, 1000]$. In total, 380 instances were solved using both the exact solver described in Section 3 and the learnheuristic method introduced in Section 6. For the PSO-based learnheuristic, standard parameter settings were adopted, with swarm size $l = 50$, a linearly decreasing inertia weight ranging from $w = 0.9$ to $w = 0.36$, a maximum velocity of 4 and acceleration coefficients $c_1 = c_2 = 1.5$, which are commonly used in the literature. The maximum number of iterations was set to 150. Preliminary experiments with different parameter configurations were conducted, but the selected settings provided a good balance between solution quality and computational time, and were therefore used in all experiments.

Table 3. Experimental input instances: 20 new customers over 10 periods.

Instance	Zip	Income	Loyalty	cp_1	Promo_1	cp_2	Promo_2	…	Promo_10	cp_10
1	7	4172.56	0	23.79	0	20.22	1	…	24.01	0
2	20	5616.02	1	22.90	1	16.23	1	…	16.47	0
3	29	4179.99	1	16.96	0	20.42	0	…	17.64	0
4	15	4177.15	1	23.44	0	14.88	1	…	13.97	0
5	11	5141.68	0	6.77	0	15.45	1	…	10.86	0
6	8	2731.05	0	8.92	0	13.55	0	…	11.57	1
7	29	2886.32	0	5.90	0	5.51	1	…	18.45	0
8	21	4060.41	0	11.51	0	7.16	0	…	20.05	0
9	7	3557.37	0	12.77	1	5.63	1	…	20.83	0
10	26	5251.95	0	10.43	0	17.73	1	…	20.79	1
11	19	3668.52	0	21.57	0	11.29	0	…	6.82	0
12	23	3163.73	1	12.14	0	15.17	0	…	14.89	0
13	11	7364.02	0	10.62	0	23.15	0	…	6.15	1
14	11	4482.00	0	15.85	0	9.99	0	…	15.99	1
15	24	4885.02	0	7.82	0	13.21	1	…	13.83	1
16	21	3152.20	0	21.04	0	20.11	1	…	22.75	0
17	4	4081.81	0	6.49	1	9.58	0	…	12.02	0
18	8	4947.64	0	24.74	0	6.54	0	…	7.34	0
19	24	3415.97	1	20.44	0	10.80	0	…	7.86	0
20	3	5347.55	1	8.97	0	8.22	0	…	20.23	0

Due to limitations in model representation, these two approaches handle s_t (which depends on the decision variables) differently. Optimization solvers such as Gurobi employ a linear expression trained via linear regression (Equation (15)), since they can only handle linear or quadratic programming problems. In contrast, heuristic algorithms such as the PSO metaheuristic applied here can accommodate nonlinear functions and therefore incorporate XGBoost predictions for sales volume. Since the linear regression model provides a less accurate fit to real data, the 'exact solution' obtained by the solver requires verification against actual sales. As discussed in Section 5, sales predicted by XGBoost are regarded as a proxy for the actual sales that would occur. Accordingly, the solutions produced by the exact method are re-evaluated using the trained XGBoost model to determine their real-world validity. Table 4 summarizes the average results obtained in this study by applying the different optimization approaches to each instance. The first column reports the different maximum inventory levels q_{max} evaluated. The columns `EM_pred`, `EM_real`, and `LH` show the average outcomes under the three main scenarios. `EM_pred` corresponds to the revenue obtained from the exact solver, which uses the linear regression model to predict sales. `EM_real` represents the 'real' revenue obtained by evaluating the `EM_pred` solutions with the XGBoost model, which is assumed to provide the real-life sales. `LH` shows the revenue obtained via the learnheuristic approach, which combines XGBoost with the PSO heuristic, directly integrating nonlinear sales predictions into the optimization. The percentage difference columns, `Gap% (1-2)` and `Gap% (3-2)`, report the average deviations between the different scenarios. Specifically, `Gap% (1-2)` measures the deviation of `EM_real` from `EM_pred`, quantifying the revenue loss due to the approximation introduced by the linear regression model. It is first computed for each instance as $(\texttt{EM_pred} - \texttt{EM_real})/\texttt{EM_pred} \times 100\%$ and then averaged across all instances. Similarly, `Gap% (3-2)` expresses, on average, the relative difference between the revenues of `EM_real` and `LH`, taking `LH` as the benchmark. We adopt `LH` as the denominator because `EM_real` occasionally produces zero revenues, which would lead to undefined or unstable ratios. Thus, the gap is computed as $(\texttt{LH} - \texttt{EM_real})/\texttt{LH} \times 100\%$ for each instance and then averaged across instances. Note that these averaged gap values are obtained not from the averaged revenues but from averaging instance-level percentage gaps. Averaging

across instances allows us to highlight consistent trends while mitigating instance-specific variability. The complete set of results for all instances is available in the online repository (https://github.com/MarcEscoto/Multi-Period-Revenue-Optimisation.git, accessed on 23 March 2026). The results reveal a consistent pattern across instances and inventory levels. At low values of q_{max}, the discrepancy between `EM_pred` and `EM_real` is extremely high, often exceeding 90%, indicating that solutions based on the linear model substantially overestimate real revenues. As q_{max} increases, `EM_pred` predictions gradually align better with real revenues. On the other hand, the learnheuristic approach achieves better results on average across all inventory levels, with the performance gap being larger for smaller values of q_{max}.

Table 4. The average results of the Exact (`EM`) and learnheuristic (`LH`) across all instances.

q_Max	EM_Pred (1)	EM_Real (2)	LH (3)	Gap% (1–2)	Gap% (3–2)
100	2259.18	101.29	1713.98	95.71	95.61
150	3232.18	105.35	2469.16	97.03	96.76
200	4113.86	102.26	3170.59	97.86	97.62
250	4888.79	140.16	3818.73	97.22	96.69
300	5582.63	225.64	4439.89	95.50	94.63
350	6188.73	478.97	5034.39	90.70	89.06
400	6723.67	918.17	5573.12	83.42	80.90
450	7188.87	1473.38	6085.93	75.46	71.82
500	7584.38	2247.42	6475.97	65.45	60.36
550	7910.98	3212.09	6978.22	54.12	49.08
600	8169.35	4190.72	7305.83	43.47	37.60
650	8361.71	5237.93	7705.33	32.75	27.99
700	8496.27	6069.92	7984.09	24.71	20.76
750	8581.97	6864.70	8243.58	17.70	14.89
800	8630.44	7329.12	8415.93	13.85	11.97
850	8657.60	7639.90	8634.26	11.60	11.71
900	8672.33	7783.47	8786.24	10.60	11.84
950	8678.09	7835.94	8905.99	10.24	12.45
1000	8679.06	7843.96	8975.89	10.19	12.90

To visualize the dispersion of results across instances for each method, Figures 1 and 2 present boxplots of the percentage gaps for each value of q_{max} (the mean value for each boxplot is represented by a red triangle). To improve readability, the boxplots are displayed in grouped pairs of consecutive q_{max} values (e.g., 100–150, 200–250, etc.). Figure 1 illustrates the gap between `EM_pred` and `EM_real`. The results show that for low values of q_{max}, the gap is consistently very high across all instances, with most values concentrated close to 100%, indicating that the linear model substantially overestimates revenues under tight inventory constraints. As q_{max} increases, both the median and the dispersion of the gap decrease progressively, revealing that the linear approximation becomes more reliable when inventory constraints are relaxed. In a few instances, slightly negative gap values are observed, indicating that the linear model occasionally underestimate revenues. In addition, variability across instances is more pronounced at intermediate values of q_{max}, while the dispersion narrows again for larger values, suggesting a stabilization of model performance. Figure 2 presents the gap between `EM_real` and the `LH` approach. The figure shows that the learnheuristic generally outperforms the re-evaluated exact solutions across all inventory levels, with the largest differences observed at low values of q_{max}. As inventory increases, the gap decreases and becomes more concentrated, mirroring the behavior observed in Figure 1. A few negative gap values are also observed, indicating that in rare cases the re-evaluated exact solution may outperform the learnheuristic.

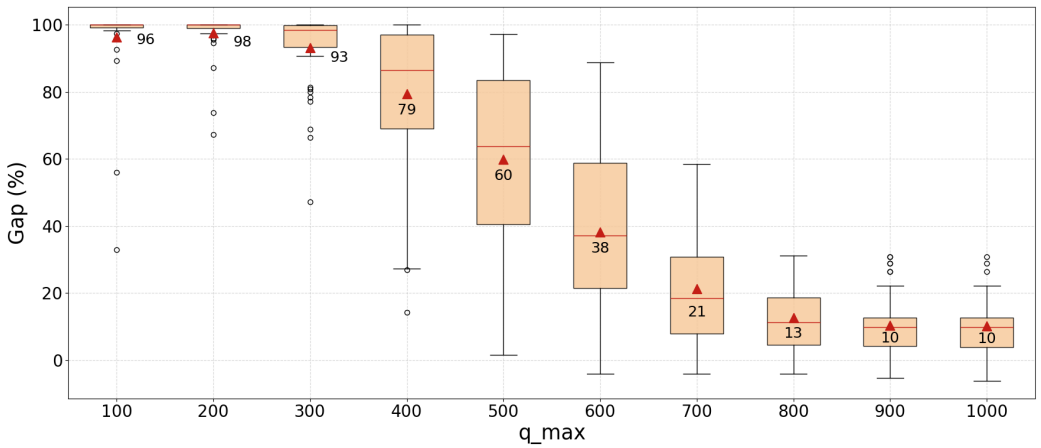

Figure 1. Revenue gap between `EM_pred` and `EM_real` for the different q_{max} values.

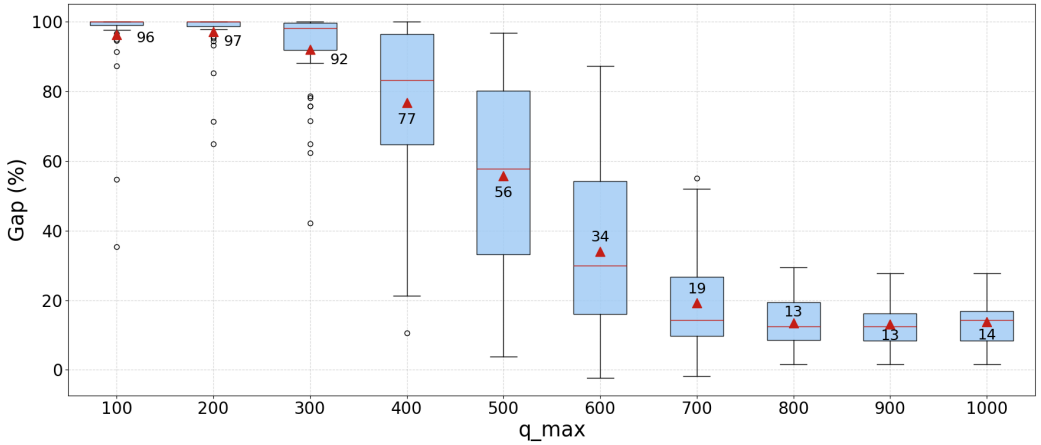

Figure 2. Revenue gap between `LH` and the `EM_real` for the different q_{max} values.

Revenue maximization occurs under a binding inventory constraint, but in practice the trade-off between profit and procurement costs forms a multi-objective optimization problem. The results are best shown via a Pareto frontier, which illustrates non-dominated solutions where higher revenue requires more inventory and vice versa. This provides a clear view of the maximum attainable profit across the fixed inventory levels. In Figure 3, the Pareto frontiers are shown for four instances, showing general similarities and particular instance-related differences. Across all instances, the Pareto frontiers show a generally consistent structure. For most instances, the exact solver based on the estimated linear sales model (blue dashed lines) achieves the highest revenues under almost all inventory limits q_{max}. However, when these 'optimal' price vectors are re-evaluated using the real-life sales, the total realized revenues (green dashed lines) are substantially lower, showing the impact of sales model misspecification. The LH and EM_pred both follow a monotonically increasing pattern: revenue rises sharply at low inventory levels and then progressively flattens as $q_{\max}$ increases, indicating diminishing returns from additional stock. In contrast, the realized revenue curve produced by the exact model takes on a pronounced sigmoid shape. At small values of $q_{\max}$, revenue remains close to zero in most instances because the linear regression approximation severely overestimates actual sales; consequently, the optimizer sets excessively high prices that result in zero sales in reality. Once inventory exceeds a critical threshold, revenue begins to grow rapidly. At the upper end of the inventory range, the exact model and, to a lesser extent, the learnheuristic frequently reach a plateau. This stagnation indicates that the maximum obtainable revenue has been reached and further increases in $q_{\max}$ no longer improve performance. Although the

overall patterns are clear, important differences emerge when the frontiers are examined instance by instance. These differences are consistent with the variability observed in the Figures 1 and 2, where a few instances exhibit atypical behaviors.

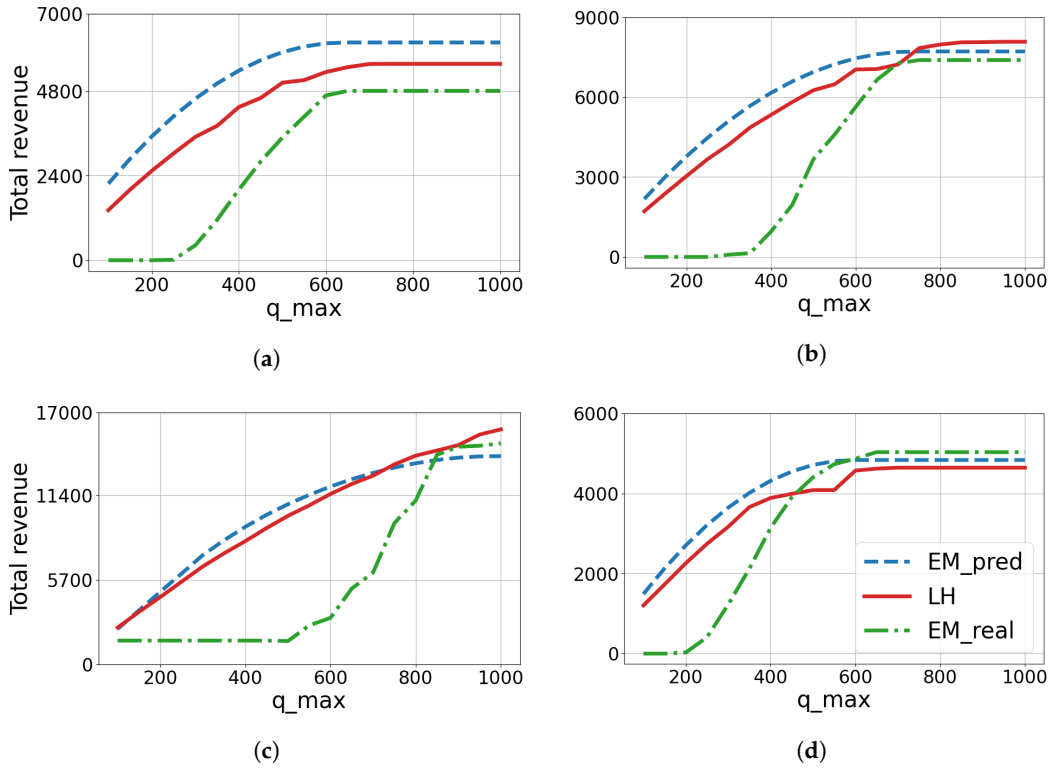

Figure 3. Pareto frontier for 4 specific customer as examples, showing total revenue versus q_{max}. (**a**) Customer 6. (**b**) Customer 11. (**c**) Customer 12. (**d**) Customer 7.

Four distinct behavioral patterns emerge from the Pareto frontiers across the instances, ranked by frequency of occurrence. The most common tendency, illustrated in Figure 3a, shows that the exact model produces high predicted revenues, yet the realized revenues show a substantial and persistent gap across most $q_{\max}$ values. In these cases, the LH provides results that fall below the EM_pred values but remain significantly superior to the realized revenues of the exact model. This pattern highlights the robustness of the learnheuristic to modeling inaccuracies, as it consistently delivers better real-world performance despite not providing optimality guarantees. A second pattern appears in Figure 3b. Here, the same gap between EM_pred and EM_real persists, but the LH occasionally outperforms EM_pred. Although the exact model solves its (approximated) problem to optimality, the maximum revenue attainable under the linear approximation can still be lower than the true maximum achievable with real sales; this highlights an additional risk of relying on a misspecified prediction model. In practice, such a risk can cause managers to underestimate the true revenue potential of their inventory and therefore order less stock than would actually be optimal. The third pattern, shown in Figure 3c, is less frequent but noteworthy: both the LH and, in some cases, the exact model itself, provide higher realized revenues for certain inventory levels than the ones initially predicted. This occurs because the linear regression sometimes underestimates sales rather than overestimating it. The exact model then optimizes under this conservative prediction and sets lower prices than the ones that would be optimal in reality. When these prices are evaluated with actual sales, the higher-than-expected sales volume produces revenues that exceed the predicted values. However, this opposite form of approximation error is equally dangerous. It can lead the decision maker to set overly conservative prices,

leaving substantial revenue on the table and underestimating the true profit potential of the inventory. Finally, the fourth pattern is visible in Figure 3d. In these instances the realized revenue of the exact model surpasses that of the learnheuristic at one or more $q_{\max}$ values. Two plausible explanations stand out: first, the learnheuristic relies on a stochastic metaheuristic search, so occasional poor random seeds can produce sub-optimal solutions; second, the linear prediction can occasionally provide a good local fit for the price levels that happen to maximize revenue in the approximated model. Beyond a certain inventory threshold (in this case, $q_{\max} \geq 600$), the predicted and realized curves of the exact model are very close, indicating that the approximation was highly accurate in that region.

Since the LH relies on a stochastic metaheuristic, a small variability analysis was conducted to evaluate the sensitivity of the results to the random seed initialization. The algorithm was executed 10 times using different seeds for a representative instance. Figure 4 shows the resulting Pareto frontiers obtained across the runs, together with the results obtained by EM_pred and EM_real. It can be observed that the minimum and maximum values obtained across the different seeds initially show virtually no variance. However, as $q_{\max}$ increases, the min–max range across seeds gradually widens. Nevertheless, even at the higher $q_{\max}$ value, the observed variability remains small and the overall trend is preserved, indicating very limited dispersion in the obtained solutions.

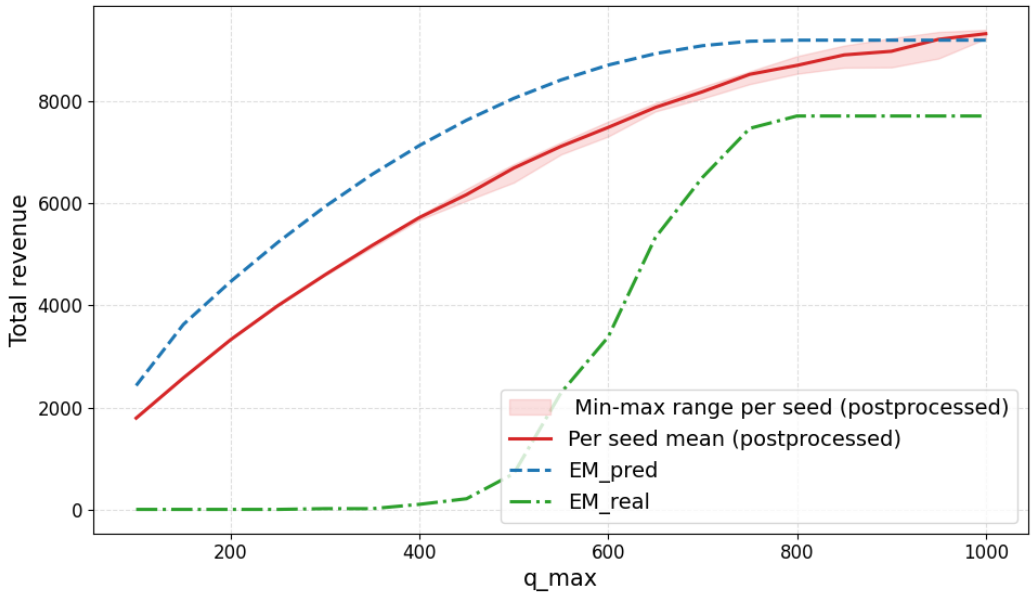

Figure 4. Pareto frontiers obtained by the LH across 10 runs with different random seeds, together with the EM_pred and EM_real results for customer 0.

8. Conclusions

The paper studies the dynamic multi-period pricing problem under inventory constraints, utilizing historical sales patterns and contextual features. We developed and compared two solution approaches: an exact QP solver based on a linear regression model for sales prediction, and a learnheuristic framework that integrates a nonlinear XGBoost model with PSO. The linear regression model, while interpretable and compatible with QP formulations, provided only moderate predictive accuracy ($R^2 \approx 0.76$ on test data) and showed substantial prediction errors, reflecting its limited capacity to capture nonlinear relationships between price, context, and sales. In contrast, the XGBoost model achieved near-perfect accuracy ($R^2 \approx 0.998$), with very low MAE and RMSE, showing its ability to closely approximate true sales dynamics.

The computational experiments involved multiple synthetic instances representing different customer profiles and multi-period settings, with varying inventory levels. The QP solver consistently produced price vectors that maximize predicted revenue under the linear model. However, when these solutions were evaluated using the XGBoost predictions as a proxy for real sales, large discrepancies emerged, especially under tight inventory constraints, with revenue overestimation often reaching 100%. This overestimation gradually decreased as inventory levels increased. In comparison, the learnheuristic approach directly incorporates the highly accurate nonlinear XGBoost predictions into the PSO optimization and generated pricing strategies that provided revenues much closer to actual sales outcomes across all instances and inventory levels. Pareto frontier analysis also remarked the same general behavior. However, although the learnheuristic generally outperforms the exact method, in a minority of instances, the exact model actually obtains higher realized revenues than the LH. This reveals both the optimality gap of the stochastic metaheuristic search in the LH and the strong potential of the EM to deliver superior results once equipped with better sales approximations.

Finally, while our study shows the performance advantages of the learnheuristic approach, it is important to recognize the trade-off between predictive accuracy and interpretability. Linear models in the exact approach offer transparency, while less transparent models like XGBoost achieve higher accuracy at the cost of interpretability. The occasional superiority of the EM further underscores the value of enhancing its predictive component while preserving interpretability. Future research could enhance the predictive component of the exact model while maintaining interpretability, using techniques such as elastic nets, linear splines, or, in the longer term, constraint learning, where machine learning predicts dynamic aspects that are then linearized for inclusion in exact models.

Author Contributions: Conceptualization, A.A.J.; methodology, A.A.J., Y.M. and V.M.; validation, M.E. and A.A.J.; data curation, Y.M. and V.M.; original draft preparation, A.A.J., Y.M., V.M. and M.E.; review and editing, A.A.J.; supervision, M.E. and A.A.J. All authors have read and agreed to the published version of the manuscript.

Funding: This work has been partially supported by the Spanish Ministry of Science, Innovation and Universities/AEI (PID2022-138860NB-I00, AIA2025-163553-C44), the Generalitat Valenciana (2024-CIAICO-117) and Coca-Cola EPP (CE20231180).

Data Availability Statement: All data is available at: https://github.com/MarcEscoto/Multi-Period-Revenue-Optimisation, accessed on 23 March 2026.

Conflicts of Interest: Author Veronica Medina was employed by the company ICSO Analytics SL. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Kopalle, P.K.; Pauwels, K.; Akella, L.Y.; Gangwar, M. Dynamic pricing: Definition, implications for managers, and future research directions. *J. Retail.* **2023**, *99*, 580–593. [CrossRef]
2. Ranjan, A.; Jha, J. Multi-period dynamic pricing model for deteriorating products in a supply chain with preservation technology investment and carbon emission. *Comput. Ind. Eng.* **2022**, *174*, 108817. [CrossRef]
3. Li, X.; Zheng, Z. Dynamic pricing with external information and inventory constraint. *Manag. Sci.* **2024**, *70*, 5985–6001.
4. Das, P.; Pervin, T.; Bhattacharjee, B.; Karim, M.R.; Sultana, N.; Khan, M.S.; Hosien, M.A.; Kamruzzaman, F. Optimizing real-time dynamic pricing strategies in retail and e-commerce using machine learning models. *Am. J. Eng. Technol.* **2024**, *6*, 163–177. [CrossRef]
5. Calvet, L.; Armas, J.d.; Masip, D.; Juan, A.A. Learnheuristics: Hybridizing metaheuristics with machine learning for optimization with dynamic inputs. *Open Math.* **2017**, *15*, 261–280. [CrossRef]
6. Bayliss, C.; Juan, A.A.; Currie, C.S.; Panadero, J. A learnheuristic approach for the team orienteering problem with aerial drone motion constraints. *Appl. Soft Comput.* **2020**, *92*, 106280. [CrossRef]

7. Luo, Y.; Sun, W.W.; Liu, Y. Distribution-free contextual dynamic pricing. *Math. Oper. Res.* **2024**, *49*, 599–618. [CrossRef]
8. Bu, J.; Simchi-Levi, D.; Wang, C. Context-based dynamic pricing with partially linear demand model. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 23780–23791.
9. Liu, P.; Yang, Z.; Wang, Z.; Sun, W.W. Contextual dynamic pricing with strategic buyers. *J. Am. Stat. Assoc.* **2025**, *120*, 896–908. [CrossRef]
10. Wang, Y.; Chen, X.; Chang, X.; Ge, D. Uncertainty quantification for demand prediction in contextual dynamic pricing. *Prod. Oper. Manag.* **2021**, *30*, 1703–1717. [CrossRef]
11. Miao, S.; Chen, X.; Chao, X.; Liu, J.; Zhang, Y. Context-based dynamic pricing with online clustering. *Prod. Oper. Manag.* **2022**, *31*, 3559–3575. [CrossRef]
12. Chen, X.; Simchi-Levi, D.; Wang, Y. Utility fairness in contextual dynamic pricing with demand learning. *Manag. Sci.* **2026**, *72*, 2619–2633. [CrossRef]
13. Nambiar, M.; Simchi-Levi, D.; Wang, H. Dynamic learning and pricing with model misspecification. *Manag. Sci.* **2019**, *65*, 4980–5000. [CrossRef]
14. Ullah, M.; Khan, I.; Sarkar, B. Dynamic pricing in a multi-period newsvendor under stochastic price-dependent demand. *Mathematics* **2019**, *7*, 520. [CrossRef]
15. Wang, Y.; Chen, B.; Simchi-Levi, D. Multimodal dynamic pricing. *Manag. Sci.* **2021**, *67*, 6136–6152. [CrossRef]
16. Chung, H.; Ahn, H.S.; Chun, S.Y. Dynamic pricing with point redemption. *Manuf. Serv. Oper. Manag.* **2022**, *24*, 2134–2149. [CrossRef]
17. Mumbower, S. Emerging research on airline ancillaries: A review of offer management and dynamic pricing literature. *J. Revenue Pricing Manag.* **2025**, *24*, 353–367 . [CrossRef]
18. Hu, Y.; Li, S.; Guo, H.; Zhang, L. Dynamic pricing for perishable product in the presence of low-carbon preference. *J. Ind. Prod. Eng.* **2025**, *42*, 88–101. [CrossRef]
19. Liang, Y.; Hu, Y.; Luo, D.; Zhu, Q.; Chen, Q.; Wang, C. Distributed dynamic pricing strategy based on deep reinforcement learning approach in a presale mechanism. *Sustainability* **2023**, *15*, 10480. [CrossRef]
20. Cho, J.; Papavasiliou, A. Pricing under uncertainty in multi-interval real-time markets. *Oper. Res.* **2023**, *71*, 1928–1942. [CrossRef]
21. Chen, X.; Wang, C.; Jia, D.; Bai, Y. Multi-period pricing and order decisions for fresh produce with option contracts. *Ann. Oper. Res.* **2024**, *335*, 79–110. [CrossRef]
22. Perivier, N.; Goyal, V. Dynamic pricing and assortment under a contextual mnl demand. *Adv. Neural Inf. Process. Syst.* **2022**, *35*, 3461–3474.
23. Hua, J.; Yan, L.; Xu, H.; Yang, C. Markdowns in e-commerce fresh retail: A counterfactual prediction and multi-period optimization approach. In Proceedings of the 27th ACM SIGKDD Conference on Knowledge Discovery & Data Mining, Singapore, 14–18 August 2021; pp. 3022–3031.
24. Tullii, M.; Gaucher, S.; Merlis, N.; Perchet, V. Improved algorithms for contextual dynamic pricing. *Adv. Neural Inf. Process. Syst.* **2024**, *37*, 126088–126117.
25. Zhu, Z.; Wu, B.; Wang, T. Research on Vegetable Cost Pricing and Sales Forecast Based on PSO Optimization and Forecast Model. *Acad. J. Bus. Manag.* **2024**, *6*, 141–146. [CrossRef]
26. Bade, S.O.; Tomomewo, O.S.; Meenakshisundaram, A.; Dey, M.; Alamooti, M.; Halwany, N. Multi-Criteria Optimization of a Hybrid Renewable Energy System Using Particle Swarm Optimization for Optimal Sizing and Performance Evaluation. *Clean Technol.* **2025**, *7*, 23 . [CrossRef]
27. Zhao, Y.; Wu, F.; Pang, J.; Zhong, W. New heterogeneous comprehensive learning particle swarm optimizer enhanced with low-discrepancy sequences and conjugate gradient method. *Swarm Evol. Comput.* **2025**, *93*, 101848. [CrossRef]
28. Goodfellow, I.; Bengio, Y.; Courville, A.; Bengio, Y. *Deep Learning*; MIT Press: Cambridge, UK, 2016; Volume 1.
29. Deka, P.P.; Weiner, J. *XGBoost for Regression Predictive Modeling and Time Series Analysis: Learn How to Build, Evaluate, and Deploy Predictive Models with Expert Guidance*; Packt Publishing Ltd.: Birmingham, UK, 2024.
30. Clerc, M. *Particle Swarm Optimization*; John Wiley & Sons: Hoboken, NJ, USA, 2010; Volume 93.
31. Gad, A.G. Particle swarm optimization algorithm and its applications: A systematic review. *Arch. Comput. Methods Eng.* **2022**, *29*, 2531–2561. [CrossRef]
32. Clerc, M.; Kennedy, J. The particle swarm-explosion, stability, and convergence in a multidimensional complex space. *IEEE Trans. Evol. Comput.* **2002**, *6*, 58–73. [CrossRef]
33. Trelea, I.C. The particle swarm optimization algorithm: Convergence analysis and parameter selection. *Inf. Process. Lett.* **2003**, *85*, 317–325. [CrossRef]
34. Coello, C.A.C. Theoretical and numerical constraint-handling techniques used with evolutionary algorithms: A survey of the state of the art. *Comput. Methods Appl. Mech. Eng.* **2002**, *191*, 1245–1287. [CrossRef]

35. Mezura-Montes, E.; Coello, C.A.C. Constrained optimization via multiobjective evolutionary algorithms. In *Multiobjective Problem Solving from Nature: From Concepts to Applications*; Springer: Berlin/Heidelberg, Germany, 2007; pp. 53–75.
36. Kadirkamanathan, V.; Selvarajah, K.; Fleming, P.J. Stability analysis of the particle dynamics in particle swarm optimizer. *IEEE Trans. Evol. Comput.* **2006**, *10*, 245–255. [CrossRef]

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.