

Multi-Agent Reinforcement Learning for Dynamic Pricing in Supply Chains: Benchmarking Strategic Agent Behaviours under Realistically Simulated Market Conditions

Thomas Hazenberg¹, Yao Ma^{1*}, Seyed Sahand Mohammadi Ziabari^{1,2}, and Marijn van Rijswijk³

¹Informatics Institute, University of Amsterdam, 1098XH Science Park, Amsterdam, The Netherlands

²Department of Computer Science and Technology, SUNY Empire State University, Saratoga Springs, NY, USA

³KPMG N.V, 1186 DS Amstelveen, The Netherlands

thomas.hazenberg@student.uva.nl,y.ma3@uva.nl,sahand.ziabari@sunyempire.edu,vanrijswijk.marijn@kpmg.nl

ABSTRACT

This study investigates how Multi-Agent Reinforcement Learning (MARL) can improve dynamic pricing strategies in supply chains, particularly in contexts where traditional ERP systems rely on static, rule-based approaches that overlook strategic interactions among market actors. While recent research has applied reinforcement learning to pricing, most implementations remain single-agent and fail to model the interdependent nature of real-world supply chains. This study addresses that gap by evaluating the performance of three MARL algorithms: MADDPG, MADQN, and QMIX against static rule-based baselines, within a simulated environment informed by real e-commerce transaction data and a LightGBM demand prediction model. Results show that rule-based agents achieve near-perfect fairness (Jain's Index: 0.9896) and the highest price stability (volatility: 0.024), but they fully lack competitive dynamics. Among MARL agents, MADQN exhibits the most aggressive pricing behaviour, with the highest volatility and the lowest fairness (0.5844). MADDPG provides a more balanced approach, supporting market competition (share volatility: 9.5 pp) while maintaining relatively high fairness (0.8819) and stable pricing. These findings suggest that MARL introduces emergent strategic behaviour not captured by static pricing rules and may inform future developments in dynamic pricing.

KEYWORDS

Multi-Agent Reinforcement Learning, Dynamic Pricing, Simulation Environment, Demand Forecasting, Supply Chain Optimization

1 INTRODUCTION

As customers demand more advanced functionalities from enterprise resource planning (ERP) systems and their vendors to manage inventories, manufacturers, suppliers, distributors, and retailers, the need for an intelligent dynamic pricing mechanism becomes more critical [18, 52, 54]. ERP systems efficiently integrate and automate core supply chain functions such as procurement, logistics, and financial planning, but their capacity to optimize dynamic pricing stays underutilized. Modern supply chains are highly interconnected and complex, with multiple stakeholders each making pricing and inventory decisions [29]. Most of the entities in these chains still rely on static pricing models. These are based on fixed rules, historical data, or cost-plus formulas, which fail to reflect real-time changes in demand, supply, or competitor behaviour [22, 35]. As a result, these static pricing strategies could lead to revenue losses and missed market opportunities. For instance, Cachon and

Feldman (2010) show that static or naïve dynamic pricing strategies can underperform by up to 22.6% in revenue compared to dynamic pricing, with an average loss of 7%, showing the real cost of failing adaptive pricing mechanisms [8]. Unlocking this would allow businesses to set prices dynamically through demand, inventory, and other factors, thus enhancing profitability and competitiveness.

Dynamic pricing offers a data-driven alternative to static pricing by adjusting prices in response to market conditions. Integrating such strategies within ERP systems comes with several challenges. It requires seamless data exchange, real-time decision-making across actors, and scalable models. Current pricing solutions lack these capabilities, especially the ability to learn autonomously from evolving market dynamics [40]. Machine learning techniques have been increasingly used in ERP systems for tasks like forecasting, but there remains a lack of solutions that realistically model decentralized and interactive pricing decisions across multiple supply chain actors in a learning manner.

Recent research has explored Reinforcement Learning (RL) as a method to optimize pricing strategies [59]. Many RL-based models, such as Bayesian approaches, feature-based learning, and Q-learning, act in a single-agent setting, where pricing decisions are made isolated. This overlooks the interdependent nature of pricing decisions across supply chain entities [29, 43]. When a retailer adjusts prices, it affects demand, thus influencing supplier production levels and inventory. Ignoring these dependencies could potentially lead to suboptimal pricing strategies.

To address this gap, we benchmark and evaluate different Multi-Agent Reinforcement Learning (MARL) algorithms for optimizing dynamic pricing in supply chains, built upon existing and extending MARL models. Unlike traditional RL methods, MARL involves multiple autonomous agents, where each represents a distinct entity with a product catalogue and pricing decisions to learn collaboratively or competitively. These agents then interact with a shared environment that is shaped by market conditions, competitor actions, and changes in supply and demand. They dynamically adjust their pricing strategies through reward-based learning, thereby potentially improving revenue generation and market adaptability.

In this context, MARL could be particularly suited for capturing the interdependent nature of pricing decisions in supply chains, which traditional single-agent RL methods fail to model [7]. This study investigates the potential of MARL to enhance dynamic pricing strategies within ERP-integrated supply chains. Existing pricing mechanisms, particularly those based on static rules or heuristics, often lack the flexibility to adapt to volatile market environments

and the strategic interactions between multiple pricing agents. By contrast, MARL offers a data-driven and adaptive alternative capable of capturing complex interdependencies and evolving agent behaviors in competitive and cooperative settings.

The research explores how various MARL algorithms, each with different learning dynamics and coordination mechanisms, can improve the adaptability, stability, and responsiveness of pricing. Comparative evaluation is conducted against traditional rule-based approaches to assess performance in environments characterized by fluctuating demand, customer segmentation, and supply-side variability.

In addition, the study examines how the efficacy of MARL-based strategies is influenced by underlying market dynamics, such as volatility, agent heterogeneity, and feedback latency. Through a series of controlled experiments, this work highlights the strengths and limitations of reinforcement learning in operational pricing tasks and contributes a practical framework for the deployment of MARL in enterprise-level decision support systems.

The remainder of this research is organized as follows. Section 2 reviews relevant literature on pricing algorithms, reinforcement learning approaches to pricing, and challenges in applying MARL to supply chain contexts. Section 3 details the methodological framework, including pre-processing of the data set, the demand model, the simulation environment, agent architectures, and the evaluation setup. Section 4 presents the experimental results, comparing MARL strategies to rule-based baselines. Section 5 interprets the findings in light of prior work, explores trade-offs, and discusses limitations. Finally, Section 6 concludes the study and outlines directions for future research.

2 RELATED WORK

Prior studies denote the importance of dynamic pricing in supply chain revenue optimization and market adaptability, but its implementation to ERP systems remains fairly limited [23, 31]. Recent work has started exploring RL for pricing optimization, but it mostly focuses on single-agent configurations. This research aims to cross this gap by using MARL to enhance pricing strategies. To position this study in the existing literature, this section focuses on three areas: (1) Pricing Algorithms, covering both static and dynamic pricing; (2) Reinforcement Learning for Pricing Optimization, distinguishing between single-agent RL and MARL; and (3) Challenges in MARL, discussing computational complexity, scalability, and data limitations in applying MARL to pricing environments.

2.1 Pricing Algorithms

To build upon existing pricing methodologies, we explore static pricing, which relies on fixed or rule-based methods, and dynamic pricing, which uses machine learning to adjust prices. These pricing approaches are fundamental in supply chain management, providing a basis for more advanced RL-based pricing strategies.

2.1.1 Static Pricing. Traditional pricing relies on rule-based methods, cost-plus pricing, and historical trends. These models typically set fixed prices based on predefined formulas rather than adapting dynamically to market conditions. Rule-based pricing, often used in retail, adjusts prices by, for instance, undercutting the lowest

competitor price or following markup rules. In spite of their simplicity, rule-based pricing remains widely used, particularly among Amazon, Walmart, and eBay [14, 55]. Still, their lack of adaptability could lead to suboptimal revenue and potential algorithmic complexity in generally competitive markets. Another common approach, cost-plus pricing, determines prices by adding a fixed margin to costs. Even though this method covers the cost, it fails to account for demand elasticity, competitive actions, or supply fluctuations, making it unsuitable for dynamic market environments [35, 49]. Static pricing strategies have also been applied in energy and financial markets, where fixed prices are used for long-term contracts and auction-based transactions [58]. While effective in stable markets, these models struggle in environments with high demand uncertainty or stochastic supply features. Recent research suggests that static pricing can sometimes achieve performance levels similar to dynamic pricing under specific conditions [53].

2.1.2 Dynamic Pricing. Dynamic pricing models are being increasingly used across industries by adjusting prices in real-time based on demand fluctuations, inventory levels, and competitive behaviour [17, 27]. Methods such as Linear Regression, Random Forests, and Gradient Boosting are commonly used to forecast sales or estimate demand [15, 21]. Unsupervised learning methods, like clustering, support segmentation-based pricing strategies. However, these approaches usually fall short in adaptability to changes and may optimize for short-term gains without thinking about long-term customer behaviour. Real-world applications for this include airline ticket pricing, hotel bookings, and e-commerce platforms, where prices change dynamically. Despite their effectiveness, dynamic pricing strategies face challenges related to customer trust, market volatility, and complexity. As a result, there is growing interest in RL techniques that can learn pricing policies through interaction and adapt over time by explicitly modelling long-term objectives.

2.2 Reinforcement Learning for Pricing Optimization

Reinforcement learning has been heavily applied in decision-making tasks, including those of pricing optimization. Traditional RL methods, such as Q-learning, Deep Q-Networks (DQN), and Actor-Critic (A2C) algorithms, extend the Markov Decision Process (MDP) framework and have shown promising results in adaptive decision-making [32, 48, 57]. This section reviews two main approaches in RL for pricing: (1) Single-Agent Reinforcement Learning, where a single decision-maker optimizes its pricing strategy independently, and (2) Multi-Agent Reinforcement Learning (MARL), which extends RL to multiple interacting agents.

2.2.1 Single-Agent Reinforcement Learning. Single-agent RL (SARL) has been extensively used in various decision-making and optimization tasks [57]. In SARL, an individual agent learns an optimal policy by interacting with an environment and receiving feedback in the form of rewards. The agent's goal is to maximize its expected reward over time. Traditional SARL methods rely on MDPs, assuming a stationary environment where the transition probabilities between states stay fixed [25]. The most fundamental RL algorithm, Q-learning, is a value-based method where an agent updates a Q-table to estimate the expected future reward for each state-action

pair. However, this method struggles with high-dimensional state spaces, leading to the invention of DQNs. These use neural networks to approximate Q-values, enabling better generalization [57]. In addition to value-based approaches, policy gradient methods optimize the policy function by calculating gradients of the expected reward for the policy parameters. A2C methods combine value and policy-based learning, where the actor learns the policy and the critic evaluates the value function to improve learning stability [20]. These methods are particularly useful for continuous action spaces. Q-learning and DQN models struggle here because of discretization challenges. Recent research has explored the scalability of SARNL models, showing that RL algorithms' performance improvements follow power-law scaling with model size [25]. This suggests that larger and more complex RL models can achieve better results. However, SARNL faces significant challenges in dynamic and multi-agent environments by assuming a stationary and predictable environment. This is unrealistic in competitive markets and supply chain systems, where multiple decision-makers interact and all influence outcomes [20].

2.2.2 Multi-Agent Reinforcement Learning. MARL extends regular RL by enabling multiple agents (such as manufacturers, suppliers, retailers, or robotic entities) to learn collaborative or competitive pricing strategies [7]. MARL has been applied successfully to logistics in the supply chain, inventory management, vehicle routing, and demand forecasting. This is not well explored in dynamic pricing [33, 47]. While single-agent RL assumes an isolated solitary decision-making entity to optimize its own strategy, MARL enables multiple agents to learn and make adaptive pricing decisions simultaneously with shared or competing objectives. This leads each agent to optimize its policy considering other evolving strategies, and hence a more dynamic and interactive environment for pricing. The interaction is most important in supply chains since the pricing, demand, inventory levels, and supply constraints are at different points of several interdependent entities [6]. MARL's ability to capture strategic interactions makes it well-suited for dynamic pricing scenarios where traditional methods fail to account for these multi-agent dependencies.

2.3 Challenges in MARL

Despite its promise, MARL faces several challenges that have limited its application in complex practical settings. The decentralized interaction of agents leads to high computational overhead, problems with scalability, and instability of the training process. Other challenges in applications from supply chain management arise where the unavailability of quality training data and effective communication mechanisms among the agents are critical issues. A combination of algorithmic advancements and computational optimizations is needed to overcome these challenges. The following subsections highlight two major obstacles: Computational Complexity and Scalability and Data Limitations, together with potential solutions that mitigate these issues.

2.3.1 Computational Complexity. MARL significantly increases computational demands due to the exponential growth of the joint state-action space as agents are added. Unlike SARNL, where policy updates are independent, MARL agents must adapt to each

others changing strategies, complicating optimization. This problem in dimensionality clogs exploration and leads to increasing training times [59, 61]. A common solution is centralized training with decentralized execution (CTDE), where agents share information during training but act independently during execution. CTDE improves learning, but it provokes a high computational cost. Alternative strategies include mean-field approximations and graph-based learning. These reduce complexity by modelling only the local interactions [59]. To ease constraints of resources, parallel and distributed learning could help, though synchronizing policies in non-stationary settings remains a burden.

2.3.2 Scalability and Data Limitations. Scaling multi-agent systems worsens state-action growth, making convergence harder and instability better, particularly in dynamic supply chains where agents must learn competing agents' behaviour. Graph-based MARL and hierarchical learning are approaches to structuring interactions more efficiently [59]. Data scarcity is another challenge: ERP pricing data is often private or hidden, which makes research lean toward synthetic or simulated environments that may not capture real-world complexity. As a result, generalization across market conditions remains a key challenge [61]. Also, inter-agent communication creates overhead, thus hindering training speed. Solutions include attention mechanisms and selective message filtering, which can improve communication efficiency without slowing down performance [59].

3 METHODOLOGY

This section describes the methodological framework developed to investigate how autonomous pricing agents can learn competitive strategies in a supply chain. The main task is to simulate a market environment in which pricing decisions from agents influence customer demand and the market. The core goal is to evaluate how different rule-based and MARL pricing strategies perform in terms of revenue, price stability, fairness, and market efficiency.

To do this, a real-world dataset was explored, preprocessed, and used to extract useful features in pricing and demand through feature selection and engineering. These features were used to train a predictive model that estimates customer demand with respect to price changes. This model functioned as the backbone for a custom simulation environment where agents set prices and receive feedback in a weekly cycle. The environment supported learning agents along with rule-based baselines. This was done by comparing strategic behaviours. The remainder of this section outlines the dataset selection and preparation, the construction of the demand model, the simulation setup, the agent architectures, and the evaluation protocol.

3.1 Dataset Exploration and Justification

To simulate realistic supply chain pricing and demand scenarios, this research made use of the publicly available Online Retail II dataset, released through the UCI Machine Learning Repository [10]. The dataset contains over one million transactions recorded by a UK-based online retailer, specialized in giftware, serving both individual consumers and wholesalers, between December 2009 and December 2011.

3.1.1 Usage in studies. Several studies have used this dataset to model customer segmentation and profitability dynamics. For instance, Chen et al. [12] employed a Recency, Frequency, and Monetary (RFM) model through k-means clustering and decision tree induction to segment customers based on behaviour patterns. This confirms the suitability of the dataset for consumer-centric business intelligence implementations. In more recent work, Chen et al. [11] continued to use this dataset to demonstrate the effectiveness of using RFM time series to model and predict customer profitability using a multilayer feed-forward neural network (MFNN). Also, the dataset’s application extends to pattern mining. Singh et al. [51] referenced real-world retail datasets, including the Online Retail II dataset, to validate algorithms to extract behavioural sequences that begin or end with specific events using prefix/suffix sequential pattern mining.

3.1.2 Dataset characteristics. The dataset’s granularity and scale enabled detailed pricing and demand modelling across time, products, and customers. It covers 5,243 unique products and 5,876 customers. Each transaction included key variables such as product ID, timestamp, price, quantity, and customer ID, allowing for temporal, behavioural, and price-level analysis. An overview of the entire dataset structure, including column names, data types, and definitions, is provided in Appendix A.

3.1.3 Exploratory Data Analysis. EDA revealed several patterns relevant to pricing and demand modelling. B2B transactions were selected due to the presence of customer identifiers and notable behavioural differences; as non-registered customers paid on average 55.7% more. Transactions were highly concentrated in the UK (92.1%) and showed strong temporal effects, such as end-of-year seasonal peaks, midday spikes, and higher activity on Thursdays. Product pricing was highly variable, with some items showing volume discounts of over 50%. Price volatility (as measured by coefficient of variation) averaged around 0.39, with a small subset exceeding 2.0 (as seen in Figure 3) [24]. Cancellations and returns were rare but were removed during cleaning. Complete descriptive statistics, volatility figures, and formulas are detailed in Appendix B.

3.2 Preprocessing

This section outlines the preprocessing pipeline applied to the dataset. This includes the removal of abnormal records, the extraction of customer and time-based features, and the construction of domain-specific variables. Additionally, product categories, week numbers, lagged and smoothed demand, price and trend indicators served as the foundational inputs for a demand prediction model as well as input states for agent models.

3.2.1 Filtering and cleaning. To ensure relevance and data quality, preprocessing started with the removal of invalid entries. Transactions with negative values for quantity or price were excluded, as well as records without a customer ID. These filters removed returns, errors, or overall incomplete purchases.

3.2.2 Feature engineering. To enhance predictive power and help align with the temporal structure of demand modelling, a variety of domain-relevant features were engineered from the dataset.

Datetime and Country Features. Multiple datetime indicators were extracted from each timestamp, including year, month, day, week, weekday, hour, and two binary flags: one for weekends (to capture behavioural differences between weekdays and weekends) and one for the holiday season, which is set to true during calendar weeks 47 to 52 to denote end-of-year peak activity. To support geographic segmentation, the categorical country column was also encoded into a numerical code label variable.

Semantic product clusters. To capture semantic similarities between products, clustering was applied to the description field. Each unique product description was embedded using Sentence-BERT (MiniLM-L6-v2), a model designed to produce semantically meaningful sentence embeddings suitable for tasks such as clustering and semantic search [46]. The resulting vector representations were grouped using K-Means clustering into 20 product categories, generating a new categorical feature. This number was chosen to balance category differentiation with avoiding fine-grained or overlapping product categories. This allowed for the representation of similar products to be grouped on semantic meaning instead of sparse identifiers, aligning with evidence that clustering based on extracted feature representations can enhance modelling performance [2, 4].

Temporal demand signals. Weekly aggregations were computed per product to align the dataset with the temporal structure of demand modelling. To capture short- and long term trends, several time-based features such as previous-week sales and over 2- and 4-week window rolling averages were derived [5].

Transformations and interaction features. To improve model behaviour and reduce skewness, demand was log-transformed. To also capture non-linear effects, price received both logarithmic and quadratic transformations. Seasonality was approximated using sine and cosine decompositions of calendar week and month values. Interaction terms (such as price-holiday, and price-category) allowed for conditional effects. Finally, features like momentum, trajectory, volatility, and price vs. category average were included to enhance sensitivity to recent temporal market behaviour. Definitions and equations for all indicators and features are in Table 8, Appendix C.

3.3 Predicting Demand

Forecasting weekly product demand is essential for simulating market behaviour and informing agent pricing. This section describes the supervised model used to simulate demand.

3.3.1 LightGBM. To forecast weekly demand, a LightGBM (Light Gradient Boosting Machine) model [30] was used. It was selected for its computational efficiency, scalability, and ability to handle high-dimensional tabular data with minimal preprocessing [50]. Unlike deep learning models, LightGBM offers interpretability via feature importance scores, aligning with ERP requirements for explainability. The model was trained on a product-week aggregated dataset (8,777 observations, 21 features) with log-transformed demand as the target.

As shown in Table 1, the model generalized well, achieving $R^2 = 0.74$ on the test set. The gap with cross-validation ($R^2 = 0.591 \pm$

Table 1: LightGBM Configuration and Performance Summary

Hyperparameter	Value
n_estimators	2048
learning_rate	0.03
num_leaves	256
Early stopping	Patience = 100 on 10% validation split
Cross-Validation R^2	0.591 ± 0.093
Validation RMSE (log)	0.4957
Test RMSE (exp.)	103.72
Test MAE (exp.)	63.45
Test R^2 (exp.)	0.74

0.093) suggests the test set contains more predictable patterns. This performance is sufficient to simulate realistic agent interactions.

3.3.2 Feature Importance and Price Sensitivity. Temporal and pricing features, including lag, rolling means, trend, and volatility, ranked highest in feature importance (Appendix C), confirming their relevance. To assess price responsiveness, counterfactual analysis was performed by scaling price in the test set ($0.5\times$ to $2.5\times$) while holding other inputs constant. The resulting demand curve (Figure 1) was smoothed using a 5-point centered rolling mean.

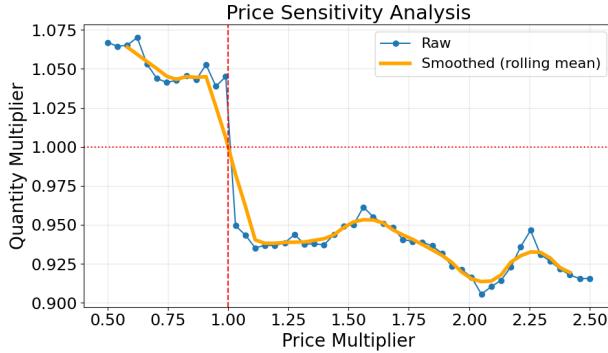


Figure 1: Smoothed price–demand curve

Price elasticity of demand (ϵ) was calculated as:

$$\epsilon = \frac{\Delta Q/Q}{\Delta P/P} \quad (1)$$

The resulting $\epsilon = -0.072$ indicates inelastic demand, which is typical for giftware products [56]. Although trained on log-transformed demand, elasticity was computed in the original scale using inverse-transformed demand predictions.

3.4 Simulation Environment Design

A custom simulation environment was developed to simulate market interactions and evaluate dynamic pricing strategies in a competitive setting. This environment provided a weekly time-stepped framework in which multiple pricing agents interact in a shared market and observe demand feedback influenced by their own and competitors’ pricing decisions.

3.4.1 Environment Overview. The environment was initialized with a list of agents and a pre-trained LightGBM-based demand prediction model (see Section 3.3.1). Each simulation step corresponds to one calendar week and updates the internal state variables such as the current date, year, week number, and holiday indicator. Agents submit product prices, and the environment uses the demand model to simulate weekly sales based on product-level features and competitive market conditions.

3.4.2 State Sharing and Agent Feedback. After predicting demand and calculating profits for each agent, the environment updates product-level demand histories and aggregates outcomes into a centralized history log. In addition, a structured market observation dictionary with week-number, holiday, and category clusters is compiled and passed to agents helping inform future pricing decisions.

3.4.3 Agent Actions and Feedback Loop. Each agent maintains a portfolio of products, each with its own cost structure and demand history. At each simulation step, agents observe the environment and select product pricing actions. For each step in the loop, it predicts demand, computes revenue, updates histories, and invokes an act on each agent to determine the next pricing action, allowing agents to refine their policies based on observed outcomes and evolving market conditions.

3.5 MARL Agents

Three distinct Multi-Agent Reinforcement Learning (MARL) frameworks were implemented and evaluated for optimal pricing strategies: Multi-Agent Deep Deterministic Policy Gradient (MADDPG), Multi-Agent Deep Q-Network (MADQN), and QMIX. Each offers different capabilities for the dynamic pricing domain, including continuous vs. discrete action spaces, degree of centralization, and agent coordination mechanisms.

3.5.1 MADDPG. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) extends DDPG to multi-agent settings, enabling stable learning in non-stationary environments through centralized training and decentralized execution [38]. Adapted from OpenAI’s reference code for TensorFlow 2.x and dynamic pricing [39], each agent uses a local actor network to map observations to continuous pricing actions. A centralized critic evaluates joint actions using the global state. The full architecture of the MADDPG agent, including actor-critic structure, smoothing, and reward design, is illustrated in Appendix E.1. This highlights the actor networks, action execution, centralized critics, and experience replay buffers, including custom contributions like recency-biased sampling and price stability penalties. The MADDPG implementation consists of:

- (1) **Actor Network:** A 3-layer fully-connected policy network (ReLU activations, tanh output) mapping local observations to continuous pricing actions.
- (2) **Critic Network:** A centralized architecture estimating Q-values based on joint observations and actions of all agents.
- (3) **Experience Replay Buffer:** Stores state-action-reward transitions with a recency bias for prioritized sampling.
- (4) **Exploration Noise:** Adds decaying Gaussian noise to the actor’s output to encourage continuous exploration during training.

While action decisions are decentralized, centralized critics leverage joint state-action information during training to improve learning stability across agents, encoding competitive market dynamics via price ratios, demand trends, seasonality, and market share metrics in agent state representations. During execution, each agent independently observes the market and adjusts prices by applying its actor network, but during training, the centralized critics leverage global information to compute more accurate gradients, facilitating stable policy learning across all agents.

3.5.2 MADQN. The Multi-Agent Deep Q-Network (MADQN) adapts DQN for multi-agent environments using discrete pricing actions (-10% to $+10\%$) [19, 41]. Unlike MADDPG’s continuous control and centralized critics, MADQN agents learn independently, relying solely on local observations and rewards. This decentralization encourages self-interested behaviour, suitable for competitive pricing without coordination. MADQN tends to produce more aggressive strategies, leading to higher average prices and sharper differentiation between agents. Its architecture (Appendix E.2) includes a fully-connected Q-network (128, 64, 32 neurons), a target network for stability, ϵ -greedy exploration, and a recency-biased replay buffer to decorrelate updates. Custom elements such as price stability penalties and prioritized sampling distinguish this implementation from standard DQN [34, 37].

3.5.3 QMIX. QMIX represents a value-based MARL approach that overcomes the limitation of independent Q-learning (like MADQN) by enabling coordinated strategies. It combines per-agent utility functions (Q-values) into a single joint action-value function through a mixing network that enforces monotonicity constraints [45].

Designed for decentralized execution with coordinated learning, each agent maintains its own Q-network for local action evaluation, while a centralized mixing network aggregates these per-agent utilities into a global action-value estimate. This mixing network, conditioned on a global state, ensures interpretability of local Q-values while enabling coordinated strategies. A coordinator pattern manages updates for efficient joint optimization during training.

QMIX’s advantage lies in representing joint action-values which is not just a sum of individual functionalities, allowing it to learn policies where inter-product pricing coordination gives better returns than independent price optimization. As shown in Appendix E.3, its architecture includes agent-level Q-networks and a centralized mixing network. In practice, QMIX fosters a more balanced price-competition dynamic, discovering strategies that maintain competitive positioning and avoid destructive spirals, ensuring pricing decisions are locally informed yet globally coordinated.

3.6 Baselines

To contextualize MARL dynamic pricing, this study implements rule-based agents. Unlike learning agents, these do not adapt strategies, but follow fixed decision rules from common retail pricing practices. These rule-based agents serve as realistic, interpretable benchmarks, representing strategies typically embedded in ERP and pricing management systems.

3.6.1 Rule-Based Pricing Agents. Each Rule-Based agent follows a specific strategy that defines how prices are adjusted weekly.

These strategies are deterministic but context-sensitive, reacting to competitor prices, historical data, demand signals, or seasonal effects. The following strategies were implemented:

- (1) **Static Markup:** A fixed cost-plus markup, independent of market conditions, serving as a simple baseline [26].
- (2) **Competitor Matching:** Agent aligns price with the average competitor price in its category, with a small (1-5%) undercut and a minimum margin [13].
- (3) **Historical Anchor:** Prices are set based on historical sales data, maintaining stability and resisting sudden market fluctuations [44].
- (4) **Demand Responsive:** Prices adjust dynamically to recent demand changes: up if demand increases, down if it declines, simulating agile data-driven responses to short-term sales [60].
- (5) **Seasonal Pricing:** Prices are modulated by seasonal demand patterns, increasing during high-demand periods (such as holidays) and normalizing off-peak, exploiting predictable consumer activity spikes [16].

3.7 Experimental Setup

The experimental framework was designed to evaluate the effectiveness of different pricing strategies in a competitive environment. A structured simulation was implemented, which allowed for controlled comparisons between MARL algorithms and baseline strategies.

3.7.1 Simulation Environment Configuration. The simulations were configured to model market dynamics over a span of two years, where each episode represented 104 weeks. At the start of each episode, four competing agents were initialized with an identical product portfolio. Each consisting of five products with category clusters (1, 2, 3, 5, and 10), where products were all initialized with identical starting prices. Then, the simulation was run for 30 episodes per experiment to allow sufficient time for MARL algorithms to converge.

3.7.2 Algorithm Hyperparameters. Each agent-type was carefully tuned and optimized for the retail pricing domain (see Table 2 for an overview).

Table 2: MARL Hyperparameters

Parameter	MADDPG	MADQN	QMIX
Learning rate	0.0001 (actor) 0.00001 (critic)	0.001	0.001
Discount factor	0.95	0.95	0.95
Exploration (decay)	Noise-based: 0.9995	ϵ -greedy: 0.995	ϵ -greedy: 0.995
Target update	$\tau = 0.001$	Every 5 steps	Every 5 steps
Batch size	64	64	64

Exploration parameters were designed to decay exponentially with increasing episodes, facilitating thorough environment exploration in early training while progressively shifting toward exploitation. For MADDPG, exploration noise started at 0.2 and decayed

multiplicatively by a factor of 0.9995 per episode, with a minimum threshold of 0.05. MADQN and QMIX employed an ϵ -greedy exploration strategy, starting at 1.0 and decaying by 0.995 per episode, also with a minimum exploration rate of 0.05.

3.7.3 Evaluation Metrics. To assess the effectiveness of the implemented MARL algorithms and benchmark strategies, multiple evaluation metrics were employed, capturing both economic performance and market dynamics. In particular, this study emphasizes emergent fairness, and market stability, reflecting the research objective to explore not only revenue optimization but also the broader implications of agent behaviour on market outcomes.

The metrics include revenue per agent, price stability, Nash Equilibrium Proximity [42], optimality gap, welfare fairness [3, 9], Jain’s fairness Index [28], market share evolution, and price volatility. Together, these metrics provide a comprehensive view of agent adaptation, competitive behaviour, and overall market outcomes. Formal definitions and computational formulas for each metric are detailed in Appendix D (Table 9).

3.7.4 Computational Resources. All experiments were executed on the Snellius National Supercomputer, operated by SURF in the Netherlands [1]. The experiments were configured to run on CPU-only mode. Each simulation utilized 8 CPU cores and was allocated 16 GB of RAM. The models were implemented using TensorFlow 2.x, optimized for CPU operations. Each full simulation was allocated up to 24 hours of computation time. The most compute-intensive components were the neural network training operations for MADDPG and QMIX.

3.8 Reproducibility

Experiments ran under fixed computational conditions (as detailed in Section 3.7.4), controlled by set scripts to minimize variability between runs. All experiment configurations (agent types, hyperparameters, simulation setups) were defined in dictionaries, serialized as JSON, and executed in isolated, and timestamped directories to prevent cross-contamination. Models, key simulation outputs, and performance metrics were systematically stored in standardized CSV files, logs, and visualizations. Methodological consistency was ensured through fixed time horizons (104 weeks per episode, 30 episodes) and unified evaluation windows. Finally, all simulations used a single, pre-trained demand model, and agents were initialized with identical product portfolios, pricing, and cost structures. The simulation environment maintained consistency with fixed weekly progressions, uniform agent action protocols, and centralized demand modelling.

4 RESULTS

Pricing adaptability refers to the ability for an agent to adjust prices to market conditions over time, which are expressed as changes in pricing magnitude and frequency. While high adaptability can enable quick responses to demand or competition fluctuations, if it is unregulated, it could cause severe instability. We evaluate it using two custom metrics (defined in Appendix D): adjustment magnitude (average absolute percentage price change between time steps) and adjustment frequency (proportion of changes exceeding a 1% threshold).

Table 3: Mean adaptability and stability metrics by agent type. Bold values indicate best-performing agents.

Agent Type	Adj. Mag.	Adj. Freq.	Stabil.	Volatil.
MADDPG	0.0152	0.423	0.985	0.052
MADQN	0.0358	0.763	0.985	0.085
MADDPG+MADQN	0.0235	0.573	0.998	0.069
MADDPG+QMIX	0.0201	0.487	0.982	0.068
MADDPG+Rule	0.0127	0.340	1.000	0.050
MADQN+Rule	0.0339	0.771	0.996	0.083
QMIX	0.0243	0.553	0.972	0.075
Rule-Based	0.0114	0.312	0.944	0.024

As shown in Table 3, MADQN agents demonstrate the most aggressive pricing, exhibiting the highest adjustment magnitude (*0.0358*) and frequency (*0.763*). This indicates a reactive, and exploratory strategy. QMIX and mixed MADQN configurations also show increased responsiveness to these metrics. Alternatively, MADDPG agents are more conservative (*mean magnitude 0.0152*), with smoother and less volatile policy updates. As expected, rule-based agents have the lowest adjustment metrics due to their static design. Crucially, this means that high adaptability does not always lead to instability. Mixed strategies (such as MADDPG + Rule, MADDPG + MADQN) maintain high price stability (*>0.98*) despite frequent adjustments. This suggests that hybrid agents can balance dynamic pricing with coordinated stability. However, MADQN’s high volatility (*0.085*) together with high adaptability implies that aggressive pricing can cause instability. Rule-based agents are the most stable (*0.944-1.000*) and least volatile (*0.024*), but mostly lack adaptability.

These findings highlight a trade-off: adaptability improves market responsiveness but it can generate instability unless balanced by conservative agents. Hybrid agent configurations are promising for achieving dynamic pricing while maintaining stable.

This subsection compares the performance of MARL agents with rule-based pricing agents across all configurations. The evaluation focuses on mean revenue generation, and market share.

Figure 2 illustrates the market share captured at the final episode for each agent across experimental configurations, including 95% confidence intervals over each independent simulation run. This offers insight into long-term competitive dominance and strategic effectiveness.

To compare configurations statistically and isolate individual agent performance, Table 4 reports the mean return per agent, averaged across all episodes and eight independent simulation runs. This allows for a normalized comparison and highlights significant performance improvements over rule-based agents.

To assess performance differences in per-agent returns, the Wilcoxon signed-rank test was applied. MARL-only configurations (B, C, F) were tested against the rule-based baseline (A). Even though consistent improvements were observed, statistical significance was not achieved ($p = 0.125$). This was due to the small sample size (four agents). Future experiments with more agents could improve statistical power. Configurations with mixed types (D, E, G, H) were excluded from direct statistical testing.

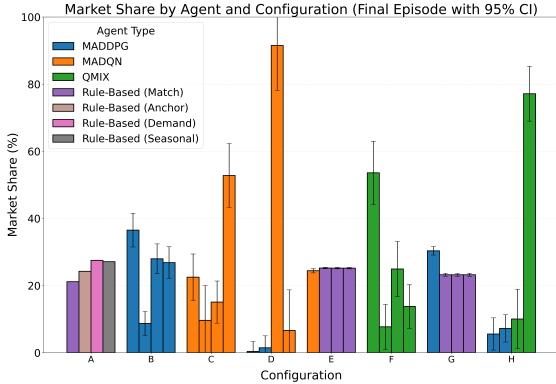


Figure 2: Final Episode Market Share per Agent across Configurations (with 95% CI).

Table 4: Mean agent return (\bar{R}) per configuration compared to Rule-Based baseline.

Configuration	\bar{R} (£)	Δ	Δ (%)
A - All Rule-Based	£22,817	—	—
B - All MADDPG	£89,861	£67,044	+293.8%
C - All MADQN	£997,669	£974,852	+4272.5%
D - MADDPG + MADQN	£709,224	£686,407	+3008.3%
E - MADQN + Rule-Based	£945,045	£922,229	+4041.9%
F - All QMIX	£393,121	£370,305	+1622.9%
G - One MADDPG	£75,734	£52,917	+231.9%
H - MADDPG + QMIX	£213,682	£190,865	+836.5%

Rule-based agents. achieve equitable market shares and highly stable pricing but generate substantially lower total revenue due to static pricing and limited responsiveness. Interestingly, in mixed configuration G (one MADDPG agent vs. three rule-based), total revenue modestly rises to £628,016, suggesting rule-based agents can constrain isolated agents and that a single MARL agent is not enough to disrupt a stable rule-based market.

MADQN. agents exhibit the most pronounced revenue dominance, generating just under £1 million in mean revenue through aggressive pricing. However, this comes with high price volatility and substantial market share fluctuations. This indicates significant short-term revenue generation but unstable competitive outcomes. This is probably due to a lack of coordinated pricing behaviour.

QMIX. agents consistently perform well in cooperative settings. They effectively coordinate pricing for strong revenue and sustained competitive positioning. Their performance deteriorates in heterogeneous configurations with learning assumptions that are conflicting. This highlights their strength in homogeneous or collaborative markets.

MADDPG. agents adopt a more cautious strategy. They achieve modest revenue and market share by prioritizing pricing stability over aggressive profit maximization. When combined with other agents (Configurations D and H), MADDPG contributes to dynamic

outcomes but rarely dominates, which suggests conservative and risk-averse learning dynamics.

Overall, MARL agents outperform rule-based strategies in revenue generation but introduce trade-offs in volatility and coordination complexity. MADQN effectively brings value but can destabilize the market. QMIX offers a strong middle ground for cooperative agents, while MADDPG provides (suboptimal) stable pricing. Even though less adaptive, rule-based agents remain competitive in mixed environments. This proves that they are valuable in systems that prioritize transparency and stability.

To assess how market dynamics influence agent effectiveness, we analyse three key categories of outcomes. First, we examine emergent market dynamics, including fairness, and market volatility, which reflect structural conditions in the simulated environment. Second, we evaluate coordination behaviour through metrics such as Nash equilibrium proximity, and price convergence. Finally, we assess agent performance in economic terms via efficiency indicators, such as revenue optimality gap, and welfare fairness. These categories together allow us to analyse how different configurations influence total performance in terms of competitiveness, coordination, and responsiveness.

Table 5 presents key indicators of market-level dynamics, coordination and performance. Fully rule-based environments (4x Rule) exhibit near-perfect fairness (0.9896 ± 0.0000), zero market share volatility, and low price instability. This forms a clear baseline for predictable dynamics. In contrast, configurations involving MADQN agents, exhibit significant volatility and lower fairness scores (such as 0.5844 ± 0.1625 for 4x MADQN). These patterns highlight a tendency for highly adaptive agents to disrupt equitable competition, introducing pricing instability. Hybrid configurations can either moderate or increase instability. For example, 1x MADQN + 3x Rule maintains high fairness (0.9991) and low volatility ($0.7pp$), indicating that the presence of a rule-based agent can constrain destabilizing behaviour. Alternatively, MADDPG + MADQN or MADDPG + QMIX configurations show lowered fairness and increased volatility. This suggests that hybridization does not necessarily stabilize dynamics without balance.

As shown in Table 5, strategic coordination varies considerably across configurations. Nash equilibrium proximity is highest in 1x MADDPG + 3x Rule (0.9999) and 2x MADDPG + 2x MADQN (0.8782). This indicates that hybrid configurations may encourage behaviour that seeks an equilibrium. This is potentially due to tension between adaptive and fixed strategies. Interestingly, 4x MADQN shows the lowest Nash equilibrium proximity (0.5788). This strengthens earlier findings that these agents focus on aggressive and divergent strategies that usually disrupt stability. Price convergence follows similar trends: 4x Rule and 1x MADDPG + 3x Rule reach the highest convergence levels (0.9761 and 0.9324), while 4x MADQN yields almost no convergence at all.

Table 5 examines whether dynamic strategies deliver economically meaningful improvements. Unsurprisingly, the lowest revenue optimality gap is observed in the rule-based configuration (0.0000). Note that this does not come with high absolute performance. In contrast, 4x MADQN reaches a high gap (0.7060), which suggests significant room between the actual and optimal pricing performance, even as it achieves the highest mean revenue overall. This

Table 5: Summary of emergent dynamics: Jain’s Fairness (JF) and Market Volatility (MV), coordination: Nash Equilibrium Proximity (NEP) and Price Convergence (PC), and performance: Revenue Optimality Gap (ROG) and Welfare Fairness (WF) metrics across agent configurations.

Agent Configuration	JF	MV (pp)	NEP	PC	ROG	WF
4x Rule (diverse strategies)	0.9896 ± 0.0000	0.0 ± 0.0	0.8502	0.9761	0.0000	0.9451
4x MADDPG	0.8819 ± 0.1032	9.5 ± 1.8	0.8639	0.8745	0.3957	0.9108
4x MADQN	0.5844 ± 0.1625	22.4 ± 5.2	0.5788	0.0131	0.7060	0.4383
2x MADDPG, 2x MADQN	0.5290 ± 0.1755	16.9 ± 12.2	0.8782	0.3698	0.7480	0.5894
1x MADQN, 3x Rule	0.9991 ± 0.0022	0.7 ± 0.4	0.7396	0.9237	0.9596	0.9908
4x QMIX	0.6953 ± 0.1570	17.5 ± 1.8	0.8071	0.0402	0.7272	0.5146
1x MADDPG, 3x Rule	0.9899 ± 0.0086	1.1 ± 0.7	0.9999	0.9324	0.5355	0.9921
2x MADDPG, 2x QMIX	0.6417 ± 0.2113	16.4 ± 8.3	0.8406	0.1329	0.5432	0.5279

confirms that aggressive pricing may exploit short-term gains without long-term efficiency. Finally, welfare fairness scores comply with earlier fairness metrics, with rule-based and hybrid configurations (such as 1x MADDPG + 3x Rule) maintaining high fairness. Conversely, MADQN and MADDPG + MADQN configurations rank lowest here, which suggests that price aggressiveness compromises not only fairness but also equitable welfare distribution.

These results highlight a fundamental feature in multi-agent learning environments. While adaptability can increase revenue, it often lowers market stability, fairness, and overall coordination. Rule-based and MADDPG agents deliver stable but less efficient outcomes. Conversely, MADQN excels in profit maximization at the cost of volatility and inequity. Hybrid configurations can strike a balance, but only when carefully configured and tuned. Agent interactions therefore shape not just outcomes but the very nature of the market itself. This makes configuration design a critical consideration in deploying such learning based pricing systems.

5 DISCUSSION

This section reflects on the key findings, examines how they relate to existing work, and considers the implications of the results. It also addresses limitations of the current study and touches on relevant ethical concerns.

5.1 Comparison with State-of-the-Art

Earlier studies on RL for pricing tend to focus on single-agent environments [20, 43]. These approaches fall short when it comes to capturing the interactions between different supply chain actors. This study expands on this by testing several MARL methods: MADDPG, MADQN, and QMIX inside a market simulation based on actual retail data. It further advances recent MARL pricing research [59, 61] through two contributions. First, it introduces hybrid agent populations, combining learning agents with rule-based strategies to reflect realistic market heterogeneity. Following, it evaluates performance using structural metrics such as fairness, coordination, and market stability, which remain under-explored in literature but are critical for practical deployment.

Among the tested models, MADQN stood out in terms of revenue generation. Its performance exceeded that of both static and single-agent baselines by a large margin. However, this level of gain came with costs which reflect similar patterns found in adversarial RL

research [38]. High revenues were often paired with instability and uneven market outcomes. In contrast, QMIX was better at promoting coordination among agents. Its behaviour aligns with findings in the cooperative MARL literature [36]. MADDPG took a more stable path, often finding a trade-off between stability and adaptability. This study is set apart from purely simulated work due to the integration of a real-world demand model, which was built using LightGBM and enriched with price elasticity estimates.

5.2 Interpretation of Findings

The experiments highlight a trade-off between adaptability, fairness, and stability. MADQN agents adjusted prices most frequently and captured the highest revenue, but introduced volatility and inequity. Rule-based agents, by contrast, provided stability and fairness but lacked responsiveness, leading to lower revenue. Mixed agent configurations performed best overall. For instance, MADDPG paired with QMIX or rule-based agents achieved moderate adaptability while maintaining fairness and coordination. This suggests that combining exploratory and conservative agents balances market responsiveness and stability. Finally, MARL agents did not always dominate in mixed-agent settings. When surrounded by rule-based agents, even highly adaptive agents were limited in influence. This underscores the need to consider market composition and real-world constraints when deploying MARL in practice.

5.3 Limitations

While the results provide valuable insights into the behaviour of MARL agents in dynamic pricing scenarios, several limitations must be acknowledged to contextualize the findings and guide future research. The following sections outline key areas where these constraints were most notable.

5.3.1 Realistic Price Elasticity. The demand model produced a near-zero elasticity ($\varepsilon = -0.072$), which denotes inelastic behaviour typical of giftware sales, where demand is relatively insensitive to price changes. However, this is a limit in generalizability. In elastic markets like travel, digital services, or groceries, consumers respond more strongly to price shifts. Several MARL agents, including MADQN and QMIX, exploited this inelasticity by raising prices across episodes, knowing demand would remain fairly stable. While optimal in this context, such behaviour would likely

be restricted in markets where elasticity burdens pricing power. This highlights the need for future simulations to use more elastic datasets or model heterogeneous consumer responses to avoid unrealistic, and loophole-driven strategies, ensuring that insights better reflect real-world dynamics.

5.3.2 Product Portfolio. Each agent handled a small, fixed group of products chosen from different clusters. While this setup allowed for controlled experiments, it does not reflect the complexity of actual supply chain environments. Real systems often involve large and nested product structures, cross-selling, and bundled pricing. Future work could explore agents that work over multiple levels of a catalogue and make combined decisions across related items.

5.3.3 Bias in Data. The dataset used for modelling comes from a single UK-based retailer, collected between 2009 and 2011. Even though it has a large transaction volume and structured format, making it suitable for simulation, several limitations affect generalizability. Customer behaviours, such as response to discounting, average order size, and purchasing frequency, could reflect UK consumer behaviour in the early 2010s. This feature heavily is shaped by local economic conditions and retail habits of that period. These patterns may be significantly different in other regions (such as North America or Asia) or in today's e-commerce environments, where consumer expectations, promotional sensitivity, and platform dynamics have significantly changed. Similarly, seasonality patterns such as holiday spikes in demand may not align with global retail volatility and changes. As a result, the dataset supports controlled experimentation, but future studies should consider more recent and regionally diverse data sources, which would enhance realism and generalizability.

5.3.4 Reproducibility and Scalability. Reproducibility was a core principle in this study. This was ensured through deterministic data splits and consistent training procedures, but some variability in outcomes was observed due to inherent stochasticity of RL. Certain agents, such as MADQN and QMIX, occasionally exhibited different convergence behaviours across independent runs even with identical initial settings. Although this makes exact replication of individual trajectories challenging, average performance across runs was stable. These same agent types were particularly resource intensive, with training times of up to 12 hours per configuration on optimized 8-core CPU infrastructure (Snellius), reflecting the computational complexity discussed in Section 2.3.1. This limits scalability in larger markets or real-time ERP contexts. The current environment did not yet include other factors such as supply limits, delayed fulfilment, or inventory-sensitive pricing. Future work should explore how such constraints shape agent behaviour and influence pricing decisions.

6 CONCLUSION

This research explored how MARL can enhance dynamic pricing strategies in supply chains. This was done by addressing the gap between static pricing logic together with the need for adaptive, decentralized decision-making under market uncertainty. By benchmarking MADDPG, MADQN, and QMIX against rule-based pricing

agents in a realistic market simulation, the study aimed to analyse and understand the trade-off between adaptability, revenue performance, fairness, and stability.

The main research question asked how MARL can improve dynamic pricing compared to traditional models while accounting for interactions between agents. The results showed that MARL, and MADQN in particular, significantly outperformed rule-based agents in terms of revenue, confirming the advantage of data-driven adaptation. However, this gain comes with a cost: higher volatility, reduced fairness, and coordination difficulties. This study showed that:

- (1) MADQN was the most adaptive and aggressive, but destabilizing.
- (2) Rule-based agents offered stability but were outperformed in revenue.
- (3) Market dynamics shaped agent effectiveness significantly, with hybrid configurations balancing the trade-offs best.

Concluding, while MARL has clear advantages, its effectiveness depends on the design of agent configurations and the underlying market conditions. Its benefits depend on how and where it is used, including algorithm setup and system limitations. The inelastic nature of demand in the dataset also limits generalizability to other markets. Nonetheless, the observed agent behaviours and trade-offs offer insights into how MARL systems behave under stable demand conditions. In particular, this study suggests that MARL is most beneficial when demand is predictable, competition is strategic, and price setting is flexible.

This study demonstrates that MARL can be practically integrated with ERP-relevant demand modelling. It also shows the importance of design considerations for dynamic pricing agents; particularly in low-elastic or regulated market segments. It also highlights that the environment and strategies of others strongly shape the performance of adaptive agents; an insight that is often missing from prior single-agent literature.

Future research should focus on two promising directions. First, it should incorporate products with elastic demand, which would help evaluate pricing responsiveness in more sensitive markets. Second, it should explore scalable MARL architectures, such as graph-based or hierarchical methods. These could improve coordination and training efficiency; especially in larger supply chains. These steps would help move MARL from experimental validation towards deployment readiness in business enterprise systems.

REFERENCES

- [1] 2025. Snellius: de Nationale Supercomputer. <https://www.surf.nl/snellijs-de-nationale-supercomputer>. Accessed: 2025-05-16.
- [2] Khaled Abdalgader, Atheer A Matroud, and Khaled Hossin. 2024. Experimental study on short-text clustering using transformer-based semantic similarity measure. *PeerJ Computer Science* 10 (2024), e2078. doi:10.7717/peerj.cs.2078
- [3] Anthony B Atkinson. 1999. The contributions of Amartya Sen to welfare economics. *The Scandinavian Journal of Economics* 101, 2 (1999), 173–190. doi:10.1111/1467-9442.00151
- [4] Kasun Bandara, Christoph Bergmeir, and Slawek Smyl. 2020. Forecasting across time series databases using recurrent neural networks on groups of similar series: A clustering approach. *Expert Systems with Applications* 140 (2020), 112896. doi:10.1016/j.eswa.2019.112896
- [5] Souhaib Ben Taieb, Gianluca Bontempi, Amir F Atiya, and Antti Sorjamaa. 2012. A review and comparison of strategies for multi-step ahead time series forecasting based on the NN5 forecasting competition. *Expert Systems with Applications* 39, 8 (2012), 7067–7083. doi:10.1016/j.eswa.2012.01.039

- [6] Mauricio F Blos and Paulo E Miyagi. 2015. Modeling the supply chain disruptions: A study based on the supply chain interdependencies. *IFAC-PapersOnLine* 48, 3 (2015), 2053–2058. doi:10.1016/j.ifacol.2015.06.391
- [7] Lucian Busoniu, Robert Babuska, and Bart De Schutter. 2008. A comprehensive survey of multiagent reinforcement learning. *IEEE Transactions on Systems, Man, and Cybernetics, Part C (Applications and Reviews)* 38, 2 (2008), 156–172. doi:10.1109/TSMCC.2007.913919
- [8] Gérard P Cachon and Prinna Feldman. 2010. Dynamic versus static pricing in the presence of strategic consumers. *The Wharton School, University of Pennsylvania* (2010).
- [9] Lidia Ceriani and Paolo Verme. 2012. The origins of the Gini index: extracts from *Variabilità e Mutabilità* (1912) by Corrado Gini. *The Journal of Economic Inequality* 10 (2012), 421–443. doi:10.1007/s10888-011-9188-x
- [10] Daqing Chen. 2012. Online Retail II [Dataset]. <https://archive.ics.uci.edu/dataset/502/online+retail+ii>. doi:10.24432/C5CG6D
- [11] Daqing Chen, Kun Guo, and George Ubakamna. 2015. Predicting customer profitability over time based on RFM time series. *International Journal of Business Forecasting and Marketing Intelligence* 2, 1 (2015), 1–18. doi:10.1504/IJBFMI.2015.075325
- [12] Daqing Chen, Sai Laing Sain, and Kun Guo. 2012. Data mining for the online retail industry: A case study of RFM model-based customer segmentation using data mining. *Journal of Database Marketing & Customer Strategy Management* 19, 3 (2012), 197–208. doi:10.1057/dbm.2012.17
- [13] Le Chen, Alan Mislove, and Christo Wilson. 2016. An Empirical Analysis of Algorithmic Pricing on Amazon Marketplace. In *Proceedings of the 25th International Conference on World Wide Web*. ACM, 1339–1349. doi:10.1145/2872427.2883089
- [14] Yixian Chen, Prakhar Mehrotra, Nitin Kishore Sai Samala, Kamilia Ahmadi, Viresh Jivane, Linsey Pang, Monika Shrivastav, Nate Lyman, and Scott Pleiman. 2021. A Multiobjective Optimization for Clearance in Walmart Brick-and-Mortar Stores. *INFORMS Journal on Applied Analytics* 51, 1 (2021), 76–89. doi:10.1287/inte.2020.1065
- [15] Pritom Das, Tamanna Pervin, Biswanath Bhattacharjee, Md Razaul Karim, Nasrin Sultan, Md Sayham Khan, Md Afjal Hosien, and FNU Kamruzzaman. 2024. Optimizing real-time dynamic pricing strategies in retail and e-commerce using machine learning models. *The American Journal of Engineering and Technology* 6, 12 (2024), 163–177. doi:10.3754/tajet/VOLUME06Issue12-15
- [16] Junfeng Dong, Beilei Rao, Yu Liu, Li Jiang, Wenxing Lu, and Qiang Guo. 2019. Pricing strategies for different periods during subsequent selling season for seasonal products. *IEEE Access* 8 (2019), 39479–39490. doi:10.1109/ACCESS.2019.2953284
- [17] Raouya El Youbi, Fayçal Messaoudi, and Manal Loukili. 2023. Machine learning-driven dynamic pricing strategies in E-commerce. In *2023 14th International Conference on Information and Communication Systems (ICICS)*. IEEE, 1–5. doi:10.1109/ICICS60529.2023.10330541
- [18] Lawrence Emma. 2024. Enterprise Resource Planning (ERP) Systems for Streamlining Organizational Processes. *Unpublished Manuscript* (2024). https://www.researchgate.net/publication/386382658_Enterprise_Resource_Planning_ERP_Systems_for_Streamlining_Organizational_Processes
- [19] Jakob N. Foerster, Yannis M. Assael, Nando de Freitas, and Shimon Whiteson. 2016. Learning to Communicate with Deep Multi-Agent Reinforcement Learning. In *Advances in Neural Information Processing Systems*, Vol. 29. Curran Associates, Inc., 2137–2145. <https://proceedings.neurips.cc/paper/2016/file/c7635bfd99248a2cdef8249ef7bfbe4-Paper.pdf> <https://arxiv.org/abs/1605.06676>.
- [20] Kallirroi Georgila, Claire Nelson, and David Traum. 2014. Single-agent vs. multi-agent techniques for concurrent reinforcement learning of negotiation dialogue policies. In *Proceedings of the 52nd Annual Meeting of the Association for Computational Linguistics (Volume 1: Long Papers)*. Association for Computational Linguistics, Baltimore, Maryland, 500–510. doi:10.3115/v1/P14-1047
- [21] Rajan Gupta and Chaitanya Pathak. 2014. A machine learning framework for predicting purchase by online customers based on dynamic pricing. *Procedia Computer Science* 36 (2014), 599–605. doi:10.1016/j.procs.2014.09.060
- [22] Adnène Hajji, Robert Pellerin, Pierre-Majorique Léger, Ali Gharbi, and Gilbert Babin. 2012. Dynamic pricing models for ERP systems under network externality. *International Journal of Production Economics* 135, 2 (2012), 708–715. doi:10.1016/j.ijpe.2011.10.004
- [23] Ye Han, Xuefei Zhang, Jian Zhang, Qimei Cui, Shuo Wang, and Zhu Han. 2019. Multi-agent reinforcement learning enabling dynamic pricing policy for charging station operators. In *2019 IEEE Global Communications Conference (GLOBECOM)*. IEEE, 1–6. doi:10.1109/GLOBECOM38437.2019.9013999
- [24] Walter A. Hendricks and Kate W. Robey. 1936. The Sampling Distribution of the Coefficient of Variation. *Annals of Mathematical Statistics* 7, 3 (1936), 129–132. doi:10.1214/aoms/1177732503
- [25] Jacob Hilton, Jie Tang, and John Schulman. 2023. Scaling laws for single-agent reinforcement learning. *arXiv preprint arXiv:2301.13442* (2023). doi:10.48550/arXiv.2301.13442
- [26] Andreas Hinterhuber. 2004. Towards value-based pricing—An integrative framework for decision making. *Industrial Marketing Management* 33, 8 (2004), 765–778. doi:10.1016/j.indmarman.2003.10.006
- [27] Samuel B Hwang and Sungho Kim. 2006. Dynamic pricing algorithm for E-Commerce. In *Advances in Systems, Computing Sciences and Software Engineering: Proceedings of SCSS05*. Springer, 149–155. doi:10.1007/1-4020-5263-4_24
- [28] Rajendra K Jain, Dah-Ming W Chiu, and William R Hawe. 1984. *A Quantitative Measure of Fairness and Discrimination*. Technical Report TR-301. Digital Equipment Corporation, Hudson, MA. <https://www.cs.wustl.edu/~jain/papers/ftp/fairness.pdf>
- [29] Vipul Jain and Lyes Benyoucef. 2008. Managing long supply chain networks: some emerging issues and challenges. *Journal of Manufacturing Technology Management* 19, 4 (2008), 469–496. doi:10.1108/17410380810869923
- [30] Guolin Ke, Qi Meng, Thomas Finley, Taifeng Wang, Wei Chen, Weidong Ma, Qiwei Ye, and Tie-Yan Liu. 2017. LightGBM: A Highly Efficient Gradient Boosting Decision Tree. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc., 3146–3154. https://proceedings.neurips.cc/paper_files/paper/2017/file/6449f44a102fde848669bdd9eb676fa-Paper.pdf
- [31] Byung-Gook Kim, Yu Zhang, Mihaela Van Der Schaar, and Jang-Won Lee. 2015. Dynamic pricing and energy consumption scheduling with reinforcement learning. *IEEE Transactions on Smart Grid* 7, 5 (2015), 2187–2198. doi:10.1109/TSG.2015.2495145
- [32] Vijay R. Konda and John N. Tsitsiklis. 2003. On actor-critic algorithms. *SIAM Journal on Control and Optimization* 42, 4 (2003), 1143–1166. doi:10.1137/S0363012901385691
- [33] Oh Byung Kwon and J.J. Lee. 2001. A multi-agent intelligent system for efficient ERP maintenance. *Expert Systems with Applications* 21, 4 (2001), 191–202. doi:10.1016/S0957-4174(01)00039-2
- [34] Derek Li, Andrew Jacobsen, and Adam White. 2021. Revisiting Experience Replay in Non-Stationary Environments. In *Proceedings of the Adaptive and Learning Agents Workshop (ALA)*. https://ala2021.vub.ac.be/papers/ALA2021_paper_51.pdf
- [35] Le Li, Xiao Lin, Rudy R Negenborn, and Bart De Schutter. 2015. Pricing intermodal freight transport services: A cost-plus-pricing strategy. In *Computational Logistics: 6th International Conference, ICCL 2015, Delft, The Netherlands, September 23–25, 2015, Proceedings*. Springer, 541–556. doi:10.1007/978-3-319-24264-4_37
- [36] Jiaxin Liang, Haotian Miao, Kai Li, Jianheng Tan, Xi Wang, Rui Luo, and Yueqiu Jiang. 2025. A Review of Multi-Agent Reinforcement Learning Algorithms. *Electronics* 14, 4 (2025), 820. doi:10.3390/electronics14040820
- [37] Jiaxi Liu, Yidong Zhang, Xiaoqing Wang, Yuming Deng, and Xingyu Wu. 2019. Dynamic Pricing on E-commerce Platform with Deep Reinforcement Learning: A Field Experiment. *arXiv preprint arXiv:1912.02572* (2019). doi:10.48550/arXiv.1912.02572
- [38] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Actor-Critic for Mixed Cooperative-Competitive Environments. In *Advances in Neural Information Processing Systems*, Vol. 30. Curran Associates, Inc., 6379–6390. <https://proceedings.neurips.cc/paper/2017/file/68a9750337a418a86fe06c1991a1d64c-Paper.pdf>
- [39] Ryan Lowe, Yi Wu, Aviv Tamar, Jean Harb, Pieter Abbeel, and Igor Mordatch. 2017. Multi-Agent Deep Deterministic Policy Gradient (MADDPG) - GitHub repository. <https://github.com/openai/maddpg> Archived codebase provided as-is, based on the NeurIPS 2017 paper.
- [40] Arun K. Menon. 2024. Integrating Pricing Optimization Models with ERP Systems to Enhance Profitability in U.S. E-commerce Supply Chains. *Global Journal of Engineering and Technology Advances* 20, 2 (2024), 242–255. doi:10.30574/gjeta.2016.2.0151 Open access under CC BY 4.0.
- [41] Volodymyr Mnih, Koray Kavukcuoglu, David Silver, Andrei A Rusu, Joel Veness, Marc G Bellemare, Alex Graves, Martin Riedmiller, Andreas K Fidjeland, Georg Ostrovski, Stig Petersen, Charles Beattie, Amir Sadik, Ioannis Antonoglou, Helen King, Dharshan Kumaran, Daan Wierstra, Shane Legg, and Demis Hassabis. 2015. Human-level control through deep reinforcement learning. *Nature* 518, 7540 (2015), 529–533. doi:10.1038/nature14236
- [42] John F Nash Jr. 1950. Equilibrium Points in N-Person Games. *Proceedings of the National Academy of Sciences* 36, 1 (1950), 48–49. doi:10.1073/pnas.36.1.48
- [43] Gonçalo Neto. 2005. From Single-Agent to Multi-Agent Reinforcement Learning: Foundational Concepts and Methods. <https://users.cs.utah.edu/~tch/CS6380/resources/Neto-2005-RL-MAS-Tutorial.pdf> Learning Theory Course 2005; 2.
- [44] Vincent R Nijs, Shuba Srinivasan, and Koen Pauwels. 2007. Retail-price drivers and retailer profits. *Marketing Science* 26, 4 (2007), 473–487. doi:10.1287/mksc.1060.0205
- [45] Tabish Rashid, Mikayel Samvelyan, Christian Schroeder de Witt, Gregory Farquhar, Jakob Foerster, and Shimon Whiteson. 2018. QMIX: Monotonic Value Function Factorisation for Deep Multi-Agent Reinforcement Learning. In *Proceedings of the 35th International Conference on Machine Learning*. PMLR, 4295–4304. doi:10.48550/arXiv.1803.11485
- [46] Nils Reimers and Iryna Gurevych. 2019. Sentence-BERT: Sentence Embeddings using Siamese BERT-Networks. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*. Association for Computational Linguistics, 3982–3992. doi:10.18653/v1/D19-1410

- [47] Lei Ren, Xiaoyang Fan, Jin Cui, Zhen Shen, Yisheng Lv, and Gang Xiong. 2022. A multi-agent reinforcement learning method with route recorders for vehicle routing in supply chain management. *IEEE Transactions on Intelligent Transportation Systems* 23, 9 (2022), 16410–16420. doi:10.1109/TITS.2022.3150151
- [48] Melrose Roderick, James MacGlashan, and Stefanie Tellex. 2017. Implementing the deep q-network. *arXiv preprint arXiv:1711.07478* (2017). doi:10.48550/arXiv.1711.07478
- [49] Shahriar Shafiee and Erkan Topal. 2010. An overview of global gold market and gold price forecasting. *Resources Policy* 35, 3 (2010), 178–189. doi:10.1016/j.resourpol.2010.05.004
- [50] Ravid Schwartz-Ziv and Amitai Armon. 2022. Tabular Data: Deep Learning is Not All You Need. *Information Fusion* 81 (2022), 84–90. doi:10.1016/j.inffus.2021.11.011
- [51] Rina Singh, Jeffrey A. Graves, Douglas A. Talbert, and William Eberle. 2018. Prefix and suffix sequential pattern mining. In *Advances in Data Mining. Applications and Theoretical Aspects*. Springer, 309–324. doi:10.1007/978-3-319-95786-9_24
- [52] Hussein Kamaldeen Smith. 2024. Dynamic Pricing and E-supply Chain Coordination: Effects on Inventory Optimization and Profit Margins. *Unpublished Manuscript* (2024). https://www.researchgate.net/publication/384805839_Dynamic_Pricing_and_E-supply_Chain_Coordination_Effects_on_Inventory_Optimization_and_Profit_Margins
- [53] Bo Sun, Hossein Nekouyan Jazi, Xiaoqi Tan, and Raouf Boutaba. 2024. Static Pricing for Online Selection Problem and its Variants. *arXiv preprint arXiv:2410.07378* (2024). doi:10.48550/arXiv.2410.07378
- [54] J Michael Tarn, David C Yen, and Marcus Beaumont. 2002. Exploring the rationales for ERP and SCM integration. *Industrial Management & Data Systems* 102, 1 (2002), 26–34. doi:10.1108/02635570210414631
- [55] Qiaochu Wang, Yan Huang, Param Vir Singh, and Kannan Srinivasan. 2023. Algorithms, Artificial Intelligence and Simple Rule Based Pricing. *SSRN Electronic Journal* (2023). doi:10.2139/ssrn.4144905
- [56] Sherry Shi Wang and Ralf Van Der Lans. 2018. Modeling gift choice: The effect of uncertainty on price sensitivity. *Journal of Marketing Research* 55, 4 (2018), 524–540. doi:10.1509/jmr.16.0453
- [57] Christopher JCH Watkins and Peter Dayan. 1992. Q-learning. *Machine Learning* 8, 3 (1992), 279–292. doi:10.1007/BF00992698
- [58] Rafal Weron. 2014. Electricity price forecasting: A review of the state-of-the-art with a look into the future. *International Journal of Forecasting* 30, 4 (2014), 1030–1081. doi:10.1016/j.ijforecast.2014.08.008
- [59] Annie Wong, Thomas Bäck, Anna V Kononova, and Aske Plaat. 2023. Deep multiagent reinforcement learning: Challenges and directions. *Artificial Intelligence Review* 56, 6 (2023), 5023–5056. doi:10.1007/s10462-022-10299-x
- [60] Tony Wu, Anthony D Joseph, and Stuart J Russell. 2016. *Automated Pricing Agents in the On-Demand Economy*. Master's thesis. University of California, Berkeley. <https://www2.eecs.berkeley.edu/Pubs/TechRpts/2016/EECS-2016-57.html>
- [61] Ziyuan Zhou, Guanjun Liu, and Ying Tang. 2023. Multi-agent reinforcement learning: Methods, applications, visionary prospects, and challenges. *arXiv preprint arXiv:2305.10091* (2023). doi:10.48550/arXiv.2305.10091

Appendices

A DATASET DESCRIPTION

Table 6 provides an overview of the Online Retail II dataset used in this study, including the column names, data types, and brief descriptions. This dataset contains 1,067,371 transactions recorded between December 2009 and December 2011 by a UK-based online retailer.

Table Scope: While the full dataset includes both B2B and B2C transactions, only B2B transactions (i.e., those with a valid Customer ID) were retained for modeling purposes. All exploratory statistics and preprocessing steps referenced in the methodology are derived from this filtered subset unless stated otherwise.

Column Name	Data Type	Description
InvoiceNo	String	Unique invoice number. Cancellations are indicated by a prefix 'C'.
StockCode	String	Unique product code.
Description	String	Text description of the product.
Quantity	Integer	Number of units sold (can be negative for returns).
InvoiceDate	Datetime	Timestamp of the transaction (down to minute precision).
UnitPrice	Float	Price per unit of product (in GBP).
CustomerID	Integer	Unique identifier for registered customers. Missing for anonymous retail purchases.
Country	String	Country of the customer.

Table 6: Overview of columns in the Online Retail II dataset.

A total of 5,243 unique products and 5,942 unique customers were identified in the full dataset. After filtering for valid B2B transactions, 77.2% of rows remained. Further preprocessing steps, such as the removal of returns (negative quantities), transactions with zero or negative prices, and aggregation at weekly intervals, are described in the main text (see Section 3.2).

B EXPLORATORY DATA ANALYSIS DETAILS

The following list summarizes key statistics identified in the EDA. All values are based on the dataset filtered for standard products (5-digit stock codes) unless stated otherwise.

- 92.1% of transactions originate from UK customers.
- 22.8% of transactions are from anonymous customers (without Customer ID).
- Non-registered customer prices are on average 55.7% higher; some product categories (e.g., mugs) show 142.3% higher pricing.
- The dataset contains 5,243 unique products and 5,942 registered customers.
- 1.7% of transactions are cancellations, 2.0% are returns.
- Transaction volume spikes during calendar weeks 47–52 (holiday season), with Thursdays and midday showing peak activity.

B.1 Statistical Formulas Used in EDA

Metric	Formula	Result / Interpretation
Volume Discount Calculation	$\text{Discount}_{\text{bucket}} = \frac{P_{\text{prev}} - P_{\text{curr}}}{P_{\text{prev}}} \times 100\%$	Prices dropped by 58.5% from the 1–10 item bucket to the 10–50 item bucket, confirming strong volume-based pricing incentives.
Price Volatility (Coefficient of Variation)	$CV = \frac{\sigma}{\mu} = \frac{\left(\frac{1}{n} \sum_{t=1}^n (V_t - \bar{V})^2\right)^{1/2}}{\bar{V}}$	Average CV is 0.39; 0.2% of products exceed $CV > 2.0$, indicating high promotional volatility.
Customer Type Price Differential	$\text{Price Diff}_{\%} = \frac{P_{\text{without ID}} - P_{\text{with ID}}}{P_{\text{with ID}}} \times 100\%$	Non-registered customers pay 55.7% more on average; the MUG category has the largest markup at 142.3%.
Price Ratio (Customer Type)	$\text{Price Ratio} = \frac{P_{\text{without ID}}}{P_{\text{with ID}}}$	Ratios peak at 1.78× during promotional months (e.g., Dec 2011), highlighting potential temporal discrimination.
Category Price Differential	$\text{Markup}_{\text{cat}} = \left(\frac{P_{\text{cat,without ID}}}{P_{\text{cat,with ID}}} - 1 \right) \times 100\%$	MUG products show the highest markup (142.3%), followed by CUSHIONS (69.1%) and LIGHTS (66.3%).
Quantity-Based Pricing Model	$P(q) = P_{\text{base}} \cdot f(q) \cdot g(\text{CustomerType})$	Model shows $g(\text{Without ID}) \approx 1.56 \cdot g(\text{With ID})$, confirming segmentation by customer type.

Table 7: Overview of key pricing-related formulas used in EDA with corresponding insights.

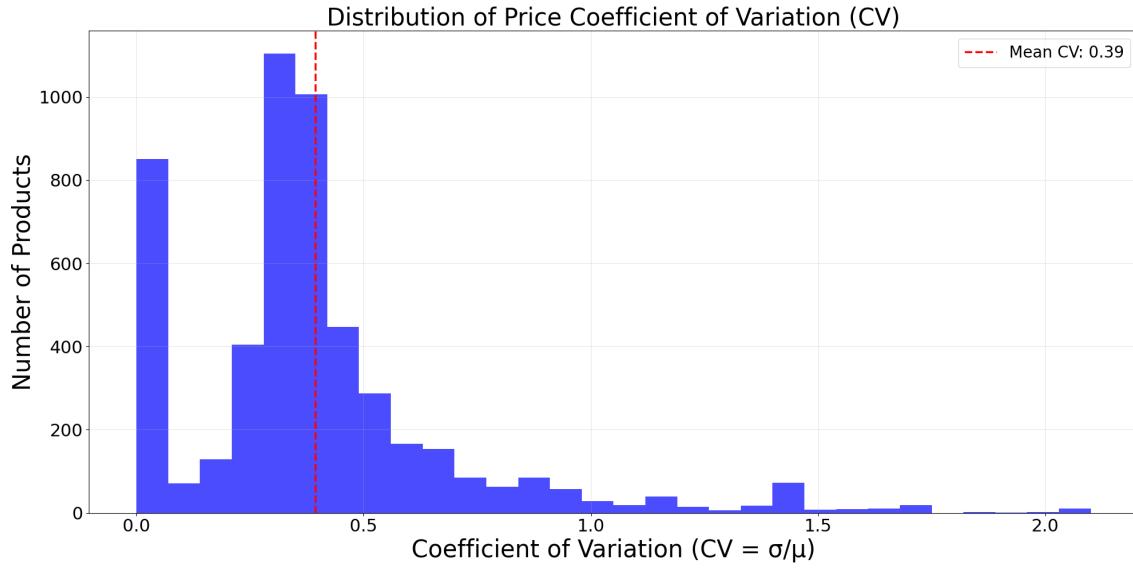


Figure 3: Distribution of price volatility measured by coefficient of variation (CV).

C FORMULAS USED IN PREPROCESSING

The following formulas were used to derive engineered features during preprocessing. These transformations aim to capture temporal trends, volatility, and relative price positioning across product categories.

Feature	Formula	Interpretation
Quantity Rolling Mean (QRM)	$QRM_k(t) = \frac{1}{k} \sum_{i=1}^k Q_{t-i}$	Captures average quantity sold over the previous k weeks, used for lag-based demand modeling.
Trend Indicator	$\text{Trend}_t = QRM_4(t) - QRM_2(t)$	Compares medium-term vs short-term average demand to measure directional change.
Acceleration	$\text{Acceleration}_t = Q_{t-1} - QRM_2(t)$	Measures deviation from short-term average, capturing momentum.
Volatility (Rolling Std. Dev.)	$\text{Volatility}_t = \sqrt{\frac{1}{k} \sum_{i=1}^k (Q_{t-i} - \bar{Q})^2}$	Estimates fluctuation in recent weekly demand.
Price vs Category Average (PVC)	$PVC_{Avg} = \frac{\text{Price}}{\mathbb{E}[\text{Price}_{\text{cluster}}]}$	Measures a product's price relative to the average of its semantic cluster (e.g., mugs).

Table 8: Derived feature formulas used in the preprocessing pipeline.

Variable Definitions: Q_{t-i} denotes the quantity sold i weeks before week t ; \bar{Q} is the mean quantity over the last k weeks; $\mathbb{E}[\text{Price}_{\text{cluster}}]$ refers to the average price of all products within a semantically clustered product group (e.g., mugs, candles); and Price is the observed price of the product.

D EVALUATION METRICS FORMULAS

The following table formally defines the evaluation metrics used to assess MARL agent performance in dynamic pricing environments.

Metric	Formula	Interpretation
Revenue per Agent	$R_{agent} = \sum_{t=1}^T P_t \cdot Q_t$	Total revenue accumulated by an agent over T time steps, where P_t is the price and Q_t is the sold quantity at week t .
Nash Equilibrium Proximity	$\Delta NE = 1 - \min\left(1, \overline{\Delta P} \cdot 10\right)$	Measures average absolute price change ratio over recent periods. Closer to 1 implies proximity to equilibrium.
Optimality Gap	$\text{Gap} = \frac{R_{max} - R_{agent}}{R_{max}}$	Relative performance gap between an agent's revenue and the observed maximum. Lower is better.
Gini Coefficient	$Gini = \frac{\sum_{i=1}^N \sum_{j=1}^N R_i - R_j }{2N \sum_{i=1}^N R_i}$	Measures inequality in revenue distribution across agents. A value of 0 indicates perfect equality; values approaching 1 reflect higher inequality.
Social Welfare	$SW = \sum_{i=1}^N R_i \cdot (1 - Gini)$	Aggregate market revenue adjusted by fairness (Gini coefficient), rewarding both efficiency and equity.
Jain's Fairness Index	$J = \frac{(\sum_{i=1}^N R_i)^2}{N \cdot \sum_{i=1}^N R_i^2}$	Evaluates fairness of revenue distribution across agents, ranging from $1/N$ (worst) to 1 (best).
Market Share Evolution	$MS_{i,t} = \frac{R_{i,t}}{\sum_{j=1}^N R_{j,t}}$	Agent i 's share of total market revenue at time t . Tracks competitive dynamics.
Price Volatility	$\text{MeanAbsChange} = \frac{1}{T} \sum_{t=2}^T \left \frac{P_t - P_{t-1}}{P_{t-1}} \right $	Mean absolute price change over time, complemented by standard deviation and max change per agent-product pair.
Price Convergence	$\text{Convergence} = 1 - \frac{\sigma_P}{\max(P)}$	Measures how closely agent prices align in the final simulation period. A value close to 1 indicates strong convergence (low price dispersion), while values near 0 indicate significant divergence in pricing strategies.
Adjustment Magnitude	$\text{AdjMag} = \frac{1}{T-1} \sum_{t=2}^T \left \frac{P_t - P_{t-1}}{P_{t-1}} \right $	Mean absolute percentage change in price between consecutive weeks. Captures how intensely agents adjust their prices over time.
Adjustment Frequency	$\text{AdjFreq} = \frac{1}{T-1} \sum_{t=2}^T \mathbb{1} \left[\left \frac{P_t - P_{t-1}}{P_{t-1}} \right > \tau \right]$	Proportion of weeks with a price change exceeding 1% ($\tau = 0.01$). Reflects how frequently agents make meaningful pricing updates.

Table 9: Evaluation metric formulas used for performance assessment of MARL agents.

Variable Definitions:

- P_t : Price set by the agent at time step t .
- Q_t : Quantity sold by the agent at time step t .
- R_i : Revenue of agent i over the evaluation period.
- $R_{i,t}$: Revenue of agent i at time step t .
- R_{max} : Maximum observed revenue achieved by any agent in the experiment.
- μ_P : Mean of the price series over time.
- σ_P : Standard deviation of the price series over time.
- $\overline{\Delta P}$: Average absolute percentage change in price between consecutive time steps.
- R_j : Revenue of agent j over the evaluation period (paired comparison with R_i for Gini calculation).
- $\Delta R, \Delta P$: Changes in revenue and price between consecutive periods or episodes.
- N : Total number of competing agents in the simulation.
- $\sigma(\Delta P_{products})$: Standard deviation of price changes across an agent's product portfolio.
- $\mathbb{1}[\cdot]$: Indicator function that equals 1 if the condition is true and 0 otherwise.
- τ : Threshold for detecting significant price changes, set to 0.01 (i.e., 1%).

E MARL AGENT ARCHITECTURE DIAGRAMS

The following diagrams provide an exact overview of the operation of MADDPG, MADQN and QMIX agents.

E.1 MADDPG Architecture

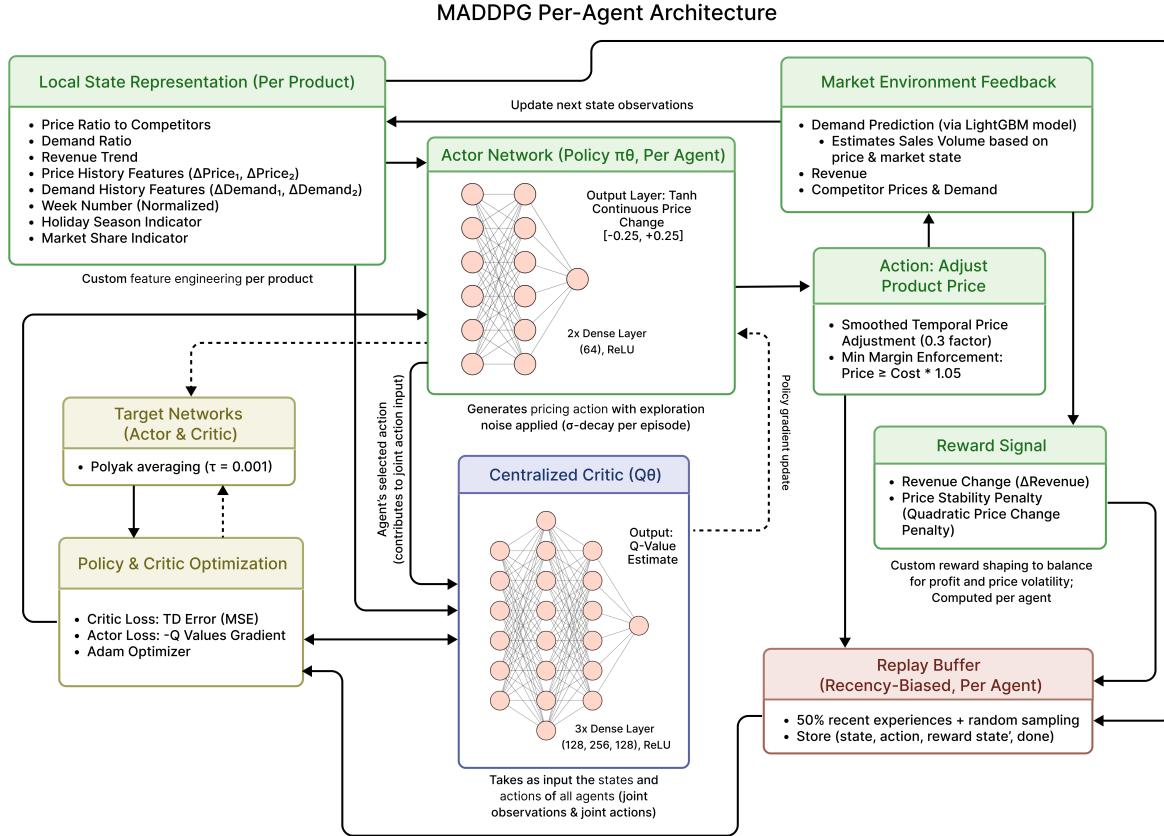


Figure 4: MADDPG Per-Agent Architecture. Each agent selects continuous price changes using a local Actor Network informed by engineered state features (such as price ratios, demand trends, market share). Actions are smoothed and constrained, then evaluated by a centralized Critic receiving joint states and actions. Learning is stabilized via a replay buffer (with recency bias), a TD error-based Critic loss, and Polyak-averaged target networks. The reward function balances revenue gains with a quadratic penalty on price instability, supporting coordinated yet adaptive pricing behaviour.

E.2 MADQN Architecture

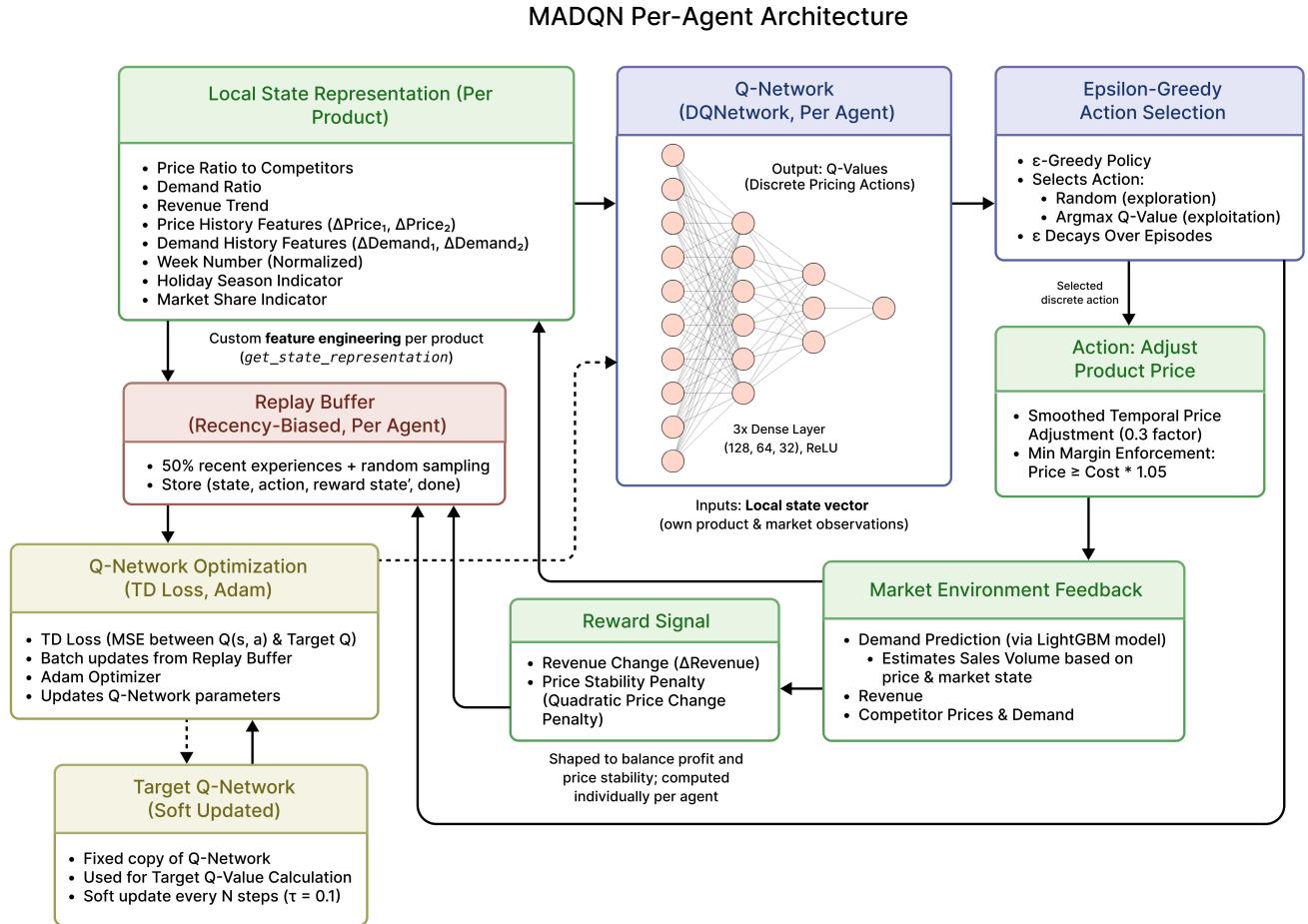


Figure 5: MADQN Per-Agent Architecture. Each agent independently learns discrete pricing actions through a local Q-Network informed by engineered state features. Actions are selected using an ϵ -greedy policy, then smoothed and constrained to enforce profitability. The agent receives individualized reward signals combining revenue change and a quadratic penalty on price instability. Learning is stabilized via a recency-biased replay buffer, TD loss optimization, and soft target network updates.

E.3 QMIX Architecture

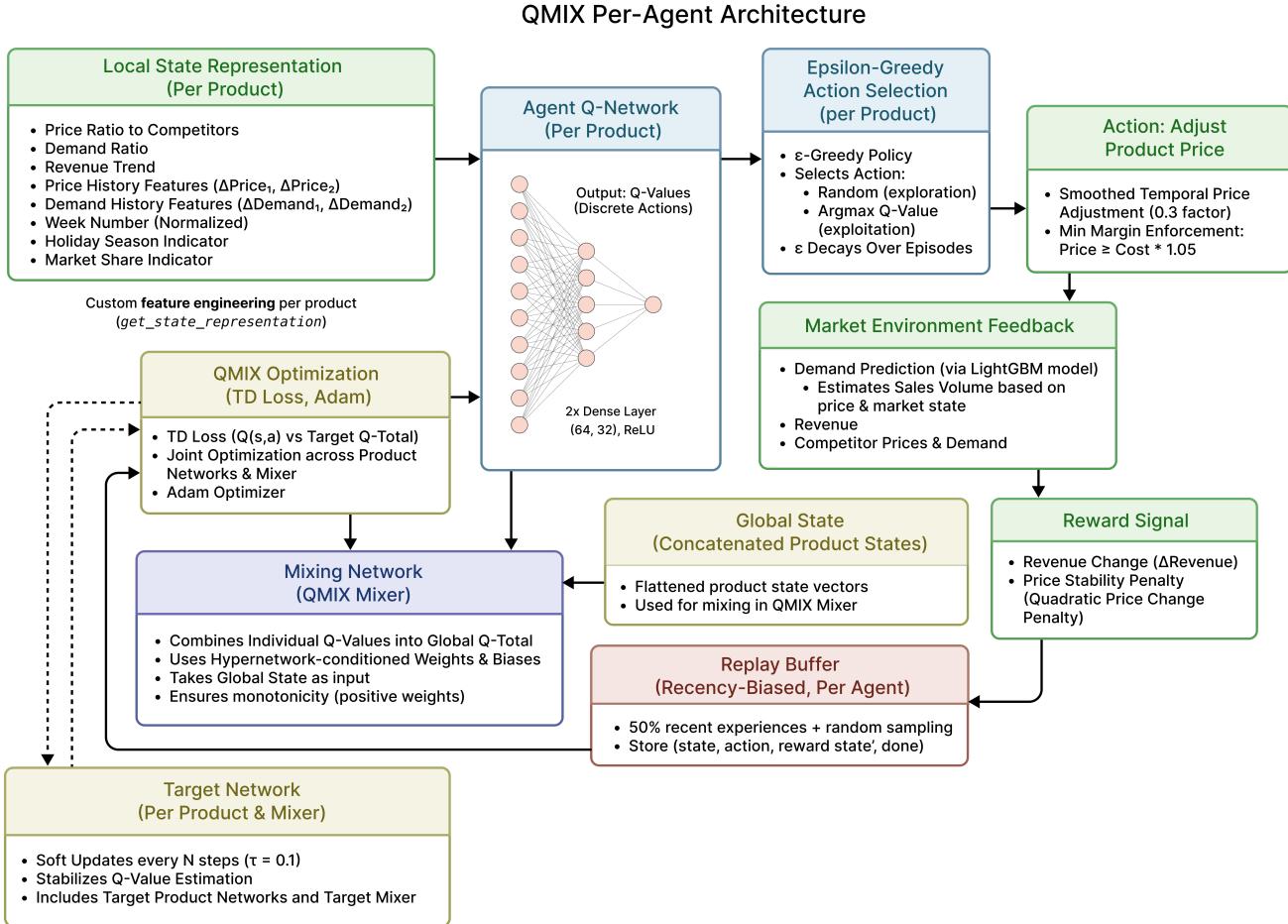


Figure 6: QMIX Per-Agent Architecture. Each pricing agent uses a local Q-Network to learn discrete price adjustments based on engineered state features. Actions are selected via ϵ -greedy exploration and post-processed through domain-specific constraints. A centralized Mixing Network then combines individual Q-values into a joint Q-total, using hypernetwork-conditioned weights based on the concatenated global state. The network enforces monotonicity to ensure consistency between decentralized decisions and centralized optimization. Training relies on a recency-biased replay buffer, joint TD loss minimization, and soft-updated target networks for both agent and mixer components. Learning is driven by a shared global reward signal balancing revenue changes and price stability, enabling coordinated but decentralized pricing strategies.