

**RESEARCH PROPOSAL**

**EKC500 SCIENCE AND ENGINEERING RESEARCH METHODOLOGY (2022/2023)**

|  |  |
| --- | --- |
| Name  Matric Number  Supervisor  School/Centre  Study Level (PhD/MSc)  Research Title  Supervisor name and stamp | Lu Chin Han  22306950  Professor Ir. Dr. Nor Ashidi Mat Isa  School of Electrical and Electronic Engineering  MSc  Real-Time Pricing Optimization Integration for Industry 4.0 Point of Sale (POS) Systems. |

**TABLE OF CONTENT**

[**Chapter 1: Introduction** 3](#_Toc156621129)

[**1.1 Background** 3](#_Toc156621130)

[**1.2 Current Trends on Machine Learning Based Pricing** 3](#_Toc156621131)

[**1.3 Problem Statement** 4](#_Toc156621132)

[**1.4 Research Objectives** 5](#_Toc156621133)

[**1.5 Research Scopes** 5](#_Toc156621134)

[**Chapter 2: Literature Review** 6](#_Toc156621135)

[**2.1 Introduction** 6](#_Toc156621136)

[**2.2 Dynamic Pricing** 6](#_Toc156621137)

[**2.2.1 Inventory-based Pricing** 6](#_Toc156621138)

[**2.2.2 Time-based Pricing** 7](#_Toc156621139)

[**2.2.3 Segmented Pricing** 8](#_Toc156621140)

[**2.2.4 Summary** 8](#_Toc156621141)

[**2.3 Machine Learning** 9](#_Toc156621142)

[**2.3.1 Concept of Machine Learning** 9](#_Toc156621143)

[**2.3.2 Type of Machine Learning Algorithm** 9](#_Toc156621144)

[**2.3.3 Machine Learning Based Real-Time Price Optimization Method** 11](#_Toc156621145)

[**2.3.4 Summary** 13](#_Toc156621146)

[**Chapter 3: Methodology** 14](#_Toc156621147)

[**3.1 Problem Identification** 14](#_Toc156621148)

[**3.2 Data Collection** 15](#_Toc156621149)

[**3.3 Data Cleaning and Preprocessing** 16](#_Toc156621150)

[**3.4 Feature Extraction** 16](#_Toc156621151)

[**3.5 Model Training** 16](#_Toc156621152)

[**3.5.1 TensorFlow** 16](#_Toc156621153)

[**3.6 Model Performance Evaluation and Fine-Tuning** 17](#_Toc156621154)

[**3.7 Integration of Model** 17](#_Toc156621155)

[**3.7.1 Node.js** 17](#_Toc156621156)

[**3.8 Monitoring and Updating** 17](#_Toc156621157)

[**3.9 Feedback Loop for Continuous Learning** 17](#_Toc156621158)

[**Chapter 4: Gantt Chart** 18](#_Toc156621159)

[**Chapter 5: Expected Outcomes** 19](#_Toc156621160)

[**Chapter 6: References** 20](#_Toc156621161)

# **Chapter 1: Introduction**

## **1.1 Background**

The unfolding of Industry 4.0 unfolds an era characterized by the fusion of advanced technologies and industry, birthing a new framework of connected, agile and data-driven systems (Lu, 2017). Central to this transformation are Point of Sale (POS) systems, traditionally used for processing sales transactions but now emerging as integrated platforms capable of inventory management, customer relationship management, and more (Inman & Nikolova, 2017). Retailers, navigating the challenges and opportunities of the digital age, find themselves amidst an evolving landscape that demands innovative pricing strategies (ehret & Wirtz, 2017). One such evolution is the shift towards real-time pricing optimization - a dynamic approach to pricing, leveraging data, and technology to determine optimal prices that maximize profits and customer satisfaction (Bose et al., 2012).

Research paper by Zhu & Mukhopadhyay (2003) focusing on the development and integration of a real-time pricing optimization model into Industry 4.0-enabled POS systems, utilizing Machine Learning (ML) technologies Incorporating elements like customer demand, market trends, competitive landscape, and more, the hypothesis is that the use of Machine Learning for pricing decisions can enhance operational efficiencies, heighten profit margins, and ultimately drive a superior customer experience (Ferreira et al., 2018). The proposed model in this research uses ML techniques (such as regression, neural networks, or ensemble methods) and real-time analytics to determine optimal pricing (Chen & Rothschild, 2010). These techniques will be applied using Python and its array of ML libraries to develop and train models, which are then integrated with Kafka or similar technologies for real-time data processing. Bose et al (2012) aims to advance the understanding of real-time pricing implementations in retail, highlighting the practicality, benefits, and potential challenges of using an ML-driven pricing strategy. This research strives to illuminate how adopting machine learning technologies can permit retailers to adjust their strategies swiftly, respond efficiently to market changes, and carve a robust pathway towards achieving competitive advantage in a highly digitalized and customer-centric retail environment (Lu, 2017).

By emphasizing the effective use of machine learning and data-driven approaches, Schwab (2017) illustrate how Industry 4.0 is not just a visionary concept but a feasible strategy that retailers can leverage today to create value. This integration of real-time pricing optimization models is poised to revolutionize the retail landscape by offering unprecedented levels of operational efficiency and customer satisfaction, positioning itself as a critical component in the shaping of the Industry 4.0 retail experience (Schwab, 2017).

## **1.2 Current Trends on Machine Learning Based Pricing**

The application of Machine Learning (ML) to pricing strategies has seen significant evolution over the past several decades, demonstrating the broad influence and reach of this technology. Years ago, dynamic pricing powered by ML was first seen in the airline industry. They leveraged ML to adjust ticket prices based on numerous variables such as demand fluctuations, seasonality, time of purchase, and flight schedules. This method effectively transformed industry pricing strategies by identifying peak travel times and adjusting prices to optimally match demand, thereby significantly increasing profitability. As this strategy proved groundbreaking, it began to find its way into the hospitality industry. Relying on similar principles, room rates were dynamically adjusted during high-demand periods or significant events resulting in a substantial improvement in their financial performance.

With the expansion of e-commerce, online retail giants like Amazon began integrating complex ML algorithms into their pricing strategies. These algorithms facilitated dynamic price adjustments according to factors such as buyer behavior, competitors' prices, and product preferences. Amazon, as an early adopter of this approach, took advantage of dynamic pricing by altering the prices of their products up to 2.5 million times daily. This enabled them to stay competitively priced while drastically enhancing their sales performance. In more recent years, ride-sharing platforms, such as Uber and Lyft, have harnessed ML to implement 'surge pricing', creating a pricing model responsive to demand. By increasing prices during peak hours, unfavorable weather conditions, or periods of heightened demand, they were able to effectively manage demand and supply equilibrium, leading to augmented revenue and incentivizing drivers to service high-demand areas.

In an even more individualized application of ML in pricing, some enterprises have begun adjusting prices per individual consumer based on their specific data, a practice known as 'price discrimination.' This method uses an analysis of a consumer’s browsing history, purchasing behavior, and socio-economic factors to detect their willingness and ability to pay. Indeed controversial, this approach greatly expands the potential of what is achievable through ML pricing. Reflecting on the evolution of ML's role in pricing strategies over the years, it's clear the technology's transformative and influential potential. As ML continues to advance, companies can expect to discover increasingly innovative applications for dynamic pricing, potentially driving profitability and business growth to new levels, while being mindful of critical ethical considerations and regulatory frameworks.

## **1.3 Problem Statement**

In today's fast-paced retail environment, the need for dynamic and responsive pricing models is more critical than ever. Traditional pricing strategies are often static and simplistic, failing to account for the rapidly changing factors such as customer demand, inventory levels, competitor pricing, and market trends. As a result, they miss out on the potential benefits that a more dynamic pricing strategy could provide.

The project, "Real-Time Pricing Optimization Integration for Industry 4.0 Point of Sale (POS) Systems," recognizes these challenges and aims to address them using the power of Machine Learning. Currently, many systems that employ Machine Learning do not adapt their prices in real-time. This can lead to outdated pricing decisions, which can negatively affect their profit margins and consumer satisfaction rates (Chiang, Chen, & Xu, 2007). Another significant problem lies is their heavy reliance on historical data. These systems are often unable to keep pace with the evolving trends and real-time information, meaning they miss opportunities for more effective pricing strategies as discovered by Lee, Kao, & Yang (2014).

Moreover, existing systems tend to overlook the importance of using the vast amount of data they hold effectively. Rather than considering critical variables like competitor pricing and economic trends, they mostly focus on demand-supply dynamics, leading to less optimal pricing decisions (Chen, Chiang, & Storey, 2012). Adding to these complications is the technical difficulty of integrating machine learning into these systems and the frequent need for manual intervention in pricing decisions (Rossmann et al., 2018, Zhang, Fjell & Dröge, 2011). The proposed project aims to resolve these barriers. It proposes a system that is responsive in real-time and optimizes the use of data for pricing decisions. In essence, it seeks to make pricing faster, smarter, and more accurate, using the most recent industry data. When implemented, this approach could help businesses to be more competitive, enhance their profit margins, and be ready for the future of retail.

## **1.4 Research Objectives**

* To design and develop a dynamic, real-time pricing optimization model that utilizes machine learning algorithms. This model should integrate seamlessly within Industry 4.0's POS systems in the retail industry.
* To examine the functionalities and capabilities of modern Industry 4.0 POS systems.

## **1.5 Research Scopes**

In this research, it will be focusing on developing a real-time based price optimization model. The model will be trained using dataset obtained from open-source platform Kagle namely “Brazilian E-Commerce Public Dataset”. Not only that, for price optimization, there are a variety of pricing, and it is mainly focusing on time-series pricing and segmented pricing. Aside from that, the model will be trained using two different type of Reinforcement Learning (RL) algorithm which is Deep Q-Learning (DQN) and Deep Deterministic Policy Gradient (DDPG).

# **Chapter 2: Literature Review**

## **2.1 Introduction**

New technologies are always transforming how retail businesses work, especially in terms of deciding prices. This literature review focuses on one important part of this ongoing change: the use of real-time pricing in Point of Sale (POS) systems within the modern, technology-driven business environment known as Industry 4.0.

## **2.2 Dynamic Pricing**

The literature review commences with a historical analysis of pricing methods in retail, from cost-plus pricing to today's dynamic pricing strategies (Smith, 1983). The limitations of static pricing models, most notably their incapacity to adapt to market changes, are widely recognized (Russo et al., 1981) thus leading to the advent of dynamic pricing models (Bose et al., 2012). Dynamic pricing, also known as surge pricing, demand pricing, or time-based pricing is a pricing strategy in which businesses set flexible prices for products or services based on current market demands.

Businesses are able to change prices based on algorithms that take into consideration competitor pricing, supply and demand, and other external factors in the market. (Azaria, 2023) Dynamic pricing is a common practice in several industries such as hospitality, tourism, entertainment, retail, electricity, and public transport. This pricing strategy is used as a response to change market scenarios, with the objective of maximizing profits by capturing the highest amount consumers are willing to pay at any particular time. By rapidly adapting to market dynamics, businesses can turn favorable market situations to their advantage and mitigate the adverse effects of unfavorable market changes.

Dynamic pricing became more feasible with the rise of internet retailing and sophisticated analytics software, which allows businesses to quickly adjust prices based on real-time supply and demand. The use of machine learning and artificial intelligence has further refined dynamic pricing techniques, increasing the precision of pricing predictions. Yet, a range of mathematical models has been employed in the calculation of dynamic prices. These models usually format the problem of dynamic pricing as an optimization problem. Depending on the particular mathematical technique emphasized, these models can be classified into five different categories such as inventory-based pricing, time-based pricing, dynamic pricing based on competitors and segmented pricing. (Azaria, 2023)

### **2.2.1 Inventory-based Pricing**

Inventory-based pricing, also identified as stock-based pricing, is a type of dynamic pricing strategy where price adjustments are made based on existing stock levels or inventory forecasts. Numerous factors contribute to an item’s price determination, such as seasonality, day of the week, competitor prices, among others. However, the ongoing availability of a particular product remains a critical factor in pricing calculations. (7Learnings, n.d.)

Various factors that influence price can change quickly, particularly as inventory levels fluctuate. For instance, a retailer might start the week with a fully stocked item. Given the ample supply to cater to market demand, the retailer might benefit from establishing lower prices. However, as the stock depletes throughout the week due to sales, the retailer might need to increase the item’s price to prevent a stock-out situation. Once the inventory is restocked, prices could be reduced accordingly. (Chen, 2018) Thus, the pricing of items is partially dependent on the ongoing availability of the stock.

Implementing an inventory-based pricing model can be achieved in two ways: through a rule-based pricing system or an algorithm-based system powered by machine learning. In a rule-based pricing system, retailers manually set the pricing rules and consistently monitor stock levels and other changing conditions. However, this process can be time-consuming. On the other hand, utilizing a machine-learning algorithm automates the entire process. With this system, users or retailers can depend on intelligent algorithms to maximize their revenue efficiently and effectively, eliminating the need for constant manual monitoring and adjustments. (7Learnings, n.d.)

### **2.2.2 Time-based Pricing**

Time-based pricing is a strategy where prices fluctuate based on the time of purchase or use, often utilized in the transportation, hospitality, and entertainment industries to maximize profits. It aligns prices with customer demand, allowing businesses like airlines to offer lower fares during off-peak hours.

Additionally, it facilitates effective capacity management, with businesses like theme parks charging higher rates during peak periods to control demand and prevent overcrowding. It is extensively used in the hotel industry to adjust room rates for optimization of occupancy and revenue, and in the electricity sector, particularly for utilities, where companies apply time-of-use pricing to encourage consumers to shift their energy consumption to off-peak periods and balance power grid load. (Faster Capital, n.d.)

However, in developing this time-based pricing model there are several factors need to be taken into consideration such as understanding target market. This is due to the reason that. Different customer groups may show varying buying habits and price sensitivities at different times. For instance, consider a restaurant located in a business district. Lunchtime could be a busy period when working professionals are ready to pay more for a swift and easy meal. However, dinner time might be less busy, and the restaurant could offer reduced prices to entice customers. By studying the behaviors and preferences of the target market, the restaurant can tactically adjust prices at different times to boost revenue. (Faster Capital, n.d.)

Not only that, analyzing demand pattern is another factor that needs to be considered. This can be done by studying the historical data and knowing the peak and off-peak periods. So with this seller can charge at a higher rate during the peak periods and lower rate on the off peak period such as mid-morning or mid -afternoon.

### **2.2.3 Segmented Pricing**

Segmented pricing or so-called price segmentation is a strategy where it is charged based on customers willingness to pay and provide some discounts to those who are more price sensitive. By adopting price segmentation, it allows businesses to be able to capture customer segments that might be neglected by its competitors.

In segmented pricing there are several types of segmented pricing as well such as volume-based pricing segmentation, geographic pricing segmentation, value-based pricing etc. Volume-based pricing is pricing based on the volume of a product or service being sold. Geographic pricing is the price of the identical product will be charged based on the location of the customers. Value-based pricing is the product or service is priced based on the perceived value of the product to customer (Ali, 2023). For instance, luxury brands will be charged a higher price for their products/services.

Adopting segmented pricing brings multiple advantages for business such as maximizing revenue, improve customer satisfaction, provides a competitive advantage, and enables efficient resource allocation. However, there are some potential disadvantages as well such as customer confusion on price, brand image affected and complexity in developing segmented pricing system (Ali, 2023).

### **2.2.4 Summary**

This section provides an overview of dynamic pricing, a strategy where prices are flexible and adjust in response to current market demands. This pricing strategy has become feasible with internet retailing and advanced analytics and is widely used in hospitality, entertainment, retail, electricity and public transport sectors among others. The pricing models are typically classified into five categories including inventory-based, time-based, and segmented pricing.

Inventory-based pricing is where prices are set based on existing stock levels or inventory forecasts. Various factors can affect price, but the available stock of a product is critical in the pricing calculation. Time-based pricing allows prices to fluctuate depending on the time of purchase or use. Several factors can guide the implementation of this method, such as understanding target markets, and analyzing demand patterns.

Lastly, segmented pricing involves charging prices based on customers' willingness to pay, allowing businesses to capture more customer segments. This involves types like volume-based pricing segmentation, geographic pricing segmentation, value-based pricing among others. This method has various pros and cons.

## **2.3 Machine Learning**

### **2.3.1 Concept of Machine Learning**

Machine Learning, an integral branch of artificial intelligence, equips computers with the ability to learn from data independently. This unique ability makes it possible for these machines to enhance their performance over time without being manually programmed to do so. (Samuel, 1959) The concept of machine learning can be compared to the process of teaching a child - the same way a child learns through examples, machine learning algorithms analyze considerable quantities of data to deliver accurate predictions and informed decisions.

In machine learning, learning techniques are categorized into three main types: supervised, unsupervised, and reinforcement learning. In supervised learning, machines are provided with labeled data, aiding the algorithm in understanding what to look for, whereas, in unsupervised learning, machines are tasked with identifying patterns and relationships in unlabeled data independently. Reinforcement learning, on the other hand, allows a machine to learn from its past actions and outcomes, akin to a person becoming proficient at a video game through repeated practice. (Samuel, 1959)

Machine learning has been adopted on different applications today. Its presence is felt in various industries and technologies, ranging from Netflix's movie recommendations to voice assistant programs like Siri or Alexa, and even to predicting diseases in the medical field. Undeniably, machine learning has become a hot topic in technology and its significance continues to grow with each passing day. Thus, in essence, machine learning is about a computer's ability to learn, grow, and make decisions, mirroring human intelligence.

### **2.3.2 Type of Machine Learning Algorithm**

#### **2.3.2.1 Deep Q-Learning**

The Deep Q-Network (DQN), as postulated by Mnih et al. (2015), forms the basis for efficient and optimal decision-making in real-time environments. A key feature of DQN is its potential to break down the complexity of large state-action spaces through the use of a deep neural network as a function approximator. This approach allows it to generate barrier-breaking predictions of Q-values for each action given a state, without the need for comprehensive storage and manual updates of each state-action pair, a limitation often faced with traditional Q-Learning. This significantly reduces computational requirements and makes DQN apt for real-world applications with large state-action spaces, such as real-time price optimization.

DQN incorporates a function known as experience replay, originally proposed by Lin (1992). This technique allows it to store transitions between states that result from particular actions and later sample these transitions randomly to update its policy. This effective use of historical data enables it to break away from the limitations of sequential data and opens the door to a wider exploration of policy space. The ability to recycle past data for new policy updates provides the agent with a higher level of data efficiency compared to models that don't employ experience replay.

Stability in the learning process is also a vital aspect addressed by the DQN. Instead of updating Q-values simultaneously, DQN uses a separate 'target' network to generate Q-learning targets. This implementation was discussed by Mnih et al. (2015) and was found to contribute significantly to countering the instability that arises from simultaneous updates. Notwithstanding its benefits, DQN poses certain challenges. As highlighted by Henderson et al. (2018), it requires an extensive dataset for effective training. Furthermore, the performance of the model is heavily hinged on the appropriate tuning of hyperparameters, demanding careful calibration for the model to perform optimally.

To summarize, the scalability, data efficiency, and stability brought in by the DQN model’s unique features make it a promising choice for handling large state-action spaces and real-time decision-making tasks. The application of DQN to real-time pricing optimization presents exciting possibilities.

#### **2.3.2.2 Deep Deterministic Policy Gradient (DDPG)**

Deep Deterministic Policy Gradient (DDPG) is an actor-critic and model-free algorithm that has been widely applied in areas that require continuous and high-dimensional action spaces. It combines the strengths of Deep Q-Learning and Policy Gradients methods (Lillicrap et al., 2015). DDPG has two main parts: the actor and the critic. The actor's role is to provide the current policy, i.e., map the given state to the specific action, while the critic provides the Q-value by evaluating the actor's decision based on the selected action and the given state (Yoon, 2019). These two components work together to continuously learn the optimal policy.

The agent, via the actor, first interacts with the environment to conduct actions and collect experience. Experience here includes the state, action, reward, and the next state. In parallel, as per Lillicrap et al. (2015), the critic learns Q-values by using Bellman's equation (the same equation used in Q-learning). This uses experiences stored in the replay buffer, thus opening the RL agent to off-policy learning and allowing it to break correlations in the observation sequence for more robust learning. Once the critic is trained, the Q-values are fixed, and the actor's parameters are then adjusted using policy gradients to generate better actions. The policy here is deterministic, meaning one particular action is estimated for each state, which is a significant aspect considering continuous action spaces.

One important feature to discuss in DDPG algorithm is soft updates for the target networks used in training both actor and critic (Lillicrap et al., 2015). In soft updates, instead of copying the weights from original networks to target networks after fixed intervals as done in DQN, weights are updated slowly at each timestep, allowing the Q-values to chase a moving target and ensuring more consistent learning. While DDPG offers notable advantages, there are challenges as well. Training DDPG may be relatively slower than DQN due to the additional complexities involved, and it may require meticulous tuning of hyperparameters like the learning rate, discount factor, and noise parameters to attain optimal performance (Henderson et al., 2018).

To summarize, DDPG stands out for its ability in continuous control tasks and handling high-dimensional action spaces, presenting a strong fit for use-cases like price optimization where actions (prices) can range over a continuous space. But its adoption comes with the need for careful tuning and potentially slower training times than other reinforcement learning algorithms.

### **2.3.3 Machine Learning Based Real-Time Price Optimization Method**

There are different studies had been conducted to study on the performance of different algorithm to develop a price optimization model. Table 2.4.1 shows a summary of performance of deploying different algorithm.

|  |  |  |
| --- | --- | --- |
| **Type of Algorithm** | **Type of Datasets** | **Result** |
| VAR Model (Supervised Learning) | Product Details | RMSE = 42.1 |
| DDPG Model (Reinforcement Learning) | EV-Charging Price | Quarter-hourly pricing best performance |

Based on table shown above, it can be seen that there are 2 different algorithms being deployed and student in developing a real-time price optimization model. The first algorithm being deployed is Vector AutoRegressive (VAR) model which falls under supervised learning. The model was fed with product details which contain Product ID, Price and Date as shown in Figure 2.4.1.

A diagram of a data flow

Description automatically generated

*Figure 2.4.1: Data stored in the database for training VAR model.* *Lundkvist (2014)*

In order to understand the model performance, Lundkvist (2014) had compared the VAR model with the simple model. The simple model basically predicts the price by simply takes the last observed value and used it as prediction for the feature value. The performance of the models, in both examples, were evaluated using the method of Residual Analysis and the Root Mean Square Error (RMSE). Residuals are the differences between the observed and predicted values. Residual analysis helps understand the variation in the errors across different scenarios. The residuals for both the VAR model and the simple model are compared in Figure 2.4.2. Residuals closer to zero suggest better model performance. For the VAR model, residuals were found to be more fluctuating as they also consider the price of the product, and thus it had lesser accuracy compared to the simple model when the prices didn't change. However, when the prices changed, the VAR model showed smaller residuals suggesting better prediction of changes than the simple model as shown in Figure 2.4.3. Another issue observed with the VAR model residuals was drift from zero, which was resolved when a "moving model" approach was considered, meaning a model that continuously updates as it gets new information. This approach yielded better results and eliminated the problem of the drift in residuals. In addition to Residual Analysis, the RMSE was used to measure the predictions' accuracy. The RMSE is a standard measure that squares the deviations before averaging them, punishing large errors more than the small ones. This measure showed that the VAR model performed better than the simple model with an RMSE of 42.1 compared to 50.7 of the simple models.

A graph of a graph

Description automatically generated with medium confidence

*Table 2.4.2: Residuals for the mean price with the VAR and simple model.*

A graph of a simple model

Description automatically generated

*Table 2.4.3: Residuals for the mean price with the VAR and simple model using a moving model part.*

For the next study by Liu (2021), the model was trained using EV-Charging price and he had trained three different models for three different strategies and the performance of three different pricing strategies (quarter-hourly pricing, peak-valley time-of-use tariff, and hourly pricing) for electric vehicle charging was analyzed based on an algorithm known as DDPG (Deep Deterministic Policy Gradient). The strategies were compared using actual charging data of 166 days in the Northern Hebei Province. The model performance was evaluated by observing the algorithm convergence and total revenue changes under daily pricing updates. The pricing strategy that provided the highest additional revenue was the quarter-hourly pricing strategy, which allowed for more flexibility in tracking the power market's pricing signals. The performance analysis also included examining load changes under dynamic pricing, tracking average returns at different time points, and identifying the sources of incremental revenue. Ultimately, the quarter-hourly pricing strategy outperformed others, increasing the overall operational income by 10%. Although the DDPG model provided more accurate pricing strategies and increased overall operational income, the model requires a large amount of data for training and can be highly sensitive to initial parameters, which is a challenge for its scalability and stability.

### **2.3.4 Summary**

This section provides a deeper understanding of two machine learning algorithms, Deep Q-Networks (DQN) and Deep Deterministic Policy Gradient (DDPG), both of which hold promise for real-time pricing optimization. DQN's scalability, efficiency, and stability make it suitable for handling large state-action spaces and real-time decision-making tasks. On the other hand, DDPG's aptitude for continuous control tasks and high-dimensional action spaces, despite potential longer training times and the need for careful hyperparameter tuning, offers flexibility for use-cases like price optimization.

A comparison of two pricing models, Vector AutoRegressive (VAR) and DDPG, revealed that both models displayed favourable results in their respective applications. The VAR model outshined a simplistic model in a Root Mean Square Error (RMSE) performance measure, while the DDPG model maximized revenue in an EV-Charging price scenario. Ultimately, the choice of machine learning model largely depends on the data's characteristics, computational constraints, and specific objectives.

# **Chapter 3: Methodology**

The methodology section of this study provides an in-depth look at the techniques and processes used to create a real-time dynamic pricing model. It presents a systematic breakdown of the research process, from data gathering and selection of the machine learning algorithm, to the training, implementation, and continuous evaluation of the model as shown in Figure 3.1.1. By detailing these steps, the study aims to offer a clear understanding of the process and gives an opportunity for the results to be validated through repeating the approach. The primary objective is to showcase how machine learning, a subset of artificial intelligence, can effectively optimize pricing strategies in the era of Industry 4.0. The explanation of the applied methodology contributes insights into the creation of this innovative pricing model, while also highlighting the transformative potential of such technologies in revamping retail pricing strategies.

## **3.1 Problem Identification**

This stage of the process clarifies the goals and objectives of the project. The project’s primary goal is to utilize technology to transition from traditional static pricing strategies to dynamic, real-time pricing strategies. Defining the challenges faced in the retail context, such as the complexity of managing multiple products and variations in demand, sets the stage for highlighting the necessity and benefits of a dynamic pricing model. The expected outcomes should consider business objectives like optimizing revenues or improving sales volume.

A diagram of a model performance

Description automatically generated

*Figure 3.1.1: Flowchart of developing the model using DQN*

## **3.2 Front End Development**

The front-end development of the Point of Sale (POS) system was approached with meticulous attention to detail, aiming to deliver a seamless and intuitive user experience that leverages the latest advancements in web development technologies. The interface, developed using HTML5, CSS3, and JavaScript, was designed with a strong emphasis on responsive design, accessibility, and real-time interaction, ensuring that the POS system not only meets but exceeds the expectations of modern retail operations. Below, we delve deeper into the key features that define the front-end of the POS system:

### **3.2.1 Product Display and Dynamic Pricing**

The interactive product catalog, a pivotal element of the Point of Sale (POS) system, was meticulously developed utilizing the latest web technologies, HTML5 and CSS3. The adoption of HTML5 enhances the catalog's structure and accessibility, offering native support for multimedia elements and optimizing the experience for mobile users through features like touch events and geolocation. CSS3 elevates the visual appeal with advanced styling options such as gradients, shadows, and animations, enabling a dynamic and engaging interface without compromising performance. This strategic choice facilitated the creation of a visually appealing grid layout for product display, where each product card is enriched with vital details such as the product name, an illustrative image, and a price that dynamically adjusts in real-time.

The dynamic pricing mechanism, powered by JavaScript and AJAX, ensures real-time updates of pricing information from the backend, reflecting changes in demand, inventory, and regional pricing trends without the need for manual page refreshes. This integration not only enhances the shopping experience by providing current pricing information but also optimizes server workload by reducing page reload requests, showcasing a commitment to a modern, efficient, and user-centric shopping experience.

### **3.2.2 Shopping Cart Functionality**

The shopping cart functionality represents a critical component of the system's architecture, designed to offer users a seamless and intuitive shopping experience. Leveraging advanced JavaScript techniques, the shopping cart feature enables users to effortlessly add or remove products from their cart. The cart's contents, along with the total price, are dynamically updated in real-time to accurately reflect the user's current selections. To further refine the user experience and ensure the continuity of shopping activities, the state of the shopping cart is preserved in the browser's local storage. This ingenious solution ensures that users can pick up exactly where they left off, even if they close their browser or navigate away from the page, thereby eliminating the frustration of losing their carefully selected items.

### **3.2.3 Checkout and Payment Processing**

The checkout and payment processing aspect of the system was designed with both simplicity and security in mind. A dedicated checkout page was developed to collect user information and display an order summary, including the dynamically calculated total price, thereby streamlining the transition to payment. The system offers a variety of payment methods, such as credit cards, digital wallets, and bank transfers, each represented with an icon and a label for easy selection through JavaScript event listeners. Before processing payments, the system validates the selected payment method and user input to ensure accuracy. Upon successful payment, users are redirected to a success page, affirming the transaction and providing detailed order information, which reinforces transparency and customer satisfaction.

**3.2.4 Enhanced Shopping Cart Functionality with Global Dynamic Pricing**

The latest iteration of the Point of Sale (POS) system introduces an advanced shopping cart feature that significantly simplifies global deployment for merchants. This enhancement leverages state-of-the-art web technologies—HTML5, CSS3, JavaScript, and AJAX—alongside a sophisticated artificial intelligence (AI) model. This powerful combination enables the system to automatically adjust product prices in real-time based on the geographical location of the store or the customer. Whether a customer is shopping in Paris, France, or Tokyo, Japan, the system intelligently displays prices that are relevant and competitive for their specific market.

At the heart of this feature is its AI-driven dynamic pricing mechanism. This mechanism conducts a thorough analysis of various factors, such as local demand, inventory levels, and prevailing regional pricing trends, to dynamically tailor prices. This ensures that customers are always presented with the most accurate and contextually appropriate pricing information. Such a feature not only enhances the shopping experience by providing regionally customized pricing but also significantly reduces the server's workload by minimizing the need for manual price adjustments and page reloads.

One of the standout aspects of this feature is its plug-and-play capability for merchants across the globe. With minimal setup required, merchants can easily select their operating region, and the system automatically configures itself to reflect the appropriate pricing strategy for that region. This global adaptability feature eliminates the need for merchants to invest in multiple systems or engage in complex customization processes. It democratizes access to a sophisticated POS system, enabling merchants of all sizes and geographical locations to leverage advanced technology to enhance their competitiveness in the global market.

Furthermore, this feature underscores the commitment to developing user-centric solutions that address the diverse needs of a global user base. It showcases the system's flexibility and readiness to support merchants and customers from various economic backgrounds and geographical locations. By seamlessly integrating cutting-edge technology with intelligent, region-specific pricing strategies, the enhanced shopping cart functionality exemplifies an innovative approach to POS system development. This approach not only prioritizes responsiveness and accessibility but also ensures the system's adaptability to the multifaceted economic conditions encountered by the global community of users.

In conclusion, the introduction of global dynamic pricing in the shopping cart functionality represents a significant advancement towards creating a more inclusive, efficient, and personalized shopping and selling experience. It reflects a dedication to innovation and a vision of empowering merchants worldwide with technology that is not only advanced but also inherently adaptable to the unique challenges and opportunities presented by the global marketplace.

**3.2.4 Responsive Design** **and Accessibility**

In the development of the POS system, responsive design and accessibility were treated as paramount considerations. Adopting a mobile-first strategy, the system employs CSS media queries to ensure an optimal viewing experience across a wide range of devices, from smartphones to desktop computers. This approach guarantees that the system is accessible and user-friendly for all customers, regardless of the device they use. Additionally, the integration of ARIA (Accessible Rich Internet Applications) roles and properties, along with a focus on keyboard navigation and focus management, makes the system fully accessible to users with disabilities. This commitment to accessibility ensures that the POS system is inclusive, catering to the needs of all users and providing them with an equitable shopping experience.

### **3.2.5 User Interface Enhancements**

User interface enhancements were implemented to elevate the shopping experience further. Interactive UI elements, such as hover effects on buttons and product cards, were introduced using CSS transitions and JavaScript, providing visual feedback and making the interface more engaging. Custom error handling and validations were also established to guide users through the form submission process effectively, ensuring that all required information is provided correctly before proceeding.

## **3.3 Backend Development**

The backend development of the Point of Sale (POS) system is a cornerstone for ensuring seamless operation and integration of the system's dynamic features. This section delves into the specifics of the backend architecture, focusing on the database design, API development, dynamic pricing logic, transaction processing, and the integration with the frontend.

### **3.3.1 Database Design and Schema with MySQL**

At the heart of the backend lies the MySQL database, chosen for its robustness, scalability, and widespread adoption in the industry. The database is meticulously structured to house essential data across several tables, each serving a distinct purpose within the POS system. The Products Table is central to the system, storing comprehensive product information including unique identifiers, names, images, and region-specific prices in a JSON column. This design choice facilitates the dynamic adjustment of prices based on regional demand, directly from the database. The Transactions Table captures every sale, logging details such as timestamps, user and product IDs, quantities, and total prices. This table is instrumental for tracking sales, analyzing trends, and managing inventory effectively. Additionally, the Users Table maintains user data, including authentication details, which are crucial for personalizing the shopping experience and ensuring system security. The relational nature of MySQL, combined with its performance capabilities, makes it an ideal choice for managing the complex data relationships inherent in the POS system.

### **3.3.2 API Development with Flask**

For the development of RESTful APIs, Flask stands out as the framework of choice due to its simplicity, flexibility, and scalability. Flask's minimalistic yet powerful approach allows for the rapid development and deployment of APIs, making it particularly suitable for projects with stringent timelines. Its flexibility is a significant advantage, offering the freedom to choose the best tools for each task and allowing for highly customized solutions. Flask's scalability ensures that the backend can accommodate the POS system's growth, while its vibrant community and rich ecosystem of extensions enhance its functionality and ease of integration. The APIs developed with Flask cover a wide range of operations, from product management in the cart to transaction processing, all secured with robust authentication mechanisms to protect sensitive data.

### **3.3.3 Dynamic Pricing Logic**

A key feature of the backend is its sophisticated dynamic pricing logic, which adjusts product prices in real-time based on regional demand. This logic, implemented through algorithms that analyze sales data, inventory levels, and regional factors, allows for the application of data-driven pricing models directly within the database. The use of Flask and MySQL for this purpose enables the efficient execution of complex queries and the seamless update of pricing models to reflect current market conditions. This ensures that the POS system remains competitive and responsive to market dynamics, offering users accurate and region-specific pricing information.

### **3.3.4 Transaction Processing and Data Persistence**

Transaction processing is a multi-step operation that includes cart validation, total price calculation, and the application of dynamic pricing adjustments. Upon the successful completion of a transaction, the details are securely stored in the MySQL database, ensuring accurate record-keeping and data integrity. The combination of Flask's capability to handle various data payloads and MySQL's transactional support guarantees that transactions are processed securely and efficiently, with all transaction data being accurately reflected in the system.

### **3.3.5 Integration with Frontend**

The integration between the backend and frontend is facilitated through the RESTful APIs developed with Flask, ensuring a smooth and responsive user experience. This integration is rigorously tested to handle concurrent requests and maintain data consistency across the system. The lightweight nature of Flask, combined with AJAX calls from the frontend, ensures real-time updates and responsiveness, providing users with an interactive and dynamic shopping experience.

In conclusion, the backend development of the POS system, leveraging MySQL for database management and Flask for API development, creates a robust, scalable, and secure platform. This backend architecture supports the system's dynamic features, such as region-specific pricing and transaction processing, ensuring a seamless and efficient user experience.

## **3.4 Model Training**

The methodology for model training in this study is a comprehensive process designed to identify high-demand regions for various product categories, leveraging a meticulously cleaned and prepared dataset. This process not only involves traditional data preparation and clustering techniques but also incorporates elements of feature extraction to enhance the analysis. The following subsections detail the approach from data preparation to model training and the application of findings.

### **3.4.1 Data Preparation and Aggregation**

Technological tools facilitating data extraction and collection are crucial in this stage. Data needed for the machine learning model can include sales history, promotional activities, customer behavior patterns, and market trends. All of this info was obtained from open-source dataset platform – Kagle and the name of dataset used is “Brazilian E-Commerce Public Dataset by Olist” (Olist, 2022). The process begins with loading the dataset into a pandas Data Frame, simplifying data manipulation and analysis. A helper column, sales\_count, is introduced to enumerate each sale, facilitating the aggregation of sales data by geolocation state and product\_category\_name\_english. This aggregation yields a count of sales for each product category within each state, serving as a crucial step in identifying regional sales trends and preferences. This step, while not labeled explicitly as feature extraction, effectively distills raw data into a more informative representation, highlighting the sales volume for different product categories across regions.

### **3.4.2 Data Transformation and Standardization**

The aggregated data undergoes a pivotal transformation into a matrix format, where rows represent regions and columns represent product categories, through a pivoting operation. This restructured data is then standardized using the StandardScaler from scikit-learn, ensuring each feature contributes equally to the analysis by normalizing the data to have a mean of 0 and a standard deviation of 1. This standardization is essential for the clustering process, as it ensures comparability across regions with varying scales of sales counts.

### **3.4.3 K-Means Clustering**

With the data standardized, the study employs K-means clustering to segment the regions based on their sales patterns across different product categories. Configured to identify three distinct clusters and initialized with 'k-means++' for optimal centroid placement, the algorithm categorizes each region into clusters that reflect their sales profile. This clustering not only categorizes regions based on sales patterns but also serves as a form of feature extraction by identifying inherent groupings within the data.

### **3.4.4 Model and Scaler Persistence**

To ensure the future applicability and consistency of the analysis, the trained K-means model and the scaler are carefully preserved using the joblib library. This preservation is essential for allowing the model and scaler to be efficiently applied to new datasets in the future without the need for retraining, thus maintaining the integrity and consistency of the methodology. The selection of joblib as the tool for this task is strategic, reflecting its superior efficiency in handling large numpy arrays, its compression capabilities for saving disk space, its simplicity and convenience for users, and its widespread acceptance within the machine learning community. These attributes make joblib the preferred choice for saving and loading the K-means model and scaler, ensuring that they can be readily utilized in subsequent analyses with the same level of precision and reliability as initially established.

### **3.4.5 Identifying High-Demand Regions**

Following clustering, the methodology seeks to identify high-demand regions by calculating the total sales for each region and determining the cluster with the highest mean total sales. Regions within this cluster are deemed high-demand areas, informing subsequent pricing strategies.

### **3.4.6 Application to Pricing Strategy**

The discovery of high-demand regions underpins a dynamic pricing strategy, where initial product prices are adjusted by 10% in these areas to reflect increased demand. This strategic price adjustment is meticulously applied to the dataset, showcasing a data-driven approach to optimizing product pricing based on regional demand patterns.

Incorporating the previous discussion on feature extraction, this methodology exemplifies a holistic and data-driven approach to understanding market dynamics. Through careful data preparation, feature extraction, clustering, and strategic application of findings, the study delivers actionable insights into regional demand patterns, facilitating informed decision-making regarding product pricing strategies.

## **3.5 Model Performance Evaluation and Fine-Tuning**

Post-training, the effectiveness of the model's predictions is examined. The model is fine-tuned as per necessity to improve its performance and accuracy. This step may involve adjusting some of the model's parameters.

## **3.6 Integration of Model**

Post-training and evaluation, the model is implemented within the existing system and integrated with the POS system. The AI model can be deployed using the TensorFlow library for runtime. The server-side runtime can then be handled using Node.js for handling concurrent connections in real-time. in order to ensure the AI model and the retail system communicate effectively, APIs (Application Programming Interface) come into use. The role of these APIs is to allow the exchange of data between the AI model and the POS system, with the AI model sending optimized prices to the POS system.

### **3.6.1 Flask**

When integrating an AI model into systems like a Point of Sale (POS) system for dynamic pricing, Flask emerges as the preferred technology due to its unique blend of simplicity, flexibility, and efficiency. Its design philosophy prioritizes ease of use, allowing developers to quickly set up a lightweight application that acts as an interface between the AI model and the system, without the complexities often associated with more cumbersome frameworks. Flask's flexibility is particularly advantageous, offering developers the freedom to customize their application architecture to meet specific model and system integration requirements. Being a micro-framework, Flask is inherently lightweight, ensuring fast and responsive applications capable of handling multiple requests efficiently critical for systems requiring real-time data processing. Moreover, Flask's extensibility allows for the seamless addition of functionalities through a wide array of available extensions, catering to the evolving needs of the project. Coupled with a strong community and comprehensive documentation, Flask provides an accessible and efficient pathway for developers to integrate AI models, making it a superior choice over other frameworks that might offer unnecessary features or a steeper learning curve for simple API development tasks.

## **3.7 Monitoring and Updating**

Post-deployment, the system is continually monitored, enabling early detection and correction of any glitches or malfunctions that may affect its functionality. The model is updated and retrained on new data as needed for optimized performance.

## **3.8 Feedback Loop for Continuous Learning**

Finally, the outcomes of the system's predictions are fed back into the model, allowing it to continually refine and improve its predictions. The AI will "learn" from its past outputs, improving the efficiency and accuracy of future operations.

# **Chapter 4: Results**

## **4.1 POS System Main Page**

The developed Point of Sale (POS) system showcased a rich array of functionalities that significantly enhanced user engagement and streamlined retail operations. A pivotal feature of the system is its dynamic pricing capability, which adjusts product prices according to the user's selected region, enabling businesses to modify their pricing strategies to reflect local market demands and conditions. By referring to Figure 4.1.2, there is a blue selection box at the middle-top of this user interface, where it serves the purpose of selecting the region this system is going to deployed. Once the region is selected, then the different product price for each product will be displayed. The different between the each of the product price can refer to both Figure 4.1.1 and Figure 4.1.2. For example, in BA region/state (representing Salvador) and ES region/state (representing Vila Velha), both air con’s price is different. This indicates that the air con’s demand in BA region has higher demand compared to ES region. This system enhances the shopping experience through an intuitive interface for product display and selection, dynamically presenting items with essential details such as names, images, and region-specific prices, thus facilitating easy browsing and selection. Additionally, the system features a real-time price update mechanism, ensuring that customers always have access to the most current and competitive pricing information. The checkout process is efficiently designed to include validation checks, ensuring a smooth transition from selection to payment. The system accommodates regional preferences by allowing users to select their region from a dropdown menu, dynamically adjusting product prices and availability to offer a personalized shopping experience. The aesthetic and functional design is further elevated by CSS styling, which ensures a visually appealing and responsive layout that enhances readability and navigability across various devices and screen sizes. Integration with backend services for dynamic pricing provides the flexibility to adapt to changing market conditions and business strategies effectively. Moreover, the potential use of local storage for preserving user selections across different pages or sessions mitigates the risk of data loss. Collectively, these features highlight the POS system's robustness and flexibility, demonstrating its ability to support effective sales transactions and inventory management while catering to the diverse needs of businesses and their customers in a detailed and user-centric manner.

A screenshot of a computer

Description automatically generated

*Figure 4.1.1: POS System Main Page for BA Region*

A screenshot of a computer

Description automatically generated

*Figure 4.1.2: POS System Main Page for ES Region*

## **4.2 POS System Shopping Cart**

The shopping cart functionality within the developed Point of Sale (POS) system represents a critical component, meticulously designed to enhance the user's purchasing process as shown in Figure 4.2.1. This feature allows for the seamless addition and removal of items, enabling users to effortlessly manage their selections with real-time updates to the cart's contents and the total price. The system's intelligent design ensures that as users navigate through the product selection interface, any action taken—whether adding a new product, adjusting quantities, or removing an item—is immediately reflected in the cart. This dynamic interaction not only improves the shopping experience by providing instant feedback on the user's actions but also aids in accurate financial planning by continuously displaying the updated total cost. Furthermore, the shopping cart's integration with the system's dynamic pricing and region-specific adjustments means that any changes in pricing or promotions are automatically applied, ensuring that the customer always receives the most current pricing information. The meticulous attention to detail in the shopping cart's functionality underscores the system's commitment to providing a user-friendly, efficient, and responsive retail environment, thereby facilitating a smooth transition from product selection to the final checkout process.

A screenshot of a computer

Description automatically generated

*Figure 4.2.1: POS System Shopping Cart UI on the right-hand side*

## **4.3 POS System Checkout Page**

For the current checkout page in the POS system as shown in Figure 4.3.1, a set of JavaScript functionalities has been developed to handle the checkout process efficiently. These functionalities are designed to manage cart data, facilitate the selection of a payment method, and process the payment upon the completion of the transaction.

Initially, when the checkout page is loaded, an event listener is established to render the order summary as soon as the DOM content is fully loaded. This is accomplished through the renderOrderSummary function, which retrieves the cart data stored in localStorage. It then iterates over each item in the cart, calculating the total price by multiplying each item's price by its quantity. The function dynamically generates HTML content to display each item's name, price, and quantity, along with the total amount due for the entire order. This provides users with a clear and concise summary of their order before they proceed with the payment.

To enable users to select a payment method, clickable div elements have been implemented for each available payment option, such as credit card, Touch n Go, and bank transfer. Each div is associated with an onclick event handler that invokes the selectPaymentMethod function when clicked. This function logs the selected payment method for debugging purposes and visually updates the UI to highlight the chosen method. It achieves this by iterating over all payment method elements and toggling a 'selected' class based on whether the method matches the one selected by the user.

Finally, the processPayment function is tasked with handling the payment process. Triggered by clicking the "Pay" button, this function logs a message indicating that payment processing has begun. For demonstration purposes, it simulates the payment process by removing the cart data from localStorage and redirecting the user to a success page. This redirection serves as a placeholder for actual payment processing, which would typically involve server-side operations and integration with payment gateways in a real-world application. Together, these functionalities form the core of the checkout page, enabling users to review their order, select a payment method, and complete the transaction with ease.

A screenshot of a computer

Description automatically generated

*Figure 4.3.1: POS System Checkout Page UI*

## **4.4 POS System Successful Payment Page**

The success page function serves as a crucial component in the user journey of the POS system, marking the completion of a transaction. Once a payment is successfully processed, the user is redirected to a success page, a deliberate design choice that provides immediate, clear confirmation that their transaction has been completed. This page not only reassures users by acknowledging the success of their payment but also enhances the overall user experience by offering a sense of closure and satisfaction. On this page, users are greeted with a message of gratitude, acknowledging their purchase and informing them that a receipt has been emailed. Additionally, a button is provided to facilitate a smooth transition back to the home page or another section of the POS system, allowing users to continue their journey or start a new transaction with ease. This function is pivotal in fostering trust and confidence in the system, ensuring users feel secure and well-informed at every step of their interaction.

A screen shot of a card

Description automatically generated

*Figure 4.4.1: POS System Successful Payment Page UI*

## **4.5 POS System Dynamic Pricing Model based on Region**

The output from the training script successfully identifies São Paulo (SP) as the high-demand region, a finding quantitatively validated by a silhouette score of 0.658376516290846. This score, indicative of a good level of clustering, underscores the model's effectiveness in discerning sales patterns and supports the implementation of targeted dynamic pricing strategies for SP. The silhouette score's role is pivotal, adding confidence to the model's capability to accurately identify high-demand regions and thereby inform strategic pricing adjustments.

The decision to increase product prices in SP by 10% across all categories demonstrates the model's practical application in optimizing pricing strategies, aiming to boost revenue and profitability. The true performance of the model, however, will be assessed over time by observing the impact of these adjustments on sales volume, revenue, and customer satisfaction. In essence, the model's identification of SP as a high-demand region, supported by a solid silhouette score, validates the use of data-driven decision-making in dynamic pricing strategies. This approach emphasizes the importance of machine learning in refining pricing strategies and highlights the need for ongoing evaluation and adjustment of the model based on new sales data and market conditions to ensure its continued effectiveness.

A screen shot of a computer

Description automatically generated

*Figure 4.5.1: AI model’s performance and output*

# **Chapter 5: Gantt Chart**

A screenshot of a computer screen

Description automatically generated

# **Chapter 5: Expected Outcomes**

By the end of this project, we anticipate a robust, operational AI model capable of executing real-time pricing optimization that can be seamlessly integrated into a Point of Sales (POS) system. This advanced model will conduct dynamic pricing adjustments in real-time based on a variety of factors such as sales trends, market conditions, customer purchasing behavior, competitive pricing information, and more.

The capability of the model to adapt to real-time changes by utilizing machine learning algorithms, particularly Deep Q-Learning (DQN), will allow for more optimal and targeted pricing strategies, leading to increased revenue and profit margins. Its integration into the POS system would allow for automatic price updates on all platforms, such as in-store, online, and mobile, fostering pricing consistency, enhancing customer experiences, and boosting overall business efficiency.

As the model continues learning and adapting to evolving market dynamics, the pricing outcomes it generates are expected to be more refined and precise as time goes on. This project will revolutionize the existing pricing model by replacing static, generic prices with an efficient, dynamic pricing scheme that caters to real-time market demands and customer behavior patterns, thereby enhancing the competitiveness and profitability of the business.

Additionally, the insights generated by the AI model can provide valuable business intelligence, allowing management to make informed strategic decisions relating to inventory management, sales forecast, and marketing strategies. In conclusion, the project is expected to deliver a technologically advanced, reliable and efficient solution for real-time price optimization, facilitating improved business performance and customer satisfaction.

# **Chapter 6: References**

Smith, J. (1983). Pricing strategies in retail: A historical analysis. Journal of Retail, 23, 34-45.

Russo, E., Johnson, E., & Stephens, D. (1981). The limitations of static pricing models. Journal of Economic Perspectives, 21(4), 145-160.

Bose, S., Chen, X., & Pashley, M. (2012). Dynamic pricing strategies: a comparative study. Journal of Revenue and Pricing Management, 11(6), 616-625.

SevenLearnings (2020). Inventory-based pricing. Retrieved from <https://7learnings.com/blog/inventory-based-pricing/>

ExpressAnalytics (2019). Developing machine learning models for dynamic pricing. Retrieved from <https://www.expressanalytics.com/blog/developing-machine-learning-models-for-dynamic-pricing/>

FasterCapital. (023.). Time-Based Pricing: How to Charge Different Prices at Different Times. Retrieved from <https://fastercapital.com/content/Time-Based-Pricing--How-to-Charge-Different-Prices-at-Different-Times.html>

Tesauro, G., Das, R., Chan, H., Kephart, J. O., Lefurgy, C., Levine, D. W., & Rawson, F. (2020). Bayesian regression and Bitcoin. Springer Nature. Retrieved [date you accessed the website], from <https://www.nature.com/articles/s43586-020-00001->2

Vendavo. (2023). Price Segmentation: What it is and How to Use it. Retrieved from <https://www.vendavo.com/all/price-segmentation/>

Ashrafi, B. (n.d.). Dynamic Pricing Using Machine Learning. Medium. Retrieved from <https://medium.com/@baabak/dynamic-pricing-using-machine-learning-5e882282effe>

AIMultiple. (n.d.). Dynamic Pricing Algorithm: A Comprehensive Guide. Retrieved [date you accessed the website], from <https://research.aimultiple.com/dynamic-pricing-algorithm/>

Ferreira, K. J., & Lee, B. (2015). The use of Machine Learning Algorithms in Recommender Systems: A Systematic Review. Expert Systems with Applications, 97, 205-227.

Bartodziej, C. J. (2017). The Concept Industry 4.0. In The Autonomy of the Person and the Digital Revolution (pp. 73-84). Springer Vieweg.

Rücker, A. Wes McKinney. (2011). Pandas: a foundational Python library for data analysis and statistics. Python for High Performance and Scientific Computing, 1-9.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Hassabis, D. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., ... & Ghemawat, S. (2016). TensorFlow: A System for Large-Scale Machine Learning. In OSDI (Vol. 16, pp. 265-283)

Chiang, W. C., Chen, J. C., & Xu, X. (2007). An overview of research on revenue management: current issues and future research. International Journal of Revenue Management, 1(1), 97-128.

Lee, J., Kao, H. A., & Yang, S. (2014). Service innovation and smart analytics for Industry 4.0 and big data environment. Procedia Cirp, 16, 3-8.

Chen, H., Chiang, R. H. L., & Storey, V. C. (2012). Business Intelligence and Analytics: From Big Data to Big Impact. MIS Quarterly, 36(4), 1165-1188.

Rossmann, B., Canzaniello, A., von der Gracht, H., & Hartmann, E. (2018). The future and social impact of Big Data Analytics in Supply Chain Management: Results from a Delphi study. Technological Forecasting and Social Change, 130, 135-149.

Zhang, D., Fjell, K., & Dröge, C. (2011). Optimal performance with real-time decision aids. Journal of Management Information Systems, 28(1), 251-286.

Mortensen, J. N., Tjøstheim, M., Løkken, I. K., Fossen, A. E., & Gondal, F. (2020). Policy Gradient Reinforcement Learning for Automated Blood Glucose Control in Type 1 Diabetes. Applied Sciences, 10(18), 6350. 1

Samuel, A. L. (1959). Some Studies in Machine Learning Using the Game of Checkers. IBM Journal of Research and Development, 3(3), 210-229.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

Lin, L. J. (1992). Self-improving reactive agents based on reinforcement learning, planning and teaching. Machine learning, 8(3), 293-321.

Mnih, V., Kavukcuoglu, K., Silver, D., Rusu, A. A., Veness, J., Bellemare, M. G., ... & Petersen, S. (2015). Human-level control through deep reinforcement learning. Nature, 518(7540), 529-533.

Lillicrap, T. P., Hunt, J. J., Pritzel, A., Heess, N., Erez, T., Tassa, Y., Silver, D., & Wierstra, D. (2016). Continuous control with deep reinforcement learning. ArXiv, abs/1509.02971.

Henderson, P., Islam, R., Bachman, P., Pineau, J., Precup, D., & Meger, D. (2018). Deep reinforcement learning that matters. In Proceedings of the AAAI Conference on Artificial Intelligence (Vol. 32, No. 1).

Toon, Chris. (2019). Retrieved from <https://towardsdatascience.com/deep-deterministic-policy-gradients-explained-2d94655a9b7b>