USING MACHINE LEARNING TO ESTIMATE CUSTOMERS' VALUATIONS IN A SINGLE PRODUCT PRICING PROBLEM

by
KHOSRO PARVIZ NADERI VARANDI

Submitted to the Graduate School of Engineering and Natural Sciences
in partial fulfillment of
the requirements for the degree of Master of Science

Sabancı University
July 2023

ABSTRACT

USING MACHINE LEARNING TO ESTIMATE CUSTOMERS' VALUATIONS IN A SINGLE PRODUCT PRICING PROBLEM

KHOSRO PARVIZ NADERI VARANDI

INDUSTRIAL ENGINEERING M.SC. THESIS, July 2023

Thesis Supervisor: Asst. Prof. Ezgi Karabulut Türkseven
Thesis co-supervisor: Asst. Prof. Burak Gökgür

Keywords: Pricing, Machine Learning, Optimization, Customers' Valuation

Estimating customers' valuations is a goal in retail to learn the customers' willingness to pay for a product so that the seller can set better price levels and in turn increase revenue. In a single product setting, the strategy determines the optimal price level that generates the highest revenue (reward). A key criterion that affects these prices is the customer profiles in the market. In e-commerce businesses, there is big data consisting of the customers' sales information. Analyzing and processing this data will provide further insight into the market and the customer profiles, and therefore is promising to contribute to increasing sales revenues. This thesis aims to estimate the customers' valuations and find the optimal price that brings the highest revenue by applying machine learning algorithms to customer sales information.

Throughout this thesis, a range of algorithms has been developed to estimate customer valuations, considering various market structures and the level of prior knowledge available. These market structures encompass parameters such as the number of segments, the segment distributions, and their order, and scenarios such as having deterministic or probabilistic valuations. In the offline setting, where a full dataset is available, the algorithms are designed to obtain initial estimates of valuations, including their lower and upper bounds when they are probabilistic, the distribution of segments, the order of these distributions, and the number of segments in the market. Furthermore, algorithms have been developed for the online setting, where sales information is gathered gradually. The primary objective in this scenario is to increase the seller's revenue through ongoing analysis and decision-making. The results obtained from the proposed algorithms and methods indicate their success in estimating the market structure and achieving high revenues, which closely align with the expected revenue. These findings highlight the effectiveness and efficiency of the developed algorithms in estimating customer valuations and optimizing revenue for the seller. Overall, the research demonstrates the value of the developed algorithms and methods in estimating crucial parameters, optimizing revenue, and achieving revenue levels that are in line with expected revenue values which we expect to achieve based on the true values of the market structure. These findings have significant implications for businesses operating in different market structures, enabling them to make informed pricing decisions and increase their overall revenue potential.

ÖZET

TEK ÜRÜNLÜ FIYATLANDIRMA PROBLEMINDE MAKINE OĞRENMESI
ILE MÜŞTERILERIN DEĞERLENDIRME TAHMINI

KHOSRO PARVIZ NADERI VARANDI

ENDÜSTRİ MÜHENDİSLİĞİ YÜKSEK LİSANS TEZİ, TEMMUZ 2023

Tez Danışmanı: Yar. Doç. Ezgi Karabulut Türkseven
Tez Eş Danışmanı: Yar. Doç. Burak Gökgür

Anahtar Kelimeler: Fiyatlandırma, Makine Öğrenmesi, Eniyileme, Ürün Değerlemeleri

Müşterilerin değerlemelerini tahmin etmek, perakendede müşterilerin bir ürün için ödemeye istekli oldukları tutarı öğrenmek amacıyla, satıcıların daha iyi fiyat seviyeleri belirlemelerine ve dolayısıyla geliri artırmalarına yardımcı olan bir hedeftir. Elde edilmek istenen strateji tek ürünlü pazarda en yüksek geliri (ödülü) üreten fiyat seviyesini belirler. Bu fiyat seviyesini etkileyen en temel kriter, pazardaki müşteri profilleridir. E-ticaret işletmelerinde, müşteri satış bilgilerinden oluşan büyük veri bulunmaktadır. Bu verinin analizi ve işlenmesi, pazar ve müşteri profilleri hakkında daha fazla içgörü sağlama ve dolayısıyla satış gelirlerini artırmaya katkıda bulunma potansiyeli taşımaktadır. Bu tez, müşterilerin değerlemelerini tahmin etmeyi ve en yüksek geliri getiren en uygun fiyatı bulmayı amaçlamakta olup, bunu yapmak için makine öğrenmesi algoritmalarını müşteri satış bilgilerine uygulamayı amaçlamaktadır.

Bu tezde, müşteri değerlemelerini tahmin etmek için çeşitli algoritmalar geliştirilmiştir; farklı pazar yapıları ve mevcut önbilgi düzeyleri dikkate alınmıştır. Bu pazar yapıları, segment sayısı, segment dağılımları ve sıralamaları gibi parametreleri kapsar ve deterministik veya rassal değerlemelerin bulunması gibi senaryoları içerir. Çevrimdışı öğrenme adımlarında, algoritmalar değerlemelerin başlangıç tahminlerini elde etmek üzere tasarlanmıştır; bunlar, rassal olduğunda dağılımın alt ve üst sınırlarını, segment dağılımını, bu dağılımların sırasını ve pazardaki segment sayısını içerir. Ayrıca, satış bilgilerinin kademeli olarak toplandığı çevrimiçi öğrenme algoritmaları geliştirilmiştir. Bu algoritmaların temel amacı, sürekli analiz ve karar verme yoluyla satıcının gelirini artırmaktır. Önerilen algoritmalar ve yöntemlerden elde edilen sonuçlar, pazar yapısını tahmin etme ve yüksek gelir elde etme konusundaki başarısını göstermekte olup, beklenen gelire yakın sonuçlar ortaya koymaktadır. Bu bulgular, geliştirilen algoritmaların müşteri değerlemelerini tahmin etme ve satıcı için geliri optimize etme konusundaki etkinliğini ve verimliliğini vurgulamaktadır. Genel olarak, araştırma geliştirilen algoritmaların ve yöntemlerin önemli parametreleri tahmin etme, geliri enbüyükleme ve beklenen gelir değerlerine uyumlu gelir seviyelerine ulaşma konusundaki değerini göstermektedir. Bu bulgular, farklı pazar yapılarında faaliyet gösteren işletmeler için önemli sonuçlar doğurur; bu sonuçlar, bilinçli fiyatlandırma kararları alabilmelerini ve genel gelir potansiyellerini artırmalarını sağlar.

ACKNOWLEDGEMENTS

You don’t get to do valuable work without standing on the shoulders of a mountain of people, so I have many incredible people to thank. First, I would like to express my deepest gratitude to my supervisors professor Ezgi Karabulut Türkseven and professor Burak Gökgür, this research work would not have been possible without their guidance and their attention. I would also like to thank all of my professors and instructors at Sabancı University for the enlightening education I received from them.

I am also extremely grateful to the Scientific and Technological Research Council of Turkey (TÜBİTAK) for supporting this work (project number: 221M392).

Dedication page
I dedicate this work to my future self.

TABLE OF CONTENTS

LIST OF TABLES

LIST OF FIGURES

1. Introduction and Literature Review

Companies operating in the retail sector use different pricing strategies intensively in order to increase the revenue obtained from the products they offer to their customers. With the developments in information technology and data science, it is observed that the methods retailers deploy for presenting prices to their customers have changed significantly. In particular, retailers that expand their operations to online sales channels can learn about customers' buying behaviors by analyzing their purchasing preferences and hence design meaningful and effective methods for offering prices to different customer groups. Tesco, a UK-based retail company, analyzes the shopping carts of its customers, creates customer segments, and offers price discrimination policies to its customers (Clive, Terry & Tim, 2004; Davenport, Mule & Lucker, 2011). (Davenport et al., 2011) proposes a method for predicting customer needs based on data analysis. The authors suggest that by analyzing patterns in customer behavior and using machine learning techniques, companies can gain insight into their customers' future needs and preferences. This allows them to tailor their offerings and marketing strategies to better meet the evolving needs of their customers, leading to increased customer satisfaction and loyalty. Large retail companies such as Safeway and Kroger also offer different price levels by monitoring their customers' purchasing activities in real-time through online channels (Clifford, 2012; Farnham, 2013). In Safeway's "just for U" program, digital coupons for personalized deals are extended to individual consumers either online or through the "just for U" app. The CEO of Safeway, Steve Burd, told analysts in 2013 (Ross, 2013): "There's going to be a point where our shelf pricing is pretty irrelevant because we can be so personalized in what we offer people." According to the results of a survey conducted by Accenture in 2016, it has been observed that 58% of customers tend to respond positively to price promotions designed using their past purchase information through online channels (Accenture, 2016). With the rapid growth of online sales (According to the analysis of eMarketer, a market research company, it is predicted that e-commerce sales will grow by 8.9% worldwide and have a volume of approximately six trillion dollars by the end of 2023 (eMarketer, 2023)), companies seize unique opportunities to both improve the customer experience and increase revenues. Firms with sales operations in their online channels can collect a large amount of transactional data about the purchasing behavior of customers. The opportunity to instantly monitor this data and the reaction of customers to current offered prices are of great importance for e-commerce companies to learn product valuations (the highest price level that customers will be willing to pay to buy the relevant product) and to develop effective price policies to increase their revenue.

This research aims to learn the customers' valuations (demand learning) in order to increase the seller's revenue for a retail firm that sells a single product in an online channel. The main aim of this research is to analyze the interaction between pricing and learning decisions in the presence of a multi-segment market structure with a diverse customer pool for a product that the seller tries to increase revenue for by considering the problem in the context of a learning mechanism. The objectives stated within the scope of this research are achieved in two stages:

- In the first stage of this research which is a warm-up phase, the efficiency of the single pricing mechanism in the process of learning the parameters of the problem are examined assuming that the customers' valuations are unknown by the retailer. The criterion to be taken into account in this learning process is the accuracy of the estimated parameters.

- In the second phase which is called online, followed by the offline phase, the estimated parameters from the warm-up period is used to maximize revenue while gaining knowledge about the parameters of the problem through the learning method to be developed.

The research questions to which we will seek answers in this research are given below:

- How effectively can machine learning-based algorithms solve the pricing problem?

- How much does the learning problem slow down with less market prior knowledge?

- How much does the accuracy of developed algorithms drop with less market prior knowledge? What is the importance of market prior knowledge?

- How successful are the approaches with deterministic methods in cases where there are random product valuations?

Algorithms are implemented within the framework of reinforcement learning in the two phases of the research. Most of the algorithms in the reinforcement learning literature are designed for models with discrete structures unlike the problem discussed in this research. Assumptions such as differentiability and noise level variance stipulated by the proposed algorithms for models with a continuous structure such as our problem are invalid for the reward function in this project. For this reason, it is not possible to directly apply an existing algorithm in the reinforcement learning literature. Within the scope of this research, reinforcement learning algorithms will be designed for problems with reward functions in a piecewise linear function structure. The algorithm presented in the machine learning literature for learning piecewise linear functions (Ferrari-Trecate & Muselli, 2002) is solved as an offline learning problem. Reinforcement learning algorithms, on the other hand, solve online problems by nature. Our aim is to design an online learning algorithm with a piecewise linear function and use it as a module in the reinforcement learning algorithm.

This thesis makes significant contributions to the fields of dynamic pricing and learning customers' product valuations in both operations management and computer science literature. In the operations management literature, traditional pricing models often rely on complex mathematical models and assumptions to find optimal prices for revenue maximization. However, this work offers a more general and versatile framework for pricing, applicable to any product and retail domain. It avoids relying on numerous assumptions and can be applied in a wide range of retail scenarios, making it more practical and adaptable. In the computer science literature, existing works on machine learning for pricing problems often focus on customized customer attributes. In contrast, this research presents a more inclusive and generalized framework that is independent of customer attributes but relies solely on their buying preferences. This learning mechanism is designed to estimate customer valuations and set price levels to maximize seller revenue effectively. The research further contributes to the development of machine learning algorithms for estimating customers' valuations and maximizing revenue. Instead of using existing algorithms, various novel algorithms have been proposed throughout the thesis, addressing different stages of the problem. These algorithms are easy to understand, implement, and apply to online channel data, requiring only offered prices and customer buying preferences. Additionally, the thesis investigates the problem under various scenarios and levels of prior market knowledge, providing valuable insights into pricing strategies and revenue maximization in different market structures. In summary, the contributions of this thesis can be summarized as follows:

- Presenting a general framework for product pricing based solely on offered prices and customers' buying preferences, applicable across diverse retail domains.

- Developing user-friendly machine learning algorithms for estimating relevant parameters and maximizing revenue.
- Exploring different market scenarios and levels of prior knowledge to gain deeper insights into pricing strategies.
- Offering valuable knowledge and strategies for pricing in various market structures.

It is possible to divide the researches in the operations management literature into two groups as studies in which the supply is unlimited and limited. In this project, the focus is on researches in the relevant literature where the supply is unlimited. The studies that assume that the product inventory is not limited are taken into account in the learning literature since the product inventories are available whenever the demand arises. In addition, after the studies that assume unlimited supply for the product, studies that consider the limited inventory scenario are also presented in order to more clearly indicate the contribution of this work to the learning literature. (Billstroem & Thore, 1964) and (Thore, 1964) first studied the pricing and learning problem in the operations management literature. In these two studies, it is presented with which pricing rules a monopolist firm can increase its income in the presence of a linear demand function whose parameters are unknown. (Thore, 1964) presents a dynamic pricing approach with the notion that if a prior price increase resulted in an increase in revenue, the price should be increased again, and vice versa. Similarly, if a prior price decrease led to an increase in revenue, the price should be decreased again, and vice versa. (Thore, 1964) also suggests that the magnitude of the price adjustment should depend on the difference between the last two revenue observations. Two pricing rules are specified, and the resulting dynamical systems' convergence properties are analyzed. (Billstroem & Thore, 1964) perform simulation experiments for one of these pricing rules. They test the rule in both a deterministic demand setting and a scenario where a normally distributed disturbance term is added to the demand. Furthermore, they extend the model to incorporate inventory replenishment and provide a rule of thumb for choosing an optimal constant for the pricing rule. As a result of these two studies, the learning problem of a product's demand with different pricing policies has attracted the attention of researchers. (Baetge, Bolenz, Ballwieser, Hömberg & Wullers, 1975) expanded upon the simulated findings of (Billstroem & Thore, 1964) by examining non-linear demand curves and analyzing the optimal selection of a constant in the pricing rules. (Aoki, 1973) studied the pricing problem of a firm that cannot observe the parameters of the demand function of the product it offers to the market.

Using the methods in stochastic adaptive control theory, he examined the efficiency of learning the demand function with the Bayesian approach. The objective is to minimize the surplus demand function, and the author demonstrates the theoretical possibility of computing the optimal Bayesian approach using dynamic programming. However, in numerous scenarios, a closed-form solution is not available. To overcome this, the author proposes two alternative approximation policies. The first method, Certainty Equivalent Pricing (CEP), selects the optimal price for each decision moment based on current parameter estimates. The second method, referred to as an approximation under static price expectation, involves the firm acting as though the chosen price will remain throughout the remainder of the selling period. Following this study, studies of learning the demand for a single product with the Bayes method were conducted in different problem frameworks. (Chong & Cheng, 1975) formulated the optimal pricing problem as a Bayesian dynamic program in which they assume the demand function is linear with two unknown parameters and normally distributed disturbance terms. They show when the slope of the demand function is known, certainty equivalent pricing is the optimal policy. For the unknown slope and intercept case, they propose three approximations to the optimal policy. (Nguyen, 1984) explored the behavior of a monopolistic firm in a market with random demand. The firm faces uncertainty about the demand and can learn about it over time using Bayesian learning. The study shows that the firm's pricing strategy depends on the degree of uncertainty about demand and the learning speed. When the firm is highly uncertain about demand, it sets a high price to compensate for the risk. However, as the firm learns about demand, it reduces the price to increase market share. The learning speed affects the firm's pricing strategy, and the more quickly the firm learns, the more aggressive it can be in setting a low price. The paper also discusses the implications of the findings for firms in real-world markets. (Nguyen, 1997) also studied demand learning for a single product with the Bayes method. (Wruck, 1989) considers optimal pricing of durable and non-durable goods in a two-period model in which the distribution of willingness-to-pay is learned by Bayesian updating a uniform prior. The optimal price policy is determined by solving a dynamic program. (Lobo & Boyd, 2003) consider the same setting as (Chong & Cheng, 1975) and compare the performance of four pricing policies with each other by simulation. (Qu, Ryzhov & Fu, 2013) introduced a pricing model based on Bernoulli distribution, where the expectation of demand is a logit function of price, and the unknown parameters are subject to a normal prior. However, due to the complexity of this distributional form, the authors discussed the challenges of calculating posterior distributions and proposed an approximation method. Additionally, they introduced a Bayesian-greedy price policy, which is also computationally challenging, and provided a method to compute an approximation.

To compare the performance of the proposed pricing policies, the authors conducted a numerical study and compared them with a few alternative policies. (Kwon, Lippman & Tang, 2012) investigated the optimal markdown pricing strategy for a retailer who faces uncertain demand and can learn about it over time. The study develops a model that incorporates Bayesian learning and dynamic pricing to determine the optimal markdown price. The results suggest that the retailer should set a high initial price and then reduce it over time as demand uncertainty decreases. The optimal markdown price depends on the learning speed, the initial price, and the distribution of demand uncertainty. The paper also provides insights into how the optimal markdown pricing strategy changes with different market conditions and shows that the retailer can achieve higher profits by using the proposed pricing strategy than by using a static pricing strategy. The findings of the study are relevant for retailers who face dynamic market conditions and can use pricing strategies to maximize their profits. In (Birge, Chen & Keskin, 2021)'s study, the pricing problem of a company selling a product to a combination of forward-looking and myopic customers is examined. The company faces uncertainty regarding the customers' behavior, arrival pattern, and product valuations, which together comprise the demand model. The company implements markdown policies over multiple selling periods, sequentially lowering the product price and observing demand to gain insight into the demand model. They analyze these policies to determine their effectiveness in accommodating a range of forward-looking customer behaviors. The study is conducted in a Bayesian framework where the seller has a prior belief distribution on different demand models, and the effectiveness of the policies is measured by the expected cumulative profit loss compared to a clairvoyant who has complete knowledge of the demand model.

In addition to the Bayesian method, studies that adopt the parametric approach have also taken their place in the learning literature in operations management. (Lobo & Boyd, 2003) examine a linear demand function expressed as $d_t(p_t) = a - b(p_t) + e_t$, where a and b are unknown and assumed to have priors with normal distributions, and e_t is a normal distribution with known parameters. The authors evaluate three pricing policies: (i) a static policy that uses the prior distributions without updating them, (ii) a myopic policy that updates the distributions but only focuses on the current period without considering future periods, and (iii) a myopic policy with dithering, which adds a random perturbation to the price to stimulate the learning process and increase the information gained. The authors demonstrate through numerical examples that the myopic policy with dithering generates higher revenue than the static or myopic policy. (Carvalho & Puterman, 2005) analyze a dynamic pricing scenario where the demand follows a logistic distribution characterized by two unknown parameters. Through a numerical assessment of various heuristic approaches, the authors show that a "one-step lookahead" policy, which forgoes immediate revenue in favor of obtaining a more precise estimate of the unknown demand parameters, performs better than a myopic policy. (Broder & Rusmevichientong, 2012) assumed that the demand for the product sold by a monopolist firm is Bernouilli-distributed, and the distribution parameters are not known by the firm. The aim of the company is to determine a price policy that will minimize the loss of income due to the inability to observe the demand parameters. In this study, a price policy based on the maximum probability estimation that works effectively under certain conditions is developed, and a lower bound study on price policy is presented. (den Boer & Zwart, 2014) explore a broader range of generalized linear single-product demand models and demonstrate through a specific example that the practice of certainty equivalent pricing lacks strong consistency. To address this issue, they suggest a pricing strategy that selects the price closest to the certainty equivalent price while ensuring a minimum level of price dispersion. This dispersion, as measured by the sample variance of selling prices, ensures that the prices converge toward the optimal price. (Den Boer, 2014) extends this policy to multiple products and assumes that the parameters of the demand function for products are not known by the firm. He has developed a pricing policy that updates the optimal price level to certain rules according to parameter estimates obtained by using statistical estimation techniques in each time period. (Bu, Simchi-Levi & Xu, 2020) studied the impact of pre-existing offline data on online learning in dynamic pricing. A single-product dynamic pricing problem over a T-period selling horizon is studied, where demand in each period is influenced by the product price using an unknown parameter linear demand model. The seller has some pre-existing offline data prior to the start of the selling horizon and aims to minimize regret of the online learning process by utilizing both pre-existing offline data and sequential online data. The optimal regret rate of the online learning process is characterized based on the size, location, and dispersion of the offline data.

There are also studies that take into account nonparametric demand models. (Kleinberg & Leighton, 2003) examined algorithms for determining prices in a basic market scenario, where a seller has an infinite supply of a single good and interacts with a group of n potential buyers one at a time. The seller proposes a price between 0 and 1 for each transaction, and the buyer decides whether to accept or reject the offer based on their own private valuation of the good. They tried to answer the question of how valuable it is for the seller to know the demand curve. In this regard, they analyzed three cases - identical, random, and worst-case valuations and derived upper and lower bounds for each case. (Lim & Shanthikumar, 2007) consider dynamic pricing strategies that are robust in the face of model uncertainty, using the classical pricing framework of (Gallego & Van Ryzin, 1994). They take a robust approach, where the demand function is not learned over time but assumed to lie in some known uncertainty set. (Eren & Maglaras, 2010) examined two scenarios where the initial inventory is infinite, and the time horizon is finite. In these scenarios, a willingness-to-pay demand model with an unknown WtP distribution function F is assumed. The aim is to maximize the worst-case competitive ratio across all possible F, which is defined as the ratio of expected revenue generated by a pricing strategy without knowledge of F to the expected revenue that could have been obtained if F was known. In the first scenario, there is no learning involved, whereas in the second scenario, the time horizon is split into two periods, with price experimentation permitted in the first period but not the second. For both situations, optimal pricing policies are determined through closed-form calculations. The results indicate that even experimenting with prices at a few points can significantly improve revenue performance. Additionally, the relative benefit of ongoing learning diminishes beyond a certain point. (Besbes & Zeevi, 2012) explore a multi-product problem with a finite inventory of each product and a finite time horizon. The demand function is unknown, except that it satisfies certain basic conditions. Their goal is to maximize the worst-case competitive ratio. They analyze two versions of the problem: one with discrete allowable prices and the other with continuous allowable prices. They present an algorithm that consists of an exploration phase and an exploitation phase. For the discrete allowable price case, they create a simple linear programming-based policy that determines the fraction of time each allowable price point should be applied. For the continuous allowable price case, they choose the "best" price to be applied throughout the exploitation phase. They prove that all the policies they develop are asymptotically optimal when the volume of sales grows large. (Cheung, Simchi-Levi & Wang, 2017) discuss a dynamic pricing problem where the demand function is not known, and price experimentation is used as a demand learning tool. However, sellers often face constraints on price changes in practice. The article proposes a dynamic pricing model where the demand function is unknown but belongs to a finite set, and the seller can make at most "m" price changes during "T" periods. The objective is to minimize the worst-case regret, and a pricing policy that incurs a regret of $O(log(m)T)$ is proposed. The pricing policy is implemented at Groupon, and a field study shows a significant impact on revenue and bookings. (Ferreira, Simchi-Levi & Wang, 2018) examined a revenue management problem faced by retailers who aim to optimize their revenue from multiple products with limited inventory over a finite selling season. The problem involves unknown demand parameters that require learning from sales data, resulting in an exploration-exploitation trade-off. The authors introduced a class of dynamic pricing algorithms based on the machine learning technique of Thompson sampling to address this trade-off under the presence of inventory constraints. The proposed algorithms demonstrated both strong theoretical performance guarantees and promising numerical performance results compared to other similar algorithms. (Miao & Wang, 2021) focus on the price-based network revenue management (NRM) problem with demand learning. The retailer's objective is to determine the prices of n products dynamically over a finite selling season, subject to m resource constraints, to maximize cumulative revenue. This paper proposes a nonparametric demand model with mild technical assumptions that are typical of commonly used demand functions. The study introduces a novel robust ellipsoid method adapted to the NRM setting. Overall, this paper offers novel insights and efficient solutions for addressing the NRM problem with demand learning.

In the operations management learning literature, there are also pricing and learning studies in the presence of limited inventory. (Lazear, 1984) examines a basic model in which a company sells a single product for a maximum of two periods. In the initial period, a group of potential customers visit the store, and if none of them make a purchase, the company adjusts their prior belief about the product's value, updates the selling price, and attempts to sell it during the second period. Through this model, the author demonstrates that the expected profit can be enhanced by having two selling periods instead of one. Furthermore, the study extends the model in various ways, including incorporating strategic behavior by customers who may delay their purchase decision in anticipation of a potential price reduction. (Sass, 1988) expands on Lazear's model and explores the relation between the optimal pricing strategy and the number of potential buyers. (Aviv & Pazgal, 2006) initiated a research stream on the application of Bayesian learning in the context of dynamic pricing with finite inventory. The authors considered a scenario where customers arrive according to a Poisson process with an unknown arrival rate and purchase a product with a known probability which is a function of the current selling price. The authors utilized Bayesian updates of a Gamma prior to learn the unknown arrival rate and subsequently derived a differential equation to characterize the optimal continuous-time pricing policy. (Besbes & Zeevi, 2009) start a stream of literature that attempts to learn the optimal static price in an incomplete information setting. They analyzed the pricing problem of a single product offered to customers by a monopolist firm within a finite planning horizon in order to maximize its expected revenue. In the problem environment, it is assumed that the basic relationship between price and demand function is unknown. They developed single product pricing policies that learned the demand function and determined price levels that would maximize expected revenue. In addition, they presented performance analyses related to the learning policies developed. (Ferreira et al., 2018) address a price-based network revenue management problem. In the problem environment, it is aimed to maximize the expected revenue to be obtained from the sale of products in a limited planning horizon of a retailer that has more than one product with limited inventory. They assume that the parameters of the demand function are not known to the retailer. They predicted that these parameter values will be learned from sales data. For this purpose, they presented pricing algorithms using Thompson sampling. In addition, it has been shown that the developed algorithms achieve both strong theoretical performance guarantees and promising numerical performance results. (Miao & Chao, 2021) discussed the product variety and pricing problem together. The purchasing behavior of the customers is designed with the multi-state preference model (Multinomial Logit). They developed a learning algorithm that balances demand learning and revenue and compared the performance of this algorithm with the assumption that the demand function is known. An upper limit is presented for the developed algorithm, and it is shown with numerical results that the algorithm performs very well.

The computer science community has contributed significantly to the literature on dynamic pricing and learning. Rather than providing mathematical analyses of pricing policies, these researches tend to focus on the design of realistic models for electronic markets and the application of machine learning techniques. This approach offers the advantage of being able to model a wide range of demand-influencing phenomena, including competition, fluctuating demand, and strategic buyer behavior. However, a potential drawback is that these models can be excessively complex to analyze analytically, and insights into the performance of various pricing strategies can only be obtained through numerical experiments. (Shakya, Oliveira & Owusu, 2009) examine the impact of demand uncertainty on dynamic pricing using Evolutionary Algorithms (EAs). The study uses a realistic stochastic model to analyze the performance of different EAs in dynamic pricing problems under various demand scenarios. The results show that EAs can be effective in optimizing pricing strategies in uncertain market conditions and can outperform traditional pricing methods. Furthermore, the paper suggests that higher demand fluctuations may not necessarily harm a firm's profitability and that EAs can be more reliable in finding accurate pricing policies in high demand fluctuation scenarios. (Ramezani, Bosman & La Poutré, 2011) also deployed evolutionary algorithms for dynamic pricing. (Shakya, Kern, Owusu & Chin, 2012) Combined neural networks and evolutionary algorithms together to optimize pricing policies. They built a neural network-based demand model and used evolutionary algorithms to optimize policy over the build model. (Gupta & Pathak, 2014) developed a general architecture using machine learning models for dynamic pricing by adapting personalized adaptive pricing in online retail which can be a primary driver for online purchases. The study proposes a framework using machine learning, data mining, and statistical methods to predict customer purchase behavior and select an appropriate price range based on dynamic pricing. The framework is tested on a large dataset for an e-commerce firm and shows encouraging results, reducing error rates and determining better price ranges appropriate for both the customer and the organization. The study focuses on predicting purchase decisions based on adaptive or dynamic pricing of a product, using different data sources such as visit attributes, visitor attributes, purchase history, web data, and context understanding. The framework can be applied to various industries working in online mode and can be customized to specific applications. (Javanmard & Nazerzadeh, 2019) discuss a pricing problem faced by a company that sells a large number of products with various features to customers over time. The company aims to maximize revenue by offering prices that depend on customer and product characteristics. The article proposes a Regularized Maximum Likelihood Pricing (RMLP) policy that leverages the structure of demand parameters to obtain a logarithmic regret in T. The policy's regret scales linearly with the sparsity of the optimal solution and logarithmically with the dimension. (Ban & Keskin, 2021) study the dynamic pricing of a product that its price can be adjusted by the seller at the individual customer level by using information about customers' characteristics. They assume a personalized demand model in which the parameters are dependent on the customers' characteristics and the relationship between the customer features and the product demand can be learned by the seller through sales observations over a selling horizon. They design near-optimal pricing policies for tackling the problem using machine learning techniques.

Our work lies in the unlimited supply and nonparametric category, and we will develop algorithms in the framework of machine learning techniques to solve the problem of pricing a single product in the presence of a multi-segment market structure considering different aspects of customers' behavior such as the valuations being deterministic or probabilistic and market prior knowledge to maximize the revenue for the seller.

In summary, the original value of this research can be listed as follows:

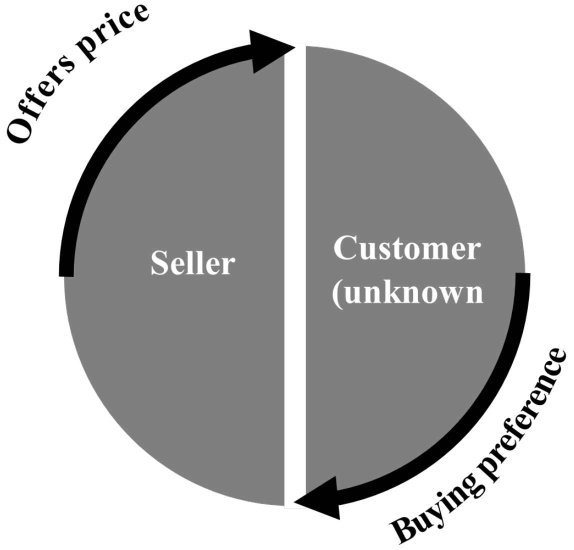
- [Interdisciplinary] This research, which will examine an operations management problem with a strong machine learning infrastructure will set an example for the contribution of interdisciplinary work both in the academic world and in the world of practice. It is expected that this analysis, where none of the mentioned areas will be sufficient alone, will lead to machine learning cooperation with different problems from the field of operations management in the future.

- [Contribution to the Academic Literature of Operations Management and Machine Learning] Even though our research is basically in a position to provide analyses on pricing policies, the tools that provide these analyzes will not be ready-to-use and 100% compatible algorithms for the problem we are trying to solve. Different algorithms based on the different levels of knowledge we have regarding the customers' behaviors and the market structures will be developed and mathematical problems that are solved from time to time will be adapted to our model.

2. Problem Definition

The learning model we plan to use in our problem consists of two decision makers, a seller, and an unlimited customer cluster. The dynamic between the seller and the customer works as follows as shown in Figure 2.1: In each iteration, the seller determines the price for the product in his hand and presents this price to the incoming customer. This product has a valuation to the customer, and the customer based on his valuation for the product and the offered price chooses to buy the product or not. Essentially, if the offered price to the customer is less than the customers's valuation, then that customer would buy the product, otherwise, he would not buy the product. These valuations are the highest price that customers are willing to pay for the product which we refer to them as customers' valuations $v_i (i = 1, \ldots, n)$. The information conveyed to the seller is the information on whether the customer buys the product or does not buy the product. Using this preference information, the seller updates its price in the next iteration and offers the product at a new price to the next customer from the unlimited customer cluster. In each iteration, only one customer's preference information is learned. It is generally accepted that the customer cluster for which we have limited information is divided into subsets called segments (n segments), that customers in the same segment have the same or similar preference profile, and customers in different segments have different preference profiles. Depending on the application areas, the number of segments, the probability distributions of the segments, and the limited preference profiles of this cluster can be known. However, the segment to which a specific customer belongs cannot be known until the customer informs the purchase decision. Therefore, there is no option to customize the price offered by the seller according to the segment of the customer. Such customer-seller interactions are common in the e-commerce industry (Ferreira et al., 2018; Miao & Chao, 2021).

Figure 2.1 Information flow between seller and customer



The goal here is to find the price that maximizes the seller's revenue. For the seller to find the optimal price, he needs to know sets of parameters: the number of segments that customers are coming from, valuations of the customers, and their distribution in the segments. By knowing these, the seller can easily find the optimal price as follows:

$$P^* = \arg \max_{i=1,\ldots,n} \{(1 - \sum_{j=1}^{i-1} \delta_j) \times v_i\} \tag{2.1}$$

In which n is the number of segments, v_i is the valuation for customers in segment i, and δ_i is the probability of being in the segment $i(i = 1, \ldots, n)$ for customers in that segment.

We investigate the problem in two phases. In the first phase referred to as Offline learning, We are provided with a dataset with size T as $\{(x^t, y^t)\}$ $(t = 1, ..., T)$ in which x^t is the price offered by the seller, and y^t is the feedback of the customer (1 for buying, and 0 for not buying). The primary objective of this phase is to solely learn the parameters of the problem. Accordingly, we will develop algorithms for tackling the problem considering various levels of market structure knowledge available to us. These include the known or unknown number of segments, known or unknown probability distribution of segments, deterministic or probabilistic customers' valuations, and uniform or non-uniform customers' valuations.

The subsequent phase, known as Online Learning, differs from the Offline Learning phase in that we no longer have access to a complete dataset. Instead, customers arrive individually, one at a time. Upon the arrival of each customer, we employ the estimated parameters obtained from the Offline Learning phase to determine the price offered to them. The ultimate goal in this phase is to maximize the seller's reward. Similarly to the Offline Learning phase, the development of distinct algorithms is contingent upon the prior knowledge we possess regarding the market structure. These algorithms are tailored to adapt to various levels of market structure knowledge, thereby facilitating effective decision-making in the pricing process.

Our research approach involves a progressive reduction of prior knowledge regarding the market structure, starting from the highest level of knowledge and gradually eliminating it. This progressive elimination of knowledge is aimed at making the problem increasingly challenging until we reach a point where no information regarding the market structure is available. At this point, the objective is to determine the optimal price for maximizing the seller's reward, despite the absence of any prior knowledge about the market structure. By systematically examining the problem under different levels of knowledge, we aim to enhance our understanding of pricing strategies and optimize the seller's revenue in scenarios where limited or no information about the market structure is available.

3. Offline Learning

The primary objective of the Offline Learning phase, as previously stated, is to learn the unknown parameters associated with the problem. In this phase, our focus is not on offering prices to maximize rewards but rather on utilizing the given dataset to learn the underlying parameters. This phase can be regarded as a warm-up stage where the estimated parameters by offline learning will be passed to the next phase (online learning) for reward maximization. We will develop algorithms tailored to different levels of foreknowledge concerning the market structure. These algorithms will be applied to the dataset of size T, enabling us to estimate and learn the remaining unknown parameters. The subsequent sections of this chapter will delve into the investigation of the problem under various scenarios, each exploring different aspects and levels of market structure knowledge. By thoroughly examining these scenarios, we aim to shed light on effective parameter estimation techniques in the context of machine learning and pricing.

3.1 Case where the number of segments (n), segment distributions (δ) and their order are known

The problem we will examine in the first step will be the problem of learning the selling price that maximizes the total revenue in the case of a single product type (A) and n customer segments. The defining features of the customer segments defined between $i = 1,...,n$ are the value they attach to product A v_{Ai} and the probability of a customer being in segment i δ_i. Without loss of generality, we can assume that the segments are in ascending order according to their v_{Ai} values $v_{A1} < v_{A2} < ... < v_{An}$. In this scenario with the highest level of foreknowledge, we assume the number of segments n, segments' distribution δ_i's, and their order are known, and the valuations are deterministic. Here, the ordering refers to the correspondence between each δ value and its respective segment. In this scenario only the product valuations are unknown. Our aim is to learn and estimate these values.

Since we do not have access to a real dataset for analysis, we will generate synthetic data to facilitate our study. Specifically, we will create a dataset denoted as $\{(x^t, y^t)\}$, comprising T data points. The continuous prices x^t will be generated within the range of $(0, 100)$, while the buying preferences y^t will take binary values of either 0 or 1 (1 for buying the product, and 0 for not buying the product). Consequently, the synthetic preference dataset will exhibit a structure resembling that shown in Table 3.1:

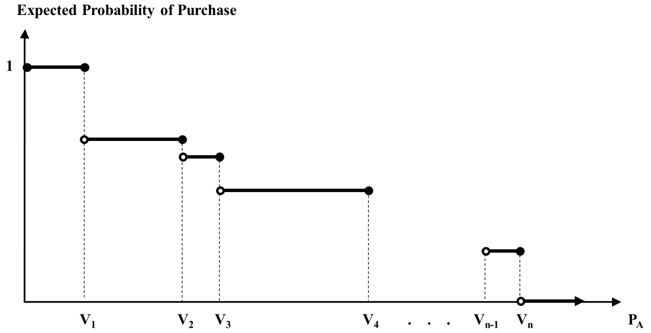
Table 3.1 Synthetic preference dataset

Offered Prices	Buying Preferences
x^1	1
x^2	0
x^3	1
.	.
.	.
.	.
x^T	0

3.1.1 Proposed Algorithm 1

The learning problem we propose in this scenario is based on learning the parameters of the pricing problem (customers' valuations) through the expected probability of purchase function shown in Figure 3.1. The first challenge we encounter during the learning problem using preference data is the process of converting the preference data with the structure 0-1 (the customer bought or did not buy when the price x^t was suggested) to the expected function data (the probability of the customer to buy when the price x^t is offered). As mentioned earlier, the dataset we collect in the current system has the binary structure (x, y) for each suggested price x and preference information $y \in \{0, 1\}$.

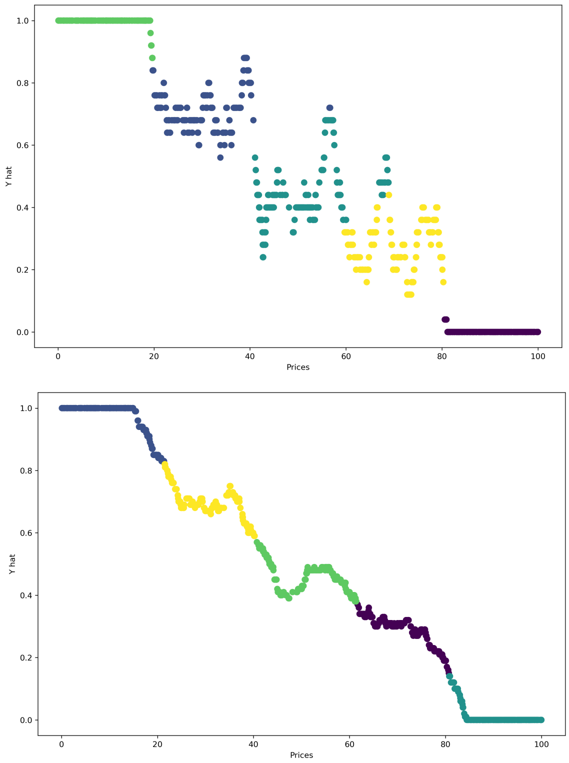
Figure 3.1 Expected purchase probability as a function of suggested price in the presence of deterministic product valuations



In cases where the suggested price options are discrete, the expected value can be taken as the average of the y values associated with the same price x, but this approach loses its validity when the suggested prices are a continuous set. The method we apply as a solution is the k - Nearest Neighbors (kNN) algorithm, which is frequently used in the world of machine learning (Kramer & Kramer, 2013). Taking the value of k as a hyperparameter, the algorithm takes the k suggested price values closest to x for each x-value and assigns the average ($\hat{y}$) of the y-values corresponding to those prices as the expected purchase probability of x. Although this method works well, its performance is highly dependent on the hyperparameter k. Figure 3.2 shows the $(x^t, \hat{y}^t)$ values obtained when kNN is applied with different k values to the $\{(x^t, y^t)\}$ dataset consisting of $T = 1000$ data points. As can be seen in Figure 3.2(a), choosing a low value of k increases the volatility in the estimated probability of purchase values while choosing a high value of k subtracts the function estimating the expected probability from the cascading function structure in Figure 3.1, as in Figure 3.2(b). As a result of this information and our observations after the experiments, the method we apply to obtain the $\{(x^t, \hat{y}^t)\}$ data is to apply the kNN algorithm to the dataset in the $\{(x^t, y^t)\}$ form, by taking the k value as a value between 3% and 5% of the number of data points in the preference dataset.

In this case, the applied learning method is designed as a standard clustering algorithm. Each value of the expected purchase probability function is a cluster, and the height of each cluster is the target value of that cluster. Due to the nature of our problem, the expected purchase probability function in an n-segment market structure will consist of $n+1$ clusters; The target value of the first cluster will be 1 and the target value of the $n+1^{\text{th}}$ cluster will be 0. These target values essentially represent the purchase probabilities p^i's associated with each cluster.

Figure 3.2 Estimates of the expected probability of purchase with kNN applied for different k values in the presence of deterministic product valuations



(a) $T = 1000, k = 25$

(b) $T = 1000, k = 100$

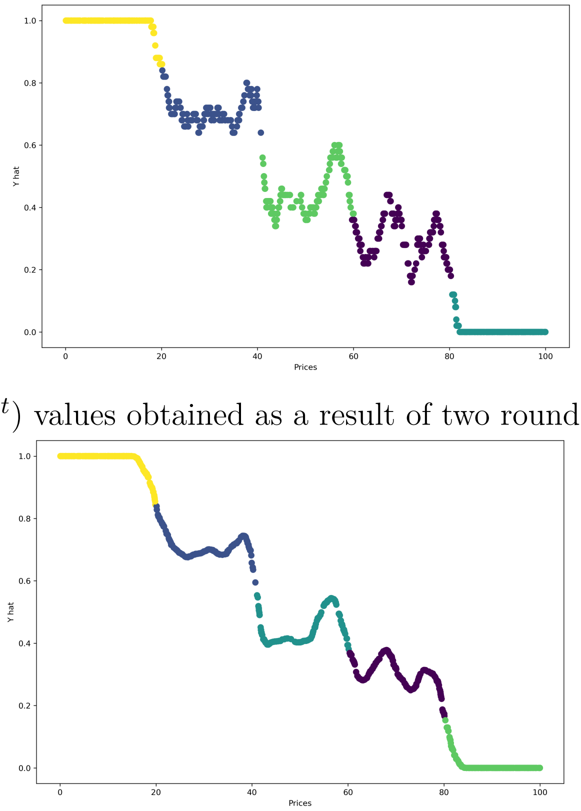
Our aim is to learn the starting and ending points of the sets on the x-axis. To achieve this, the proposed learning algorithm calculates the initial estimates of the product valuations by offline learning among randomly generated prices using a uniform distribution in the range of $[0, 100]$ during the T iteration, called the warm-up period.

3.1.1.1 On the converting the preference dataset

The initial step in our proposed algorithm involves converting the dataset $\{(x^t, y^t)\}$ to $\{(x^t, \hat{y}^t)\}$ by computing the average of the y values. This conversion process holds significant importance as the resulting converted dataset is utilized for clustering purposes. Therefore, it is crucial to ensure an appropriate conversion of the dataset to facilitate accurate parameter estimation. A key factor in this conversion is the hyperparameter k, which plays a vital role in shaping the expected purchase probability function. We have observed that the value of this hyperparameter has a substantial impact on the smoothness of the purchase probability function's shape, ultimately influencing the quality of the obtained clustering results. A higher value of k leads to a smoother shape, enabling improved clustering outcomes. Alternatively, achieving a smooth shape for the purchase probability function can also be accomplished by performing multiple conversions of the preference dataset. To do this, we introduce a new hyperparameter called $n_{converison}$ which determines the number of times the dataset is converted. When the value of this hyperparameter is greater than 1, the preference dataset is initially converted, followed by subsequent conversions of the previously converted dataset for $n_{conversion} - 1$ times. For example, if the value of this hyperparameter is set to 2, the $\{(x^t, y^t)\}$ dataset is first converted using KNN to obtain $\{(x^t, \hat{y}^t)\}$, and then KNN is applied once again on $\{(x^t, \hat{y}^t)\}$ to obtain $\{(x^t, \hat{\hat{y}}^t)\}$.

Figure 3.3 The effect of applying one or two rounds of kNN on expected purchase probability estimates in the presence of deterministic product valuations

(a) $(x^t, \hat{y}^t)$ values obtained as a result of one round of kNN



(b) $(x^t, \hat{\hat{y}}^t)$ values obtained as a result of two rounds of kNN

Figure 3.3 illustrates the effect of dataset conversion by applying KNN algorithm, executed once and twice. The dataset size remains constant, as does the value assigned to the hyperparameter k in both instances. By applying the conversion twice, a smoother shape of the expected purchase probability function is achieved. This enhanced smoothness offers potential advantages during the clustering phase of the algorithm, facilitating a more precise estimation of the problem's parameters.

Our proposed algorithm is as follows:

- Convert the dataset for $n_{conversion}$ times
  - For each price, take the average of y^t's for k nearest data points
  - If $n_{conversion} > 1$:
    Convert the converted dataset $n_{conversion} - 1$ times
- Cluster the data points
  - Using k-means clustering algorithm
  - Using only one feature vector ($\hat{\hat{y}}$ values)
  - Calculate the target values (p^i's)
- Estimate the parameters
  - Estimate the v_i's based on the clustered data points
- Find the optimal price

The first step of Algorithm 1 involves converting the preference data to a format suitable for clustering. In this algorithm, there is an option to perform multiple conversions on the dataset. When the hyperparameter $n_{conversion}$ is greater than 1, the preference data is first converted, and then the converted dataset is further converted for $n_{conversion} - 1$ iterations. In our experiments, we opt to perform the conversion process twice, as it yields a well-smooth purchase probability function that is conducive to clustering. Consequently, our converted dataset is of the form $\{(x^t, \hat{\hat{y}}^t)\}$.

In the next step, we need to cluster the data points in the converted dataset. A standard clustering algorithm to be applied to a dataset consisting of two-dimensional vectors considers the distances in both coordinates when calculating the distance between the elements. The outcome of such an approach is that the observed output points are clustered in such a way that they divide the price range between $[0, 100]$ by similar lengths. While the desired ideal clusters require the $\hat{\hat{y}}$ values of the points to be close to the cluster center, the scale of the price values depends on the market structure and should not affect the learning process much. For this reason, the clustering algorithm we apply uses only $\hat{\hat{y}}$ information, not two-coordinate $(x, \hat{\hat{y}})$ information for the distance calculation. To initiate the clustering process, we need to compute the target values (purchase probabilities) p^i. Assuming the probability of being in segment i is δ_i, we calculate the target values for each cluster using the following formula:

$$p^i = 1 - \sum_{j=1}^{i-1} \delta_j \tag{3.1}$$

Once the target values (purchase probabilities) have been calculated, we proceed to cluster the data points in the converted dataset based on their $\hat{\hat{y}}$ values. For each point $(x^t, \hat{\hat{y}}^t)$ in the converted dataset, we calculate the distance between the $\hat{\hat{y}}^t$ value and the target values p^i of each cluster. Subsequently, the point $(x^t, \hat{\hat{y}}^t)$ is assigned to the cluster whose target value is closest to the $\hat{\hat{y}}$ value:

$$\arg \min_{i=1,...,n+1} |\hat{\hat{y}}^t - p^i| \quad \forall t = 1, ..., T \tag{3.2}$$

Upon completion of the clustering process on the converted dataset, we obtain clusters that consist of prices x's and $\hat{\hat{y}}$ values. These clusters provide us with the necessary information to estimate the customers' valuations. To estimate v_i's, we take the average of the maximum and minimum prices in two consecutive clusters which basically gives us the middle point of these two clusters:

$$\hat{v}_i = \frac{\max\{x_i^t\} + \min\{x_{i+1}^t\}}{2} \tag{3.3}$$

In which, x_i^t's are the prices in cluster i, and x_{i+1}^t's are the prices in cluster $i+1$.

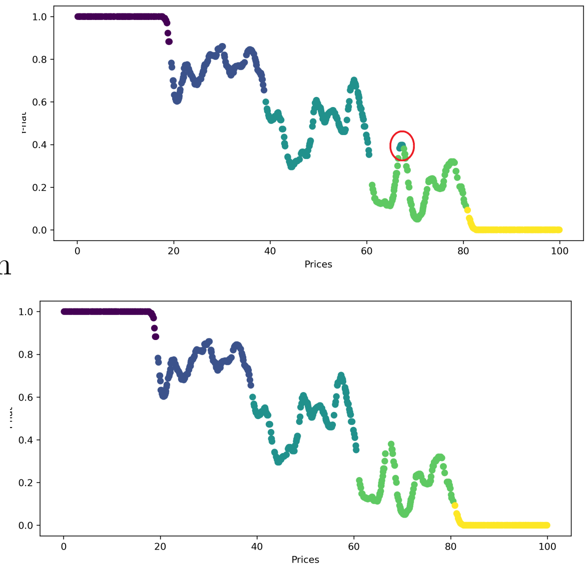
Finally, by estimating the v_i's and knowing the δ_i's we can calculate the optimal price by Eq 2.1.

3.1.2 Outlier elimination

During the clustering process of the converted data points, it is observed that some data points are inaccurately assigned to clusters that differ from their true cluster, i.e., the cluster to which they ideally belong. These data points can be regarded as outliers within the clusters identified by the clustering method. This situation can pose challenges when estimating the parameters, as the accuracy of parameter estimates relies on the data points within the identified clusters. To address this issue, it becomes necessary to identify the misclustered data points and devise an appropriate approach for handling them. The method employed to identify such data points is the Interquartile Range (IQR) method.

As can be seen in the region marked with red in Figure 3.4(a), some of the elements of the third set, which are expressed in light blue, are in the middle of the elements of the fourth set, which are expressed in light green. In the desired ideal clusters, the price (x) values of the points belonging to the same cluster are expected to be consecutive. Our solution to this problem is to remove the outliers on the x-axis of each cluster from the clusters after calculating them with the rule $Q_1 - 1.5 \times IQR$ and $Q_3 + 1.5 \times IQR$ (Tukey, 1977). The new clusters obtained after this process can be seen in Figure 3.4(b).

Figure 3.4 Identifying the wrongly clustered data points (outliers) with the IQR method and eliminating them



(a) Before outlier elimination

(b) After outlier elimination

In the proposed algorithms, when addressing outliers, we consistently employ the IQR method to identify these data points. Subsequently, we proceed with a straightforward approach of removing these outliers from the dataset.

3.1.3 Simulation

To evaluate the proposed algorithm, we will conduct a simulation study that encompasses various market structures, allowing us to analyze and discuss the results.

Table 3.2 presents the different settings we intend to test during the evaluation process. In this study, we focus on a single feature vector consisting of the converted preference values ($\hat{\hat{y}}$). In order to handle outliers within the dataset, we apply the IQR method. This method enables us to identify outliers and subsequently eliminate them from the dataset.

During the simulation study, we begin with a small dataset size of $T = 50$, and gradually increase the dataset size up to $T = 1000$. This approach allows us to analyze the impact of dataset size on the accuracy of the algorithm. Additionally, we aim to determine if we can obtain accurate estimates close to the true valuations within the warm-up period using smaller datasets. To ensure robustness and reliability of the results, each setting is replicated 200 times, providing a comprehensive evaluation of the algorithm's performance across different scenarios and dataset sizes.

Table 3.2 Setting of the simulation study for estimating the valuations in the presence of deterministic valuations and known segment distributions

Number of segments	n	4
Number of data points	T	[50,100,250,500,1000]
Number of nearest neighbors	k	[5,7,9,11,13,15,20,25,30,35,50]
Number of data conversion	$n_{conversion}$	[1,2]
Non-uniform valuations	v_i	[20, 50, 70, 80]

In this simulation study, we examine two distinct distributions: 1) an increasing distribution denoted as $\delta_i = [0.1, 0.2, 0.3, 0.4]$, and 2) a decreasing distribution denoted as $\delta_i = [0.4, 0.3, 0.2, 0.1]$. The outcomes of the simulation study are presented in Table 3.3, illustrating various settings and distributions. The average estimated values for customers' valuations are reported, along with their corresponding average absolute percent deviation (Error%). The error is calculated as follows:

$$\frac{1}{n}\sum_{i=1}^{n}(|\frac{\hat{v}_i - v_i}{v_i}|) \tag{3.4}$$

In which $\hat{v}_i$ denotes the estimated valuation of segment i and v_i denotes the real valuation of segment i.

Table 3.3 Simulation results for estimated valuations in the presence of deterministic valuations and known segment distributions

			δ_i = **[0.1, 0.2, 0.3, 0.4]**					δ_i = **[0.4, 0.3, 0.2, 0.1]**				
			Estimated Valuations				**Error%**	**Estimated Valuations**				**Error%**
			$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	
$T = 50$	$k = 5$	$n_{conversion} = 1$	38.75	56.47	68.33	80.79	27.52	30.61	41.85	55.37	72.40	24.94
		$n_{conversion} = 2$	33.79	56.13	68.84	78.22	21.28	26.43	44.72	58.43	69.56	18.07
	$k = 7$	$n_{conversion} = 1$	34.53	51.17	69.10	79.65	19.18	22.41	44.69	60.69	74.06	10.85
		$n_{conversion} = 2$	27.78	47.33	65.41	77.33	13.54	21.93	46.12	65.83	75.94	7.10
	$k = 9$	$n_{conversion} = 1$	31.58	46.11	67.35	79.72	17.45	22.09	41.75	67.23	75.39	9.17
		$n_{conversion} = 2$	32.16	48.94	66.45	79.48	17.16	23.28	44.60	65.52	79.22	8.64
	$k = 11$	$n_{conversion} = 1$	34.71	54.02	65.36	80.15	22.10	21.95	51.42	67.58	80.32	4.11
		$n_{conversion} = 2$	28.06	48.49	66.44	79.98	12.11	23.95	48.58	66.04	78.11	7.65
$T = 100$	$k = 5$	$n_{conversion} = 1$	38.68	55.02	67.57	79.05	27.02	30.07	43.17	57.66	69.98	23.54
		$n_{conversion} = 2$	38.38	52.82	64.85	77.06	27.15	26.45	44.76	59.45	72.83	16.69
	$k = 7$	$n_{conversion} = 1$	37.22	48.87	69.28	78.84	22.70	24.34	42.53	63.60	73.16	13.58
		$n_{conversion} = 2$	31.10	48.73	65.07	78.09	16.86	23.06	47.58	65.83	75.36	7.98
	$k = 9$	$n_{conversion} = 1$	34.81	44.69	70.16	78.93	21.56	23.68	42.77	68.70	75.18	10.19
		$n_{conversion} = 2$	31.69	47.10	65.63	78.84	17.99	22.76	46.70	64.07	75.92	8.49
	$k = 11$	$n_{conversion} = 1$	29.76	50.67	64.16	79.11	14.90	24.76	50.76	64.00	78.82	8.84
		$n_{conversion} = 2$	29.87	47.82	67.72	79.43	14.42	23.74	47.05	66.30	76.31	8.63
	$k = 13$	$n_{conversion} = 1$	27.56	49.28	67.61	80.41	10.80	20.57	50.86	66.06	76.88	3.52
		$n_{conversion} = 2$	23.82	48.88	65.55	81.32	7.33	22.89	49.95	70.01	79.06	3.94
	$k = 15$	$n_{conversion} = 1$	23.99	53.75	67.40	81.29	8.19	22.53	50.68	66.18	80.79	5.12
		$n_{conversion} = 2$	24.95	48.18	64.80	78.79	9.33	20.36	46.74	66.68	79.25	3.49
$T = 250$	$k = 5$	$n_{conversion} = 1$	39.48	56.98	69.49	79.15	28.29	31.43	41.62	57.15	71.25	25.81
		$n_{conversion} = 2$	36.10	52.48	67.23	78.87	22.70	25.63	44.12	61.08	74.12	15.00
	$k = 10$	$n_{conversion} = 1$	32.33	54.53	70.04	80.15	17.75	23.37	45.23	60.29	74.85	11.67
		$n_{conversion} = 2$	29.80	50.58	67.69	79.28	13.59	21.07	47.66	65.12	76.14	5.46
	$k = 15$	$n_{conversion} = 1$	23.55	53.72	68.46	80.49	7.00	23.09	48.94	64.97	79.40	6.39
		$n_{conversion} = 2$	27.80	50.46	69.47	79.76	10.24	20.54	50.47	67.67	78.35	2.26
	$k = 20$	$n_{conversion} = 1$	20.00	55.28	69.88	79.94	2.70	21.22	50.47	65.64	76.73	4.33
		$n_{conversion} = 2$	23.82	49.05	67.87	79.50	6.17	20.95	48.77	65.58	79.60	3.51
	$k = 25$	$n_{conversion} = 1$	26.47	52.26	68.31	80.50	9.98	21.27	49.21	66.81	77.48	3.91
		$n_{conversion} = 2$	25.43	51.15	69.43	80.54	7.74	18.90	50.17	68.60	80.53	2.12
	$k = 30$	$n_{conversion} = 1$	27.38	52.83	69.02	80.87	11.26	20.44	48.96	67.15	79.40	2.27
		$n_{conversion} = 2$	26.68	51.35	69.42	80.52	9.39	20.34	50.21	68.86	79.93	0.96
$T = 500$	$k = 10$	$n_{conversion} = 1$	32.60	55.88	71.37	80.05	19.20	23.24	45.25	62.78	76.78	10.01
		$n_{conversion} = 2$	28.50	52.21	68.99	79.73	12.17	21.52	47.25	64.84	76.39	6.25
	$k = 15$	$n_{conversion} = 1$	25.05	54.87	67.32	80.15	9.75	22.13	50.55	63.66	78.23	5.74
		$n_{conversion} = 2$	26.29	51.76	68.49	79.49	9.44	20.89	48.54	65.83	77.20	4.20
	$k = 20$	$n_{conversion} = 1$	22.58	54.12	71.77	80.31	6.01	20.50	50.81	66.15	76.48	3.51
		$n_{conversion} = 2$	24.85	51.94	70.11	79.46	7.24	20.45	50.34	69.45	77.66	1.66
	$k = 25$	$n_{conversion} = 1$	28.54	53.15	69.87	79.93	12.32	21.10	48.13	64.99	78.35	4.61
		$n_{conversion} = 2$	25.23	49.82	69.66	79.65	6.86	20.08	49.34	69.36	80.11	0.69
	$k = 30$	$n_{conversion} = 1$	23.11	54.11	69.95	79.58	6.09	20.32	50.52	67.07	78.97	2.03
		$n_{conversion} = 2$	21.67	49.69	69.69	79.05	2.64	20.15	49.43	69.85	78.86	0.88
	$k = 35$	$n_{conversion} = 1$	20.33	51.59	69.40	80.09	1.45	20.03	51.60	67.73	79.17	1.91
		$n_{conversion} = 2$	23.22	50.06	68.88	80.14	4.49	20.02	49.47	69.39	79.13	0.78
	$k = 50$	$n_{conversion} = 1$	22.58	50.86	70.16	80.27	3.80	20.44	50.53	68.09	80.31	1.59
		$n_{conversion} = 2$	21.92	50.20	69.51	80.01	2.68	19.97	50.14	69.00	79.24	0.70
$T = 1000$	$k = 10$	$n_{conversion} = 1$	33.55	57.38	68.42	79.03	21.49	24.77	45.36	61.63	76.26	12.44
		$n_{conversion} = 2$	29.12	53.31	69.67	79.86	13.22	20.65	48.50	65.01	76.73	4.36
	$k = 15$	$n_{conversion} = 1$	25.86	55.68	68.63	79.47	10.82	21.75	49.15	64.28	77.92	5.31
		$n_{conversion} = 2$	27.57	51.07	69.36	79.61	10.35	20.71	49.31	67.22	78.04	2.85
	$k = 20$	$n_{conversion} = 1$	21.77	54.30	70.89	79.85	4.73	21.02	51.73	64.79	75.21	5.50
		$n_{conversion} = 2$	23.76	52.39	71.31	79.81	6.43	20.03	48.58	67.29	78.06	2.32
	$k = 25$	$n_{conversion} = 1$	27.28	52.81	70.54	80.05	10.71	20.64	50.17	65.85	78.12	2.96
		$n_{conversion} = 2$	23.10	51.18	69.65	79.74	4.67	19.95	50.87	69.33	77.86	1.41
	$k = 30$	$n_{conversion} = 1$	24.58	52.42	69.73	80.02	7.03	19.89	50.17	66.44	77.91	2.14
		$n_{conversion} = 2$	21.75	50.95	70.15	79.88	2.76	20.03	49.45	68.31	78.92	1.26
	$k = 35$	$n_{conversion} = 1$	23.82	52.54	69.27	80.08	6.33	20.09	50.01	67.74	78.29	1.46
		$n_{conversion} = 2$	21.92	51.38	69.93	79.92	3.14	20.06	49.82	68.87	79.04	0.87
	$k = 50$	$n_{conversion} = 1$	22.55	51.72	70.26	79.93	4.17	20.19	50.02	67.74	79.18	1.30
		$n_{conversion} = 2$	21.99	50.11	69.94	80.07	2.59	20.01	50.14	69.94	78.98	0.43

One noteworthy observation is that as the number of data points (T) increases, we achieve more accurate estimates of valuations close to their true values with lower errors. This is attributed to the diminished fluctuations in the converted dataset when more data points are available, resulting in a more stable dataset that is less prone to wrongly cluster the data points which facilitates valuations estimation. Furthermore, it is evident that increasing the value of the hyperparameter k leads to more accurate estimation. However, it is essential to consider the value of k relative to the size of the dataset, as excessively large values may yield poorer estimates. Hence, the selection of the hyperparameter k should be proportional to the dataset size, T. Additionally, the impact of the hyperparameter $n_{conversion}$ on estimation performance can be observed. The results indicate that settings involving double conversion of the data exhibit lower errors compared to cases where the data is converted only once. This can be attributed to the increased smoothness of the purchase probability function resulting from the additional conversion step, which reduces fluctuations and, consequently, the likelihood of misassigning data points to incorrect clusters. Therefore, better estimation outcomes can be obtained by employing a two-step conversion process. In conclusion, it is evident that with a sufficiently large dataset, an appropriate value of k, and employing two conversions of the preference data, accurate estimates with minimal errors can be achieved.

3.1.4 Random distributions

In the previous simulation study, our investigation focused on two distributions with distinct patterns, one characterized by increasing values and the other exhibiting decreasing values, thereby possessing an orderly and structured nature. Nonetheless, it is imperative to acknowledge that real-world problems do not always adhere to such neat distributions. In numerous instances, the distribution of customers across segments assumes a random structure. For instance, the proportions of customers assigned to two segments may be relatively small and closely aligned, while the proportion for another segment could markedly differ and be large. Consequently, the distribution of customers does not exhibit a uniform structure similar to the increasing and decreasing distributions previously examined.

To accommodate this inherent variability in customer distributions, an additional distribution is incorporated into our study: the random distribution featuring a minimum probability assigned to each segment. Here, the minimum probability denotes a lower threshold representing the likelihood of customers belonging to specific segments. To execute our experiments, we initially generate a random distribution for the segments, incorporating a minimum probability of R. Subsequently, we generate a dataset based on this distribution. This approach serves to evaluate the robustness of the proposed Algorithm 1. Specifically, we aim to determine whether our algorithm can provide accurate estimates close to the true values of customers' valuations when customers are distributed randomly among the segments.

3.1.5 Simulation

Table 3.4 Setting of the simulation study for estimating the valuations in the presence of deterministic valuations and known random segment distributions

Number of segments	n	4
Number of data points	T	500
Number of nearest neighbors	k	[20,50]
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 50, 70, 80]
Minimum probability for random distribution	R	0.05

To evaluate the efficacy of the proposed Algorithm 1 in handling random distributions, a comprehensive simulation study encompassing diverse market structures will be conducted. The different settings selected for assessing the algorithm's performance are outlined in Table 3.4. It is noteworthy that only a single feature vector ($\hat{\hat{y}}$ values) is considered in this study. Building upon the observation that better estimates are achieved by converting preference data twice, we exclusively examine the scenario where the dataset undergoes conversion twice.

To ascertain the effectiveness of the proposed algorithm in the context of random distributions, we also undertake experiments involving two additional distributions for comparative purposes: 1) an increasing distribution, denoted as $\delta_i = [0.1, 0.2, 0.3, 0.4]$, and 2) a decreasing distribution, denoted as $\delta_i = [0.4, 0.3, 0.2, 0.1]$. Furthermore, to evaluate the impact of outlier elimination, we investigate two cases: one where the IQR method is applied to identify and eliminate outliers, and another where the IQR method is not utilized, allowing outliers to remain and potentially influence the estimation process. Each setting is subjected to 200 replications, ensuring a robust analysis of the algorithm's performance across various scenarios.

Table 3.5 Simulation results for estimated valuations in the presence of deterministic valuations and known random segment distributions

		With applying IQR					**Without applying IQR**				
		Estimated Valuations				**Error%**	**Estimated Valuations**				**Error%**
		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	
$k = 20$	δ_i = [0.1, 0.2, 0.3, 0.4]	24.44	51.11	69.79	79.85	8.66	28.07	49.41	68.97	79.73	13.83
	δ_i = [0.4, 0.3, 0.2, 0.1]	20.27	50.02	67.53	77.96	4.33	21.05	47.85	67.13	77.70	6.64
	δ_i = Random with $R = 0.05$	21.16	48.19	65.26	78.01	7.43	22.47	47.45	64.15	77.99	9.04
$k = 50$	δ_i = [0.1, 0.2, 0.3, 0.4]	21.08	50.63	69.79	79.92	5.60	22.05	50.19	70.10	79.92	6.26
	δ_i = [0.4, 0.3, 0.2, 0.1]	20.20	50.24	69.29	79.45	3.97	20.42	50.15	69.78	79.83	3.99
	δ_i = Random with $R = 0.05$	20.49	49.49	67.66	79.66	5.74	20.38	49.91	68.39	79.61	6.02

The outcomes of the simulation study are summarized in Table 3.5, showcasing the results obtained for different settings and distributions. The table presents the average estimated values for customers' valuations, accompanied by their corresponding average absolute percent deviation (Error%). The error is calculated using Equation 3.4.

The obtained results reveal a consistent trend across all hyperparameter k values and various distributions: applying the IQR method leads to more accurate estimates of valuations close to the true values, exhibiting lower error rates compared to scenarios where this data cleansing technique is not applied. Furthermore, when comparing $k = 20$ and $k = 50$, it becomes evident that employing a larger value of k ($k = 50$) yields improved valuation estimates with lower errors, irrespective of whether the IQR method is employed or not. Regarding the robustness of the proposed algorithm, the estimation errors for the random distribution are comparable to those observed in other distributions. In some cases, the errors are even lower, indicating that the algorithm demonstrates robustness in accurately estimating customers' valuations regardless of the underlying distribution. These findings emphasize the algorithm's ability to handle varying distribution types effectively and provide reliable estimations.

3.1.6 Smart learning

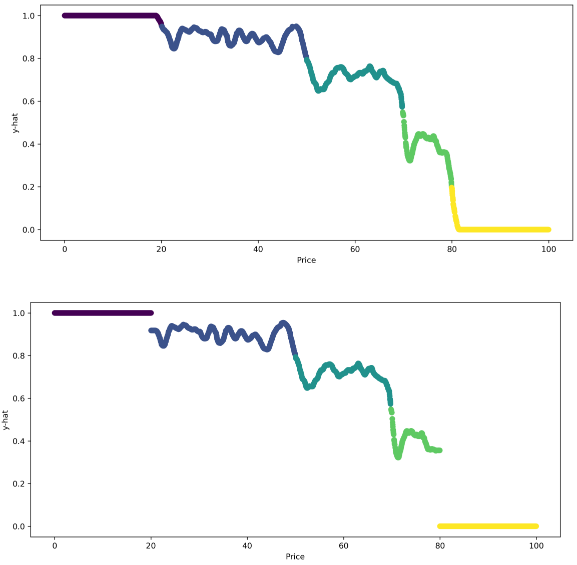
It appears that certain valuations can be directly learned from the original data without the need for data conversion and subsequent clustering. Specifically, the first (v_1) and last (v_n) valuations can be easily derived from the original data. When considering very low prices, customers consistently provide "buying" feedback as these prices are lower than the valuation of the first segment. Consequently, regardless of the segment they originate from, customers always make purchases at such low prices. Conversely, when it comes to very high prices, customers consistently provide "not buying" feedback since these prices surpass the valuations of the last segment, making the product undesirable for purchase. Leveraging these buying preferences, the first and last valuations can be directly inferred from the original dataset. To achieve this, the following approach is employed:

- Firstly, the dataset is sorted along the x-axis. Starting from the lowest price, the search is conducted for a "not buying" feedback (0). Once such a point is identified, it becomes a breaking point, and all preceding data points, each associated with a "buying" feedback (1), form the first cluster.

- Subsequently, the search begins from the highest price towards lower prices, aiming to locate a "buying" feedback (1). Upon finding this point, it too becomes a breaking point, and all subsequent data points, each accompanied by a "not buying" feedback (0), constitute the last cluster.
- For the data points falling between these two breaking points, the standard data conversion and clustering processes are performed to identify the remaining clusters, and subsequently, the valuations are estimated as previously outlined.

This approach not only enables the direct derivation of the first and last valuations from the original dataset but also reduces computational costs significantly. Figure 3.5 exemplifies the application of smart learning technique.

Figure 3.5 The effect of applying smart learning technique on expected purchase probability estimates in the presence of deterministic product valuations



(a) Before applying smart learning

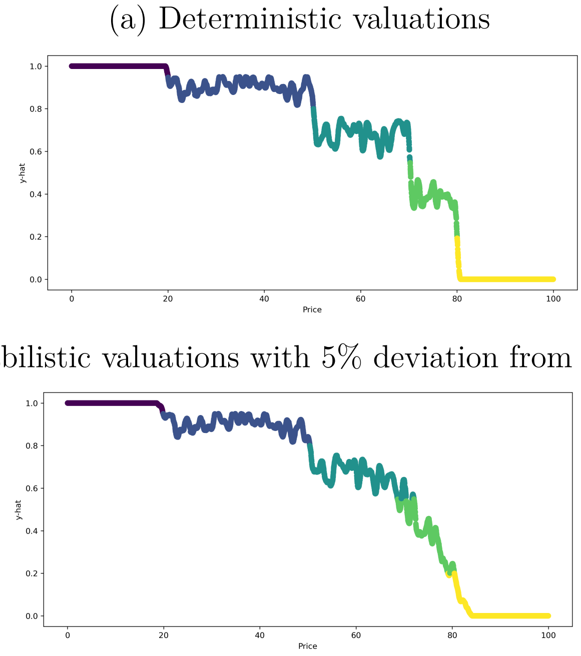
(b) After applying smart learning

3.1.7 Probabilistic valuations following the Uniform distribution

Up until now, our analysis has been based on deterministic valuations, assuming that all customers within each segment have identical valuations and are equally willing to pay for the product. However, this assumption may not accurately reflect real-world scenarios, as individuals within a segment may assign different values to a product, even when sharing similar attributes. To address this limitation, we introduce the concept of probabilistic valuations, which allows for varying valuations within a segment. Probabilistic valuations refer to the notion that the valuations of customers within a given segment i can fluctuate within a specified range $[l_i, u_i]$. Consequently, when a price is presented to a customer from segment i, their valuation will assume a value within this range, which may differ from the valuations of other customers within the same segment.

To incorporate probabilistic valuations into our analysis, we modify the generation of the synthetic dataset. After generating the continuous prices (x^t), we account for valuation deviations by introducing a percentage deviation ($p\%$) to the original valuations of each segment. For instance, if v_i represents the valuation of segment i and the valuations are subject to a percentage deviation of $p\%$, the valuation of a customer from this segment can vary within the range $[v_i \times (1 - p/100), v_i \times (1 + p/00)]$ according to an uniform distribution. Hence, when a customer arrives, their buying preference is determined by comparing the offered price with a valuation that is chosen from this range uniformly. It is important to note that the parameter governing the deviation of valuations remains constant across all segments; in other words, all valuations experience a uniform percentage change, such as 5%, 10%, and so forth.

Figure 3.6 The effect of uniform probabilistic valuations on expected purchase probability function


(a) Deterministic valuations

(b) Probabilistic valuations with 5% deviation from true values

Probabilistic valuations not only impact the valuations themselves but also influence the shape of the purchase probability function. This effect is demonstrated in Figure 3.6. In the presence of deterministic valuations, Figure 3.6(a) reveals a distinct decline in the $\hat{\hat{y}}$ values when transitioning from one cluster to another. This decline is characterized by sharp drops, indicating a clear distinction between the clusters. However, when probabilistic valuations are considered, as depicted in Figure 3.6(b), the sharp drops observed in the deterministic case transform into a more gradual decline. The inclusion of probabilistic valuations smooths out the transitions between clusters, resulting in a more continuous and nuanced change in the purchase probability function. Consequently, the boundaries between clusters become less pronounced, allowing for a smoother and more realistic representation of customer preferences and purchase behavior.

By incorporating the concept of probabilistic valuations, our objective is to evaluate the performance of the proposed Algorithm 1 in estimating valuations when such probabilistic valuations are present. To achieve this goal, an extensive simulation study will be conducted, aiming to observe and assess the efficiency of the algorithm.

3.1.8 Simulation

The simulation study will involve the generation of synthetic data sets where probabilistic valuations exist. These data sets will be designed to capture the variation in valuations within each segment, considering the range specified by the probabilistic valuation concept. By applying Algorithm 1 to these synthetic data sets, we will be able to analyze its performance in estimating the valuations accurately and efficiently. By conducting this comprehensive simulation study, we aim to gain insights into the strengths and limitations of Algorithm 1 when dealing with probabilistic valuations.

Table 3.6 Setting of the simulation study for estimating the valuations in the presence of uniform probabilistic valuations and known segment distributions

Number of segments	n	4
Number of data points	T	500
Number of nearest neighbors	k	20
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Minimum probability for random distribution	R	[0.05, 0.1, 0.15]
Valuation percentage deviation	p	[5%, 10%, 20%]

To assess the effectiveness of Algorithm 1 in handling probabilistic valuations, a comprehensive simulation study encompassing diverse market structures will be conducted. The different settings selected for assessing the algorithm's performance are outlined in Table 3.6. The simulation study focuses on the scenario where the dataset undergoes two conversions, as it has been observed that this approach leads to improved estimation results. We investigate five different probability distributions: an increasing distribution ($\delta_i = [0.1, 0.2, 0.3, 0.4]$), a decreasing distribution ($\delta_i = [0.4, 0.3, 0.2, 0.1]$), and three random distributions with varying minimum probability (R) values. This variation in distributions allows for an examination of the algorithm's performance across different levels of randomness. Additionally, we analyze the impact of employing a smart learning technique. Two cases are considered: one where the smart learning method is applied to identify the first and last valuations from the original dataset, and another where this method is not used, and valuations are estimated conventionally. We also investigate the impact of different values for the valuation percentage deviation $p\%$ on the algorithm's performance. This analysis aims to understand how the algorithm performs when valuations deviate within larger intervals. This investigation allows us to evaluate the algorithm's robustness in estimating valuations when faced with larger intervals of deviation. To handle outliers present in the dataset, the IQR method is employed. Each setting undergoes 200 replications, ensuring a robust evaluation of the algorithm's performance across diverse scenarios.

Table 3.7 Simulation results for estimated valuations in the presence of uniform probabilistic valuations and known segment distributions

		With applying smart learning					**Without applying smart learning**				
		Estimated Valuations				**Error%**	**Estimated Valuations**				**Error%**
		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	
$p = 0\%$	δ_i = [0.1, 0.2, 0.3, 0.4]	21.37	44.85	55.78	79.73	4.52	21.47	44.64	55.44	79.83	5.94
	δ_i = [0.4, 0.3, 0.2, 0.1]	20.25	44.63	55.04	78.36	3.71	20.09	44.71	55.04	78.04	4.65
	δ_i = Random with $R = 0.05$	20.42	43.61	57.27	78.33	6.78	20.22	44.22	56.53	77.22	7.63
	δ_i = Random with $R = 0.1$	20.47	43.98	56.38	78.85	5.48	20.36	43.78	56.45	79.06	6.06
	δ_i = Random with $R = 0.15$	20.59	44.04	56.03	79.17	5.28	20.26	44.23	55.87	79.68	5.41
$p = 5\%$	δ_i = [0.1, 0.2, 0.3, 0.4]	21.40	45.09	55.20	81.97	5.04	22.31	45.04	55.59	79.64	7.14
	δ_i = [0.4, 0.3, 0.2, 0.1]	20.11	44.31	55.20	79.12	3.76	20.20	44.54	54.77	76.72	4.70
	δ_i = Random with $R = 0.05$	20.11	43.99	56.95	79.45	5.82	20.35	44.34	57.95	78.15	6.98
	δ_i = Random with $R = 0.1$	20.32	44.26	56.61	80.21	5.62	20.09	43.82	56.51	79.53	5.83
	δ_i = Random with $R = 0.15$	20.49	43.70	56.30	80.70	4.83	20.24	44.39	56.38	79.11	5.73
$p = 10\%$	δ_i = [0.1, 0.2, 0.3, 0.4]	21.11	44.80	56.10	84.95	6.77	22.02	44.95	55.38	79.81	7.78
	δ_i = [0.4, 0.3, 0.2, 0.1]	19.49	44.56	55.13	81.78	4.49	20.20	44.63	55.67	75.94	5.44
	δ_i = Random with $R = 0.05$	19.65	43.41	57.07	82.28	6.79	20.29	43.42	57.35	77.28	7.20
	δ_i = Random with $R = 0.1$	19.69	43.53	56.46	82.91	6.20	20.24	43.80	56.21	78.74	6.45
	δ_i = Random with $R = 0.15$	19.70	43.83	55.86	83.32	5.48	20.29	43.81	56.34	79.13	6.09
$p = 20\%$	δ_i = [0.1, 0.2, 0.3, 0.4]	20.49	43.41	56.21	91.49	9.45	21.84	43.21	56.42	78.93	7.48
	δ_i = [0.4, 0.3, 0.2, 0.1]	18.22	43.54	56.72	86.65	7.66	20.11	43.90	56.13	75.73	6.69
	δ_i = Random with $R = 0.05$	17.98	43.01	57.16	88.24	9.41	20.05	43.49	57.60	77.68	7.99
	δ_i = Random with $R = 0.1$	18.37	41.82	56.59	88.48	8.80	20.15	42.62	57.09	78.13	7.07
	δ_i = Random with $R = 0.15$	18.45	42.90	56.71	89.42	8.53	20.16	43.21	56.57	78.74	6.80

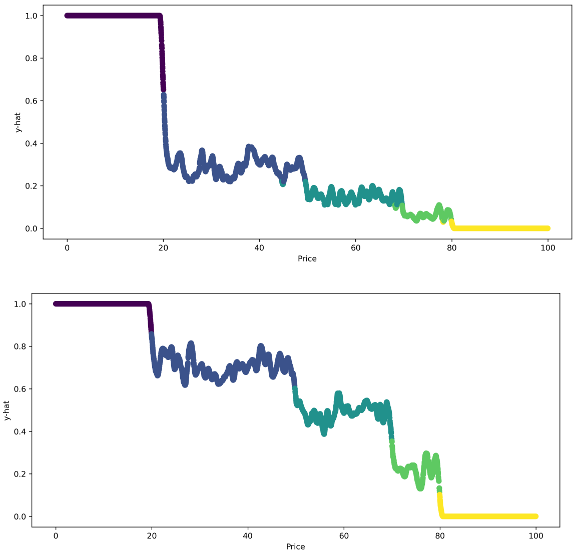
Table 3.7 provides a comprehensive summary of the simulation study's findings,

presenting the results obtained under various settings.

One crucial aspect to discuss is the impact of implementing smart learning technique. Across all examined scenarios, with the exception of the last one, it is evident that employing smart learning leads to improved outcomes; more accurate estimates of valuations, closely aligned with their true values, and lower error rates. The heightened accuracy primarily stems from the improved estimation of first and last valuations, as the smart learning approach directly utilizes the preference dataset for estimating these values. An additional observation indicates that as the range of valuation fluctuations widens, the disparity in error rates between utilizing smart learning and not implementing it diminishes. However, when valuations undergo significant changes within a considerably wide interval, such as in the case of $p = 20\%$, smart learning does not perform optimally, resulting in higher error rates compared to scenario where this method is not applied.

Furthermore, as the range of possible valuation variations expands (represented by higher $p\%$ values), the error in valuation estimation increases. This phenomenon is attributed to the fact that wider intervals of fluctuating valuations lead to a more monotonous shape of the purchase probability function, deviating from the characteristic step-like pattern. Consequently, the algorithm encounters difficulties in accurately identifying the boundaries between clusters, thereby resulting in poorer estimation outcomes. This change in the shape of purchase probability function can be seen in Figure 3.6.

Figure 3.7 The effect of minimum probability for random distribution (R) on expected purchase probability function



(a) $R = 0.05$

(b) $R = 0.2$

Additionally, when the minimum probability of belonging to a segment (R) increases, the error in estimation decreases. This phenomenon arises from the fact that higher values of R yield more distinct clusters, enabling the algorithm to more accurately identify and delineate these clusters. Consequently, the algorithm generates more accurate estimates, as illustrated in Figure 3.7.

3.1.9 Probabilistic valuations following the Weibull distribution

In the previous section, our investigation focused on probabilistic valuations, where the valuations of customers within each segment changed uniformly within a specified interval. In this section, we aim to explore an alternative scenario of probabilistic valuations, wherein the customer valuations within a segment follow the Weibull distribution instead of the uniform distribution. Our objective is to assess the performance of Algorithm 1 in the presence of this particular scenario.

To accomplish this, we utilize the Weibull distribution to model the valuations of the segments. The Weibull distribution is characterized by two parameters: α, which determines the shape of the distribution, and λ, which determines the scale. For our analysis, we set α to a fixed value of 2 ($\alpha = 2$), as it is commonly employed in pricing problems. To determine the appropriate value for λ, we derive it such that the expectation of the Weibull distribution ($E(X)$) equals the valuations of the segments, as indicated by the following equation:

$$v_i = \lambda_i \times gamma(1 + \frac{1}{\alpha}) \tag{3.5}$$

In which v_i is the true valuation of segment i. By solving the above equation, we can determine the λ value for each segment. Subsequently, we draw samples from the Weibull distribution using the obtained λ_i values. The methodology employed for generating the preference dataset involves generating random prices (x^t) within the range of $(0, 100)$. For each data point x^t, we assign it to a segment based on the segments' distribution δ_i, and subsequently, for that particular segment's valuation, we generate a random valuation based on a Weibull distribution with predefined α and λ_i values, obtained through the Eq.3.5.

3.1.10 Simulation

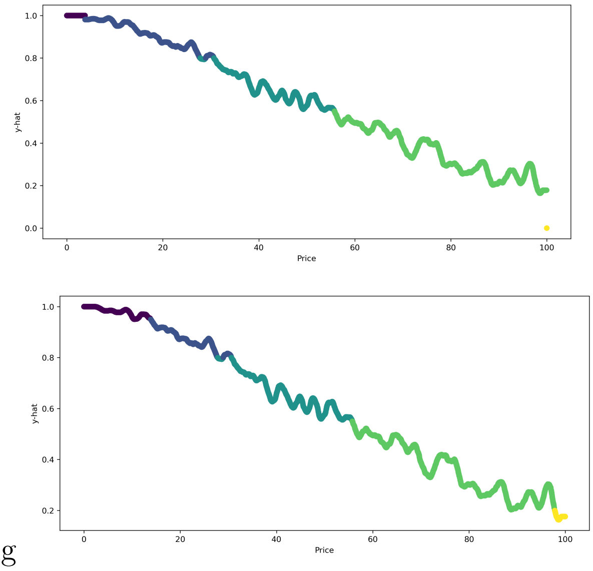
To evaluate the efficacy of Algorithm 1 in dealing with probabilistic valuations drawn from the Weibull distribution, an extensive simulation study will be conducted, encompassing various market structures. The simulation setting employed will be consistent with the setting described in section 3.1.8, with the only distinction being that customers' valuations will no longer follow a uniform distribution. Instead, they will be sampled from the Weibull distribution.

Table 3.8 Simulation results for estimated valuations in the presence of weibull probabilistic valuations and known segment distributions

	With applying smart learning					**Without applying smart learning**				
	Estimated Valuations				**Error%**	**Estimated Valuations**				**Error%**
	$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	
δ_i = [0.1, 0.2, 0.3, 0.4]	10.57	27.00	48.24	99.00	30.80	11.60	26.67	48.06	87.07	26.07
δ_i = [0.4, 0.3, 0.2, 0.1]	7.26	35.16	62.46	97.11	29.95	15.87	35.95	61.68	88.86	16.00
δ_i = Random with $R = 0.05$	7.80	36.98	60.58	97.55	29.95	16.15	37.30	59.79	86.70	16.03
δ_i = Random with $R = 0.1$	7.80	34.88	57.55	97.94	29.47	16.36	35.08	57.99	87.57	16.69
δ_i = Random with $R = 0.15$	7.67	34.21	56.58	98.30	29.08	15.72	33.63	56.66	86.84	15.87

The initial observation from the results indicates that smart learning is ineffective when valuations follow a Weibull distribution. Across all five different distributions considered, the use of smart learning for valuation estimation yields higher errors compared to not utilizing smart learning. This outcome can be attributed to challenges in estimating the first and last valuations. The wide range from which customers derive their valuations contributes to this issue. Consequently, in the smart learning approach, it is possible to encounter a 0-value preceding v_1 or a 1-value succeeding v_4, resulting in estimations that deviate significantly from their true values. This phenomenon is depicted in Figure 3.8. When examining the random distributions, it is observed that as the minimum probability R increases, the error decreases, which corroborates previous findings. Overall, the proposed algorithm fails to accurately estimate valuations, primarily due to the shape of the purchase probability function. In the presence of Weibull-distributed valuations, this function exhibits a monotonic and smooth form, making it challenging to distinguish between clusters. This difficulty is evident in Figure 3.8.

Figure 3.8 The effect of Weibull-distributed valuations on expected Purchase probability function



(a) With applying smart learning

(b) Without applying smart learning

3.2 Case where the number of segments (n) and segment distributions (δ) are known, but segment ordering is unknown

In this case, the segments in the market and their distribution are known, while the comparison between the product valuations of the segments (in other words, in which order the segments will be lost as the suggested price is increased) is not known. More precisely, while we have access to the δ_i values, we do not know which specific segment's distribution corresponds to each δ_i value.

Estimating customers' valuations in this scenario entails solving a series of problems. Considering the known segment distribution values and the unknown order, for the y-axis values of the expected purchase probability function, at most $n!$ different options are available. Since the number of segments in the markets we are interested in is not very high, the $n!$ value is at a reasonable level for individual counting.

Our work under this scenario is to apply the clustering process for all possible orders of the δ_i values at the end of the warming period consisting of T iterations and to choose the order with the lowest clustering error. The mentioned error is the square error calculated as the squares of the differences between $\hat{\hat{y}}$ and the target height (purchase probability) of the cluster (p^i) to which it is assigned for each $(x, \hat{\hat{y}})$ element. The calculation is as follows:

$$error_{clustering} = \sum_{i=1}^{n+1} \sum_{j=1}^{|cluster_i|} (\hat{\hat{y_{ij}}} - p^i)^2 \tag{3.6}$$

In which $|cluster_i|$ is the length of the cluster i, $\hat{\hat{y_{ij}}}$ is the $\hat{\hat{y}}$ value for the j^{th} element of cluster i, and p^i is the purchase probability of cluster i.

3.2.1 Simulation

A simulation study will be conducted to evaluate the clustering performance of the proposed Algorithm 1 in accurately identifying the underlying distribution order. As the market under investigation comprises four distinct segments ($n = 4$), there will be a total of 24 possible combinations of segment distribution orders. Consequently, for each replication of the simulation, the algorithm will be iterated 24 times to determine the true order of the provided distribution. The true valuations considered for this study will remain consistent with previous simulation studies, denoted as $v_i = [20, 45, 55, 80]$. Three different distribution scenarios will be examined: an increasing distribution represented by $\delta_i = [0.1, 0.2, 0.3, 0.4]$, a decreasing distribution characterized by $\delta_i = [0.4, 0.3, 0.2, 0.1]$, and a random distribution with a minimum probability threshold of $R = 0.15$. It is worth noting that our observations have indicated higher accuracy rates when employing smart learning techniques. Therefore, the simulation will be conducted exclusively with smart learning enabled. Various values for the dataset size (T) will be taken into account, and the value of k will be adjusted proportionally to the dataset's size. To ensure robustness and reliability, the experiments will be repeated 200 times, with each repetition involving different warm-up period lengths (T).

The experimental findings pertaining to this study are presented in Table 3.9. Each row of the table displays different values for T (dataset size) and their corresponding k values. For each combination, the probability of selecting the correct order is reported based on 200 replications. Additionally, the percentage deviation between the error amount of the selected order after the warm-up period and the error amount of the correct order is provided for each distribution scenario. Simulation runs were not conducted for certain settings in this study. Specifically, for the increasing and decreasing distribution scenarios, the last two settings ($T = 15000, T = 20000$) were excluded. This decision was made based on the observation that sufficiently high probabilities could already be achieved with smaller datasets, rendering the larger ones unnecessary for these distributions. Similarly, in the case of the random distribution, the settings $T = 6000$, $T = 7000$, and $T = 8000$ were omitted. This was due to the low probabilities obtained with these dataset sizes, indicating the need for larger datasets to obtain more reliable results in the random distribution scenario.

Table 3.9 Simulation results on the effect of the length of the warm-up period on finding the right sequence in the presence of deterministic valuations, known distribution, and unknown order

	δ_i = [0.1, 0.2, 0.3, 0.4]		δ_i = [0.4, 0.3, 0.2, 0.1]		δ_i = Random with $R = 0.15$	
	Percent deviation between error amounts	Probability of choosing the right sequence	Percent deviation between error amounts	Probability of choosing the right sequence	Percent deviation between error amounts	Probability of choosing the right sequence
$T = 1000, k = 20$	9.150%	0.520	10.356%	0.525	7.783%	0.345
$T = 2000, k = 20$	4.184%	0.620	4.930%	0.575	3.417%	0.415
$T = 5000, k = 20$	1.753%	0.735	1.331%	0.785	2.018%	0.391
$T = 6000, k = 25$	0.453%	0.890	0.746%	0.840	-	-
$T = 7000, k = 30$	0.323%	0.925	0.214%	0.955	-	-
$T = 8000, k = 35$	0.188%	0.965	0.198%	0.975	-	-
$T = 10000, k = 50$	0.002%	0.995	0.000%	1.000	0.768%	0.735
$T = 15000, k = 50$	-	-	-	-	0.557%	0.775
$T = 20000, k = 100$	-	-	-	-	0.303%	0.855

The results demonstrate a clear relationship between dataset size (T) and the probability of accurately identifying the true order. Specifically, as the dataset size increases, the probability of identifying the true order also increases. This phenomenon can be attributed to the diminishing fluctuations and increased stability in the shape of the purchase probability function, resulting in improved clustering accuracy. Consequently, the likelihood of successfully discovering the true order becomes more pronounced. In scenarios involving both increasing and decreasing distributions, employing a dataset consisting of 10,000 data points yields an almost perfect probability of identifying the true order. Therefore, in such cases, confidently determining the true order becomes a relatively straightforward task, substantiating the efficacy of our learning method.

Conversely, in instances characterized by random distributions, the likelihood of accurately identifying the true order diminishes significantly. Even with a dataset comprising 20,000 data points, the probability of correctly identifying the true order remains at a modest 85%. This limitation primarily arises from the close proximity of segment probabilities, which poses challenges for the algorithm in distinguishing between them.

3.3 Case where only the number of segments (n) is known

In this section, the focus shifts to a lower level of foreknowledge regarding the market structure. While the number of segments is known, the distribution of these segments becomes unknown. The objective is to estimate both the customers' valuations and the distribution of segments through offline learning using the preference dataset. Two scenarios are investigated under this limited knowledge availability. The first scenario assumes that the unknown parameters follow a uniform distribution, while the second scenario considers a non-uniform distribution.

3.3.1 Uniform Valuations and Uniform Segments' Distribution

In this scenario, we have less foreknowledge regarding the market structure. We assume only number of segments n is known, while the customers' valuations (v_i)'s and segments' distribution (δ_i)'s are unknown. We also assume that these parameters (v_i and δ_i) are uniformly distributed and the valuations are deterministic. These parameters are uniformly distributed within the range of $(0, 100)$ and $(0, 1)$ respectively. For example, if we consider the case of having four segments ($n = 4$), the valuations v_i's would be $[20, 40, 60, 80]$, and the distribution δ_i for each segment is uniformly set at 0.25, signifying that the probability of each segment is 25%.

3.3.1.1 Proposed Algorithm 2

In this scenario, we are to estimate the valuations and the segments' distribution through the provided dataset. We assume we are provided with a full dataset of size T. The proposed algorithm to achieve our goal is outlined as follows:

- Convert the dataset
    - For each price, take the average of y^t's for k nearest data points
- Cluster the data points
    - Using k-means clustering algorithm

  - Using different number of features

- Estimate the parameters
  - Estimate the v_i's and δ_i's based on the clustered data points

- Find the optimal price

By calculating the $\hat{y}$ values for all offered prices as explained in Algorithm 1, we will have our converted dataset as $\{(x^t, \hat{y}^t)\}$ which is the first step in our proposed algorithm.

In the next step, the converted dataset $\{(x^t, \hat{y}^t)\}$ is subjected to clustering into $n+1$ distinct clusters. To accomplish this, we employ the K-means clustering algorithm (Hartigan & Wong, 1979). The clustering process is based on two columns within our converted dataset, which serve as the features for clustering. We can consider either both columns as features or solely the $\hat{y}$ values. Therefore, we introduce an additional hyperparameter called $n_{features}$ which determines the number of features utilized in the clustering procedure. It takes values of either 1 or 2. When set to 1, the K-means clustering algorithm solely considers the $\hat{y}$ values for clustering, whereas a value of 2 incorporates both columns of the converted dataset for clustering. Prior to applying the K-means clustering algorithm, the values along both axes are normalized within the range of $0-1$.

Upon completion of the clustering process on the converted dataset, we obtain clusters that consist of prices x's and $\hat{y}$ values. These clusters provide us with the necessary information to estimate the parameters pertinent to our problem, namely the valuations and segments' distribution.

To estimate v_i's, we take the average of the maximum and minimum prices in two consecutive clusters which basically gives us the middle point of these two clusters:

$$\hat{v}_i = \frac{\max\{x_i^t\} + \min\{x_{i+1}^t\}}{2} \tag{3.7}$$

In which, x_i^t's are the prices in cluster i, and x_{i+1}^t's are the prices in cluster $i+1$.

To estimate the distribution of the segments, first, we need to calculate the average of $\hat{y}$ values ($\bar{\hat{y}}$) for each cluster as Eq.3.8 which gives us the purchase probability for customers in that cluster. Then, we can calculate the segments' distribution by subtracting the $\bar{\hat{y}}$ values for two consecutive clusters as Eq.3.9:

$$\bar{\hat{y}}_i = \frac{\sum_{t \in i} \hat{y}^t}{|cluster_i|} \tag{3.8}$$

In which, $|cluster_i|$ shows the number of data points in cluster i.

$$\hat{\delta}_i = \bar{\hat{y}}_i - \bar{\hat{y}}_{i+1} \tag{3.9}$$

$\bar{\hat{y}}_i$ represents the purchase probability for customers in cluster i. As we transition from one cluster to the next, a certain proportion of customers is lost due to an increase in their valuations. Consequently, the purchase probabilities exhibit a descending order. The lost portion during this transition signifies the proportion of customers within a segment. Therefore, by subtracting the purchase probability values we can estimate the probability distribution of customers in the segments (δ_i).

Finally, by estimating the v_i's and δ_i's we can calculate the optimal price by Eq.2.1.

3.3.1.2 Simulation

To test the performance of the proposed algorithm, we will generate datasets with different numbers of data points, and subsequently, we will execute simulations on these datasets employing diverse settings. The objective is to observe how our proposed algorithm effectively estimates the valuations and the segments' distribution. Table 3.10 provides an overview of the settings utilized in the simulation runs.

Table 3.10 Setting of the simulation study for estimating the valuations and segments' distribution in the presence of deterministic and uniform valuations and segments' distribution

Number of segments	n	4
Number of data points	T	[100,200,500,1000]
Number of nearest neighbors	k	[4, 5, 6, 10, 15, 20, 25]
Number of feature vectors	$n_{features}$	[1,2]

We consider a market structure with 4 segments. In order to examine the impact of dataset size (T) on estimation accuracy, we will utilize datasets with varying size values. Furthermore, we will experiment with different values of the hyperparameter k to analyze its effect on the estimation accuracy. Additionally, we will explore the effectiveness of clustering using one and two feature vectors to determine which approach yields better results.

Table 3.11 Simulation results for estimated valuations and distribution with applying 1 feature vector in the presence of deterministic and uniform valuations and segments' distribution

		$n = 4$, $n_{features} = 1$								
		Estimated Valuations				**Estimated distribution**				
		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$	**Error%**
$T = 100$	$k = 4$	31.93	47.48	57.08	69.55	0.25	0.25	0.25	0.25	12.03
	$k = 5$	30.13	42.80	58.25	68.73	0.26	0.22	0.25	0.25	11.30
	$k = 6$	28.95	45.80	59.98	74.35	0.23	0.24	0.25	0.26	10.27
	$k = 10$	27.03	44.85	59.43	76.80	0.26	0.24	0.24	0.23	8.96
	$k = 15$	25.63	44.95	60.73	76.70	0.27	0.22	0.23	0.22	10.37
	$k = 20$	20.85	37.48	59.73	76.70	0.22	0.24	0.24	0.22	6.04
	$k = 25$	23.13	41.05	58.68	75.68	0.19	0.22	0.24	0.21	10.32
$T = 200$	$k = 4$	35.38	45.68	54.65	67.58	0.25	0.25	0.25	0.25	14.44
	$k = 5$	30.00	44.78	59.23	70.68	0.23	0.27	0.27	0.22	14.04
	$k = 6$	29.70	42.48	57.23	72.00	0.24	0.25	0.26	0.24	10.32
	$k = 10$	32.90	44.93	58.58	73.43	0.26	0.24	0.22	0.25	13.37
	$k = 15$	23.60	43.30	56.28	76.35	0.25	0.25	0.24	0.23	6.20
	$k = 20$	21.55	41.23	60.30	79.38	0.24	0.23	0.26	0.24	4.16
	$k = 25$	21.80	39.20	57.85	78.43	0.22	0.24	0.26	0.24	4.80
$T = 500$	$k = 4$	36.90	48.13	55.65	63.83	0.25	0.25	0.25	0.25	16.54
	$k = 5$	35.60	43.50	56.50	68.33	0.23	0.25	0.28	0.24	16.54
	$k = 6$	34.43	41.88	56.90	69.55	0.22	0.26	0.27	0.23	15.76
	$k = 10$	28.28	41.40	54.93	70.25	0.24	0.22	0.25	0.26	11.16
	$k = 15$	26.13	42.60	61.65	72.63	0.25	0.25	0.23	0.25	7.44
	$k = 20$	24.63	42.03	59.48	75.95	0.24	0.23	0.25	0.25	5.81
	$k = 25$	23.65	39.70	57.28	77.35	0.23	0.23	0.25	0.26	5.85
$T = 1000$	$k = 4$	37.75	46.90	55.95	66.05	0.25	0.25	0.25	0.25	16.27
	$k = 5$	32.03	44.03	57.18	66.85	0.23	0.24	0.26	0.26	13.83
	$k = 6$	32.55	43.73	56.00	70.90	0.24	0.23	0.27	0.24	13.77
	$k = 10$	31.35	43.13	59.68	69.13	0.26	0.23	0.22	0.25	12.85
	$k = 15$	27.68	41.75	58.00	73.98	0.25	0.22	0.24	0.25	8.48
	$k = 20$	27.33	43.48	60.80	74.55	0.26	0.25	0.23	0.24	8.79
	$k = 25$	24.48	42.95	61.68	76.43	0.25	0.25	0.24	0.24	5.76

Table 3.12 Simulation results for estimated valuations and distribution with applying 2 feature vectors in the presence of deterministic and uniform valuations and segments' distribution

		$n=4,\ n_{features}=2$								
		Estimated Valuations				**Estimated distribution**				
		$\hat{v}_1$	$\hat{v}_2$	$\hat{v}_3$	$\hat{v}_4$	$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$	**Error%**
$T=100$	$k=4$	27.13	44.05	60.23	75.05	0.31	0.22	0.09	0.38	26.02
	$k=5$	24.43	43.28	60.58	77.78	0.28	0.17	0.27	0.27	11.65
	$k=6$	22.75	40.65	58.68	77.20	0.28	0.24	0.15	0.31	12.95
	$k=10$	23.83	40.58	59.75	77.60	0.27	0.23	0.25	0.23	6.31
	$k=15$	20.55	38.88	60.70	79.25	0.25	0.23	0.22	0.26	4.14
	$k=20$	21.95	40.28	60.15	79.85	0.22	0.27	0.21	0.21	7.81
	$k=25$	22.50	42.53	61.63	78.83	0.20	0.21	0.28	0.19	11.43
$T=200$	$k=4$	26.38	41.10	58.33	73.28	0.32	0.16	0.20	0.32	19.50
	$k=5$	26.13	41.95	59.63	74.83	0.25	0.24	0.23	0.28	8.48
	$k=6$	23.65	42.18	56.00	75.20	0.25	0.21	0.26	0.28	8.49
	$k=10$	22.35	42.40	59.50	78.00	0.27	0.26	0.21	0.25	6.30
	$k=15$	21.65	41.30	60.08	79.30	0.23	0.28	0.23	0.24	5.10
	$k=20$	20.85	40.40	60.45	80.23	0.23	0.27	0.25	0.23	4.22
	$k=25$	20.28	39.85	59.75	79.73	0.21	0.29	0.23	0.23	5.99
$T=500$	$k=4$	27.65	42.08	56.78	72.93	0.25	0.26	0.21	0.28	10.80
	$k=5$	25.70	42.08	55.58	74.20	0.26	0.24	0.20	0.29	11.67
	$k=6$	24.28	41.68	58.98	75.18	0.24	0.24	0.30	0.22	9.50
	$k=10$	22.93	41.90	60.75	78.68	0.23	0.27	0.28	0.21	8.14
	$k=15$	21.18	41.55	61.00	80.33	0.23	0.27	0.26	0.23	4.62
	$k=20$	21.18	40.95	59.28	79.73	0.24	0.23	0.27	0.24	4.03
	$k=25$	20.30	40.10	60.08	79.78	0.23	0.25	0.27	0.24	3.10
$T=1000$	$k=4$	28.65	46.15	60.45	73.73	0.28	0.28	0.12	0.32	21.81
	$k=5$	27.28	42.93	58.08	73.83	0.26	0.29	0.16	0.29	15.28
	$k=6$	27.40	44.03	60.55	75.73	0.28	0.25	0.21	0.26	10.40
	$k=10$	24.28	43.55	60.65	78.20	0.25	0.25	0.25	0.24	5.22
	$k=15$	21.98	41.75	59.55	79.15	0.24	0.25	0.26	0.24	3.34
	$k=20$	20.35	40.18	58.95	79.80	0.25	0.24	0.25	0.25	0.95
	$k=25$	21.35	41.58	59.93	79.88	0.24	0.25	0.24	0.25	2.22

Table 3.13 True values of valuations and segments' distribution in the presence of deterministic and uniform valuations and distribution in a market structure with 4 segments

v_1	v_2	v_3	v_4	δ_1	δ_2	δ_3	δ_4
20	40	60	80	0.25	0.25	0.25	0.25

Table 3.11 presents the simulation results corresponding to the scenario where only one column ($\hat{y}$) from the converted dataset is considered for clustering. In Table 3.12, on the other hand, the simulation results are displayed for the scenario where both columns of the converted dataset are taken into consideration. The average estimated values of v_i and δ_i, along with the corresponding error values, are provided. These results are based on 200 replications. The true values of valuations and the distribution of segments can be found in Table 3.13. The error is computed using the following formula:

$$\frac{1}{2n}\sum_{i=1}^{n}(|\frac{\hat{v}_i - v_i}{v_i}| + |\frac{\hat{\delta}_i - \delta_i}{\delta_i}|) \tag{3.10}$$

In which $\hat{v}_i$ is the estimated valuation of segment i, v_i is the real valuation of segment i, $\hat{\delta}_i$ is the estimated probability of being in segment i, and δ_i is the real probability of being in segment i.

In several instances, the proposed algorithm demonstrates success in estimating the v_i and δ_i values, closely approximating their true values with minimal error. It is important to note that the error is significantly influenced by the size of the dataset and the chosen value of k. These two values should exhibit a proportional relationship to ensure accurate parameter estimation by the algorithm. While an increase in the dataset's size does not necessarily guarantee low error in parameter estimation, the selection of an appropriate k value becomes crucial. The findings indicate that, generally, considering both feature vectors for clustering leads to more precise parameter estimates, closely aligning with their true values. Among the various experimental settings, the best result is achieved when employing 1000 data points, $k = 20$, and both feature vectors, resulting in an error of less than 1%. This outcome reinforces the effectiveness of the proposed algorithm in accurately estimating the parameters of the problem when they are uniformly distributed.

3.3.2 Non-uniform Valuations and Non-uniform Segments' Distribution

Assuming uniformity for valuations and distribution of segments may not accurately reflect real-world scenarios. Due to market dynamics, it is more likely that customers from different segments are distributed non-uniformly. For example, the proportion of customers willing to pay a low price for the offered product differs from those willing to pay a high price. Similarly, the valuations of customers may also exhibit non-uniform distribution patterns. With these considerations in mind, we aim to address this limitation by introducing a more realistic problem setting in this section. Specifically, we incorporate customers' valuations and segments' distribution that are non-uniformly distributed. We proceed by applying the proposed algorithm discussed in Section 3.3.1.1 to this problem, assessing the algorithm's effectiveness in accurately estimating the problem's parameters.

3.3.2.1 Simulation

In order to incorporate non-uniform valuations, we assume the following values: $[20, 50, 55, 60]$. Additionally, for non-uniform distribution, we consider two distributions: $[0.1, 0.2, 0.3, 0.4]$ and $[0.4, 0.3, 0.2, 0.1]$. By utilizing these values, we can create various market structures where one or two of these parameters exhibit non-uniformity. This allows us to analyze and compare the performance of the proposed algorithm in estimating the parameters under different market structures.

Table 3.14 Setting of the simulation study for estimating the valuations and segment distributions in the presence of deterministic and non-uniform valuations and segment distributions

Number of segments	n	4
Number of data points	T	1000
Number of nearest neighbors	k	20
Number of data conversion	$n_{conversion}$	2
Number of feature vectors	$n_{features}$	2

Table 3.15 Simulation results for estimated valuations and distribution with applying 2 feature vectors in the presence of deterministic and non-uniform valuations and segment distributions

	Customers' Valuations				**Segments' Distribution**				**Average Absolute Percent Deviation**
Market 1 (Real)	**20**	**40**	**60**	**80**	**0.4**	**0.3**	**0.2**	**0.1**	
Market 1 (Estimate)	20.068	39.904	59.413	79.489	0.394	0.293	0.205	0.100	1.093
Market 2 (Real)	**20**	**40**	**60**	**80**	**0.1**	**0.2**	**0.3**	**0.4**	
Market 2 (Estimate)	20.549	40.575	60.261	79.910	0.101	0.205	0.297	0.388	1.585
Market 3 (Real)	**20**	**50**	**55**	**60**	**0.25**	**0.25**	**0.25**	**0.25**	
Market 3 (Estimate)	20.680	48.185	58.157	78.891	0.243	0.316	0.418	0.017	29.241
Market 4 (Real)	**20**	**50**	**55**	**60**	**0.1**	**0.2**	**0.3**	**0.4**	
Market 4 (Estimate)	24.223	52.001	59.552	79.743	0.103	0.397	0.479	0.010	40.618
Market 5 (Real)	**20**	**50**	**55**	**60**	**0.4**	**0.3**	**0.2**	**0.1**	
Market 5 (Estimate)	20.329	44.825	55.582	77.633	0.376	0.247	0.350	0.019	27.689

In Table 3.15, the customers' valuations and segments' distribution of five different market structures along with the average of the estimates of these values, and the mean absolute percent deviation of the actual values after 200 replications of simulation are reported. The values of hyperparameters T, k and $n_{features}$ are specified in Table 3.14. To ensure a smoother purchase probability function, we employ a two-times data conversion approach in this simulation study. Furthermore, for the identification and handling of outliers, we utilize the IQR method.

In accordance with these results, the learning method gives successful results regardless of the segments' distribution in cases where customers' valuations have equal intervals (Market 1 and Market 2). In these cases, the proposed algorithm accurately estimates both the valuations and the distribution with low error. However, for market structures characterized by uneven distribution on the x-axis (Market 3, Market 4, and Market 5), the proposed algorithm fails to accurately estimate both sets of parameters. The reason is attributed to the K-means algorithm, which works effectively when dealing with evenly distributed data points. In the cases of market structures 4 and 5, where data points are unevenly distributed on both axes, and in market structure 3, where only data points are unevenly distributed on the x-axis, the K-means method proves ineffective. These findings highlight the necessity of devising a solution for cases where data points do not adhere to uniform distribution patterns.

As observed in the results of this section the proposed Algorithm 2 was effective in estimating valuations and distributions when they followed a uniform distribution. However, it encountered challenges when dealing with non-uniform distributions. Therefore, in the next section, the research focus shifts towards addressing the scenario where diverse distributions exist on both the x-axis (valuations) and the y-axis (probability of purchase). Building upon the available foreknowledge about the market structure, the research aims to develop learning methods that improve the accuracy of parameter estimation for the problem at hand. These algorithms are designed to handle the complexity arising from diverse distributions and enhance the estimation process.

3.3.3 Estimating segments' distribution (δ_i) through discrete prices

In order to estimate the valuations and distribution of segments when only the number of segments is known, we adopt an approach where the price values in the preference dataset are no longer continuous but discrete. In the offline phase, where we are provided with a complete dataset $\{(x, y)\}$, it comprises m distinct discrete price values (x) which are in the range of $(0, 100)$, with each price being offered to customers m' times along with the binary buying preferences (y). The discrete preference dataset provided by the seller is structured as Table 3.16. Based on this dataset, we will introduce an algorithm to estimate the parameters of interest.

Table 3.16 Discrete preference dataset

	Offered Prices	Buying Preferences
	x^1	1
	.	1
m' times {	.	0
	.	0
	x^1	1
	x^2	1
	.	0
m' times {	.	0
	.	0
	x^2	1
	.	.
	.	.
	.	.
	x^m	1
	.	0
m' times {	.	0
	.	1
	x^m	1

3.3.3.1 Deterministic valuations

In this section, our focus is on estimating the distribution of segments (δ_i) when confronted with deterministic valuations. We assume that the number of segments n is already known.

The learning problem we propose in this scenario revolves around the expected probability of purchase function. Similar to previous algorithms, we cannot directly utilize the preference dataset to learn the parameters. Instead, we need to transform this dataset with a 0-1 structure into expected function data. Considering the discrete nature of the suggested price options, we can calculate the expected value as the average of the y values associated with the same price x. Consequently, for each price x, we compute the average of the corresponding y values and assign the resulting value to price x as its expected purchase probability ($\hat{y}$). By doing so, our preference dataset $\{(x,y)\}$ is transformed into $\{(x,\hat{y})\}$. The Figure below illustrates the form of the expected purchase probability function when discrete prices are offered.

Figure 3.9 Expected purchase probability as a function of discrete suggested prices in the presence of deterministic valuations

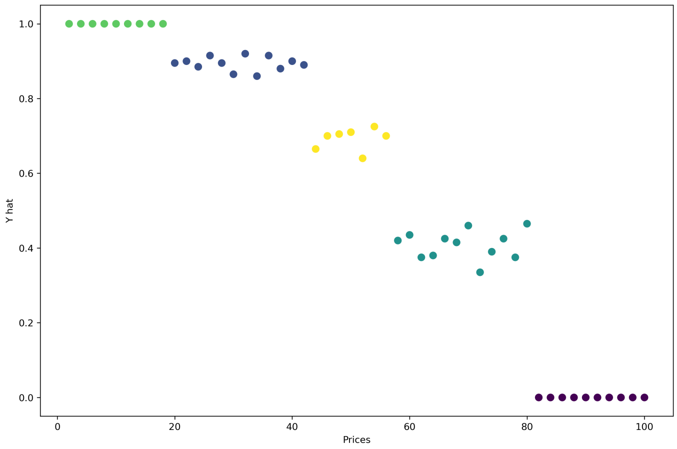


Figure 3.9 depicts the graphical representation of the expected purchase probability function in scenarios where offered prices are discrete and valuations are deterministic. In contrast to the continuous price scenario, wherein a smooth and uninterrupted trend was observed in the shape of this function, the discrete price scenario exhibits a distinct and discrete pattern characterized by sharp declines when transitioning between clusters. After converting the discrete preference dataset, we can employ this converted dataset for clustering and estimating the distribution of segments.

3.3.3.2 Proposed algorithm for estimating the segments' distribution (Distribution Estimation Algorithm)

The following outlines the proposed algorithm for estimating the distribution of segments when the number of segments (n) is known, using the provided discrete preference dataset:

- Convert the dataset
  - For each price x in the preference dataset, take the average of y values associated with the same price x and assign the resulting value as its expected purchase probability ($\hat{y}$)
- Cluster the converted dataset

  - Using k-means clustering algorithm

- Estimate the distribution of segments
  - Estimate the δ_i's based on the clustered data points

The initial step in our proposed algorithm involves converting the preference dataset, which has $m * m'$ data points. As previously stated, our learning problem revolves around the expected purchase probability function. To compute these expected values and convert our dataset, for each price value x in the preference dataset, we gather all m' associated y values corresponding to the same price x. Then we calculate the average of these y values and assign the resulting average value as the expected purchase probability ($\hat{y}$) for the price x. By implementing these steps, the dataset will be transformed into a converted format, denoted as $\{(x, \hat{y})\}$, consisting of m data points. The resulting converted dataset exhibits a structure resembling that shown in Table 3.17:

Table 3.17 Converted discrete preference dataset

Offered Prices	Expected Purchase Probability
x^1	$\hat{y}^1$
x^2	$\hat{y}^2$
x^3	$\hat{y}^3$
.	.
.	.
.	.
x^m	$\hat{y}^m$

Subsequently, in the following step, the converted dataset $\{(x, \hat{y})\}$ undergoes clustering to partition it into $n+1$ distinct clusters. This clustering process utilizes the K-means algorithm. The algorithm relies on two columns from our converted dataset, which act as the features for clustering. Similar to the proposed Algorithm 2, we can consider either both columns as features or solely the $\hat{y}$ values. Prior to applying the K-means clustering algorithm, the values along both axes are normalized to fall within the range of $0-1$. This normalization step ensures that the features are on a consistent scale and helps facilitate accurate clustering.

After the completion of the clustering process on the converted dataset, we obtain clusters comprising of prices (x) and corresponding expected purchase probabilities ($\hat{y}$) values. These clusters provide us with the necessary information to estimate the segments' distribution.

To estimate the distribution of the segments, first, we need to calculate the average of $\hat{y}$ values ($\bar{\hat{y}}$) for each cluster as Eq.3.11 which gives us the purchase probability for customers in that cluster. Then, we can calculate the segments' distribution by subtracting the $\bar{\hat{y}}$ values for two consecutive clusters as Eq.3.12:

$$\bar{\hat{y}}_i = \frac{\sum_{t \in i} \hat{y}^t}{|cluster_i|} \tag{3.11}$$

In which, $|cluster_i|$ shows the number of data points in cluster i.

$$\hat{\delta}_i = \bar{\hat{y}}_i - \bar{\hat{y}}_{i+1} \tag{3.12}$$

$\bar{\hat{y}}_i$ represents the purchase probability for customers in cluster i. As we transition from one cluster to the next, a certain proportion of customers is lost due to an increase in their valuations. Consequently, the purchase probabilities exhibit a descending order. The lost portion during this transition signifies the proportion of customers within a segment. Therefore, by subtracting the purchase probability values we can estimate the probability distribution of customers in the segments ($\hat{\delta}_i$).

3.3.3.3 Simulation

To evaluate the performance of the Distribution Estimation Algorithm in estimating the distribution of segments in the presence of deterministic valuations, two simulation studies were conducted. These studies involved considering various values for m, m', and distributions. The objective was to assess the algorithm's effectiveness under different conditions.

In the first simulation study, both columns in the converted dataset were used for clustering. The valuations considered for this study were $v_i = [18, 43, 56, 81]$. The purpose of this study was to analyze the algorithm's performance when utilizing both features for clustering. In the second simulation study, only the $\hat{y}$ values column in the converted dataset was employed for clustering. The valuations considered for this study were $v_i = [8, 46, 56, 81]$. The rationale behind changing the valuations in this study was to disrupt the uniformity or near-uniformity of the data points. This change aimed to assess the impact of using one versus two feature vectors in the clustering process.

By conducting these two simulation studies, we aimed to gain insights into the performance of the algorithm under different scenarios and determine the effect of utilizing different feature vectors on the clustering results and subsequent distribution estimation.

The outcomes of these simulation studies, comprising 200 replications, are presented in Tables 3.18 and 3.19.

In these tables, the median of the estimated values for the segments' distribution is reported, accompanied by their corresponding average absolute percent deviation error (Error%). The calculation of the error is performed using the following formula:

$$\frac{1}{n}\sum_{i=1}^{n}(|\frac{\hat{\delta}_i - \delta_i}{\delta_i}|) \tag{3.13}$$

In which $\hat{\delta}_i$ denotes the estimated distribution of segment i and δ_i denotes the real distribution of segment i.

This error metric quantifies the average percentage deviation between the estimated values and the true values of the segments' distribution. It provides a measure of the accuracy of the algorithm's estimations, allowing for a comparison of performance across different scenarios considered in the simulation study.

Table 3.18 Estimated segments' distribution by Distribution Estimation Algorithm using two feature vectors in the presence of deterministic valuations

		True Distribution				Error %	True Distribution				Error %	True Distribution				Error %
		$\delta_1 = 0.1$	$\delta_2 = 0.2$	$\delta_3 = 0.3$	$\delta_4 = 0.4$		$\delta_1 = 0.4$	$\delta_2 = 0.3$	$\delta_3 = 0.2$	$\delta_4 = 0.1$		Random with $R = 0.1$				
		Estimated Distribution					Estimated Distribution					Estimated Distribution				
		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$	
$m = 20$	$m' = 50$	0.100	0.200	0.300	0.400	13.208	0.392	0.312	0.189	0.100	12.278	0.389	0.208	0.165	0.155	15.581
	$m' = 100$	0.098	0.200	0.299	0.400	9.653	0.400	0.298	0.203	0.100	8.771	0.411	0.224	0.145	0.142	10.602
	$m' = 150$	0.097	0.197	0.299	0.399	7.657	0.399	0.301	0.200	0.101	7.079	0.357	0.203	0.150	0.170	8.675
	$m' = 200$	0.099	0.200	0.301	0.400	6.389	0.401	0.302	0.199	0.101	5.802	0.395	0.220	0.146	0.140	6.884
	$m' = 250$	0.098	0.200	0.300	0.399	5.450	0.401	0.298	0.202	0.100	5.283	0.406	0.216	0.150	0.151	6.456
	$m' = 500$	0.100	0.199	0.301	0.400	4.082	0.400	0.302	0.199	0.100	3.653	0.385	0.194	0.148	0.145	4.733
	$m' = 1000$	0.100	0.199	0.298	0.401	2.820	0.400	0.300	0.200	0.100	2.751	0.385	0.194	0.145	0.150	3.370
$m = 50$	$m' = 20$	0.099	0.218	0.290	0.392	13.513	0.387	0.290	0.214	0.100	13.016	0.343	0.268	0.168	0.154	17.060
	$m' = 40$	0.101	0.200	0.299	0.400	8.456	0.397	0.299	0.199	0.100	8.891	0.395	0.211	0.162	0.141	11.004
	$m' = 60$	0.100	0.199	0.302	0.399	6.969	0.396	0.303	0.196	0.100	7.701	0.379	0.215	0.145	0.145	7.955
	$m' = 80$	0.101	0.201	0.298	0.398	6.281	0.399	0.301	0.202	0.099	6.087	0.409	0.212	0.142	0.145	7.016
	$m' = 100$	0.098	0.200	0.302	0.398	5.552	0.399	0.303	0.199	0.098	5.568	0.400	0.204	0.153	0.146	6.076
	$m' = 200$	0.100	0.199	0.299	0.399	3.629	0.400	0.299	0.201	0.100	3.949	0.388	0.222	0.157	0.150	4.349
	$m' = 400$	0.100	0.202	0.300	0.400	2.964	0.399	0.299	0.201	0.100	2.743	0.399	0.222	0.151	0.143	3.383

The results in Table 3.18 indicate that the proposed algorithm demonstrates success in estimating the distribution with low error, particularly when the dataset size is not very small. As the dataset size increases, the error decreases. This can be attributed to the reduced fluctuations in the expected purchase probability function, leading to a more consistent trend that facilitates accurate distribution estimation by the algorithm. Furthermore, the estimation error does not exhibit significant variations across different distributions and m values. This finding highlights the robustness of the proposed algorithm, as it consistently performs well regardless of the specific characteristics of the dataset. With a preference dataset size of 5000, the estimation error is approximately 5%, which can be considered low for accurately estimating the distribution. This suggests that the proposed algorithm is reliable and effective in estimating the distribution of segments. In conclusion, based on the analysis of the simulation study, it can be affirmed that the proposed algorithm is robust and successful in estimating the distribution, especially when applied to datasets of reasonable size.

Table 3.19 Estimated segments' distribution by Distribution Estimation Algorithm using one feature vector in the presence of deterministic valuations

		True Distribution				Error %	True Distribution				Error %	True Distribution				Error %
		$\delta_1 = 0.1$	$\delta_2 = 0.2$	$\delta_3 = 0.3$	$\delta_4 = 0.4$		$\delta_1 = 0.4$	$\delta_2 = 0.3$	$\delta_3 = 0.2$	$\delta_4 = 0.1$		Random with $R = 0.1$				
		Estimated Distribution					Estimated Distribution					Estimated Distribution				
		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$	
$m = 20$	$m' = 50$	0.112	0.207	0.220	0.372	21.271	0.368	0.190	0.235	0.147	33.720	0.343	0.167	0.195	0.165	19.397
	$m' = 100$	0.099	0.200	0.284	0.392	13.594	0.398	0.295	0.201	0.102	8.964	0.379	0.181	0.181	0.147	13.900
	$m' = 150$	0.100	0.204	0.285	0.392	10.017	0.398	0.300	0.197	0.101	7.278	0.392	0.203	0.154	0.155	9.972
	$m' = 200$	0.099	0.199	0.298	0.399	6.586	0.400	0.300	0.202	0.101	6.038	0.403	0.212	0.148	0.145	7.585
	$m' = 250$	0.100	0.199	0.300	0.400	5.823	0.400	0.299	0.199	0.101	5.838	0.388	0.213	0.148	0.163	6.245
	$m' = 500$	0.100	0.200	0.298	0.402	3.610	0.400	0.301	0.199	0.100	3.680	0.390	0.209	0.145	0.146	4.713
	$m' = 1000$	0.100	0.201	0.301	0.401	2.768	0.400	0.299	0.203	0.100	2.733	0.384	0.233	0.142	0.147	3.192
$m = 50$	$m' = 20$	0.143	0.259	0.219	0.340	31.189	0.320	0.184	0.267	0.186	47.297	0.307	0.183	0.209	0.186	28.397
	$m' = 40$	0.113	0.217	0.223	0.373	17.497	0.358	0.169	0.254	0.164	41.558	0.348	0.161	0.220	0.173	18.370
	$m' = 60$	0.098	0.201	0.276	0.393	10.524	0.386	0.256	0.226	0.117	13.010	0.373	0.157	0.191	0.169	13.633
	$m' = 80$	0.093	0.197	0.289	0.394	8.343	0.398	0.287	0.206	0.106	8.203	0.366	0.199	0.180	0.159	10.236
	$m' = 100$	0.093	0.195	0.298	0.398	7.073	0.400	0.292	0.204	0.102	5.858	0.339	0.221	0.167	0.172	8.053
	$m' = 200$	0.099	0.200	0.299	0.399	4.148	0.400	0.300	0.202	0.099	3.847	0.412	0.185	0.150	0.152	5.453
	$m' = 400$	0.100	0.201	0.300	0.399	2.706	0.400	0.300	0.201	0.100	2.713	0.417	0.212	0.139	0.156	3.336

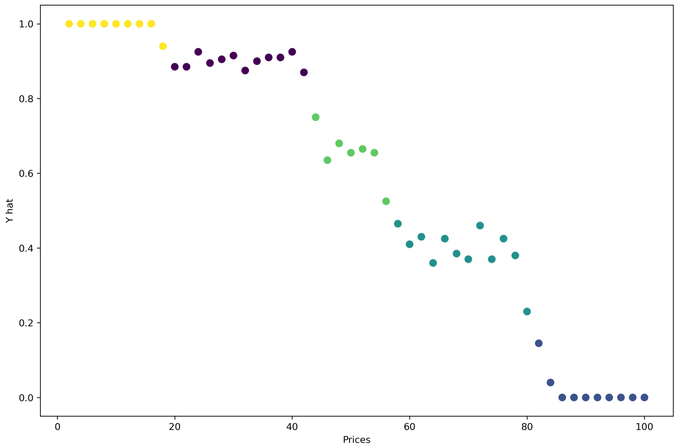
Based on the results presented in Table 3.19, it is evident that when only one feature vector is used for clustering, the resulting error is higher compared to the case where two feature vectors are considered. This can be attributed to the impact of fluctuations in the expected values on the clustering process when only one feature vector is utilized. The absence of additional information from the second feature vector may lead to less accurate clustering. However, it is worth noting that when the size of the dataset is large, the fluctuations in the expected values become less significant, and the data points are clustered more accurately even with the use of a single feature vector. This is evident from the estimation error of large datasets, which is quite close to the estimation error observed when two feature vectors are considered.

Overall, when utilizing a reasonably large dataset, the proposed algorithm demonstrates success in estimating the distribution regardless of the number of features used for clustering. While the inclusion of additional features may improve clustering accuracy in certain cases, the algorithm can still provide reliable estimations with sufficient data even when utilizing only one feature vector.

3.3.3.4 Probabilistic valuations

When probabilistic valuations are considered, the structure of the expected purchase probability function undergoes changes, as observed earlier. In Figure 3.10, the structure of the expected purchase probability function is illustrated in the presence of probabilistic valuations. Instead of sharp declines observed in the deterministic cases, a distinct discrete gradual decrease in the expected probability values can be observed.

Figure 3.10 Expected purchase probability as a function of discrete suggested prices in the presence of probabilistic valuations



3.3.3.5 Simulation

In this section, we aim to conduct a similar simulation study as we did in Section 3.3.3.3, with the difference being that the valuations are now probabilistic. This is done to assess the performance of the Distribution Estimation Algorithm in this particular scenario. The valuation percentage deviation ($p\%$) considered for this scenario is 5%. In this simulation, we consider both columns in the converted dataset as features for clustering. Considering the observations made in the simulation study conducted in Section 3.3.3.3 by using two feature vectors for clustering, where no significant difference was observed between different distributions, we will simplify the comparison between the deterministic and probabilistic cases by focusing on a single distribution. For the purpose of this comparison, we will consider a distribution consisting of values $\delta_i = [0.1, 0.2, 0.3, 0.4]$. This will allow us to specifically evaluate the performance and differences between the deterministic and probabilistic scenarios in estimating the distribution of segments. The results of this simulation study are presented in Table 3.20, providing insights into the algorithm's performance and accuracy in estimating the distribution of segments under probabilistic valuation conditions.

Table 3.20 Estimated segment distributions by Distribution Estimation Algorithm using two feature vectors in the presence of deterministic and probabilistic valuations

		Probabilistic Valuations ($p\% = 5\%$)					Deterministic Valuations				
		True Distribution				Error %	True Distribution				Error %
		$\delta_1 = 0.1$	$\delta_2 = 0.2$	$\delta_3 = 0.3$	$\delta_4 = 0.4$		$\delta_1 = 0.1$	$\delta_2 = 0.2$	$\delta_3 = 0.3$	$\delta_4 = 0.4$	
		Estimated Distribution					Estimated Distribution				
		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$		$\hat{\delta}_1$	$\hat{\delta}_2$	$\hat{\delta}_3$	$\hat{\delta}_4$	
$m = 20$	$m' = 50$	0.095	0.234	0.295	0.364	15.111	0.100	0.200	0.300	0.400	13.208
	$m' = 100$	0.098	0.229	0.296	0.366	11.521	0.098	0.200	0.299	0.400	9.653
	$m' = 150$	0.098	0.227	0.298	0.372	10.394	0.097	0.197	0.299	0.399	7.657
	$m' = 200$	0.100	0.229	0.297	0.370	9.328	0.099	0.200	0.301	0.400	6.389
	$m' = 250$	0.098	0.227	0.301	0.369	8.586	0.098	0.200	0.300	0.399	5.450
	$m' = 500$	0.100	0.229	0.300	0.368	7.933	0.100	0.199	0.301	0.400	4.082
	$m' = 1000$	0.100	0.230	0.300	0.369	7.010	0.100	0.199	0.298	0.401	2.820
$m = 50$	$m' = 20$	0.105	0.234	0.288	0.345	17.990	0.099	0.218	0.290	0.392	13.513
	$m' = 40$	0.099	0.212	0.297	0.367	11.259	0.101	0.200	0.299	0.400	8.456
	$m' = 60$	0.097	0.214	0.292	0.365	10.168	0.100	0.199	0.302	0.399	6.969
	$m' = 80$	0.097	0.205	0.299	0.368	8.120	0.101	0.201	0.298	0.398	6.281
	$m' = 100$	0.099	0.204	0.302	0.369	7.888	0.098	0.200	0.302	0.398	5.552
	$m' = 200$	0.097	0.210	0.296	0.368	6.387	0.100	0.199	0.299	0.399	3.629
	$m' = 400$	0.098	0.212	0.295	0.368	5.585	0.100	0.202	0.300	0.400	2.964

In the table, the median of the estimated values for the segments' distribution is provided, along with their corresponding average absolute percent deviation error (Error%). Upon examining the results, it is evident that for each m and m' value, the estimation error for probabilistic valuations is higher compared to the deterministic case. This discrepancy can primarily be attributed to the expected purchase probabilities. In the case of probabilistic valuations, the transition from one cluster to another is less distinct compared to the deterministic scenario. This leads to estimations with higher errors, as the algorithm encounters challenges in accurately capturing the subtle changes in the expected purchase probabilities. However, it is worth noting that with a reasonably large dataset, the proposed algorithm still demonstrates the capability to estimate the distribution with relatively low errors.

For instance, when utilizing a dataset size of 5000, the estimation error is around 8%. Although not negligible, this error can still be considered acceptable when estimating the distribution. In summary, while the estimation errors are higher for probabilistic valuations compared to deterministic cases, the algorithm's performance remains promising, particularly when employed on datasets of sufficient size.

3.4 Case where all of the parameters are unknown

In this section, we aim to estimate the number of segments (n) in the market while having the lowest level of foreknowledge about the market structure, including unknown values for n (number of segments), v_i (valuations), and δ_i (distribution of segments). However, we focus specifically on the scenario where deterministic valuations are present. The objective is to develop an algorithm that can infer the appropriate number of segments in the market based on the available data. In this case, we make use of the discrete preference dataset. By analyzing the patterns and characteristics of the discrete preference dataset, we aim to estimate the underlying segmentation of customers without prior knowledge of the specific market structure. Through this analysis, we can gain insights into the potential number of segments in the market, contributing to a better understanding of customer behavior and market dynamics in the presence of deterministic valuations.

3.4.1 Proposed algorithm for estimating the number of segments (Number of Segments Estimation Algorithm)

Our proposed algorithm for estimating the number of segments in the market, considering the lowest level of foreknowledge and the presence of deterministic valuations, is outlined as follows:

- Start with an assumption of a minimum number of segments ($n' = 1$)
- Apply Distribution Estimation Algorithm to the provided discrete preference dataset assuming $n = n'$

- Calculate the fitting error of clustered data points
- Feasibility check:
    - Deacresaing cluster centers
    - Difference between cluster centers > 0.05
- If feasibility checks True:
    - Increment n' and go to step 2.
- If feasibility checks False:
    - Stop applying the Distribution Estimation Algorithm and go to next step
- Pick the estimated distribution with the lowest fitting error.
- The number of elements in the estimated distribution vector is the estimated value for the number of segments ($\hat{n}$)

By following this algorithm, we aim to estimate the number of segments in the market by iteratively adjusting the assumed number of segments and evaluating the quality of the resulting segmentation. The algorithm enables us to adaptively determine the appropriate number of segments based on the characteristics of the preference dataset, providing insights into the market structure without prior knowledge.

In our proposed algorithm, we begin by assuming that the number of segments in the market is 1 ($n' = 1$). Subsequently, we employ the Distribution Estimation Algorithm on the given preference dataset, assuming that the number of segments is n' ($n = n'$). By implementing the Distribution Estimation Algorithm, we obtain an estimated distribution ($\hat{\delta}_i$) for the segments and clustered data points. Following this, we proceed to calculate the fitting error of the clustered data points. This error is calculated using the following formula:

$$error_{fitting} = \frac{1}{n'+1} \sum_{i=1}^{n'+1} \sum_{j=1}^{|cluster_i|} \hat{y}_{ij} - center_i \tag{3.14}$$

In which n' is the assumed number of segments, $|cluster_i|$ is the length of cluster i, $\hat{y}_{ij}$ is the $\hat{y}$ value for element j in cluster i, and $center_i$ is the center of cluster i in y axis.

The fitting error quantifies the level of agreement between the observed preference values and the expected values derived from the estimated distribution. This metric serves as an indicator of how well the estimated distribution aligns with the actual data points, enabling the evaluation of the accuracy of the algorithm in capturing the underlying segmentation of the market.

In each iteration of the algorithm, after applying the Distribution Estimation Algorithm and obtaining the estimated distribution and fitting error, a feasibility check is performed. This feasibility check aims to verify if the centers of the clusters in the y-axis exhibit a decreasing trend and if the differences between these centers are at least 0.05. The rationale behind this feasibility check is as follows: As mentioned earlier, when transitioning from one cluster to the next, a portion of customers are lost due to an increase in valuations. Consequently, the purchase probabilities represented by the cluster centers in the y-axis should generally exhibit a decreasing pattern. If the cluster centers do not demonstrate a decreasing trend, it suggests that the clustering may not be based on an appropriate number of segments. Moreover, the choice of 0.05 as the threshold for differences between cluster centers is motivated by the idea that if these differences are less than this value, it indicates that the data points within these clusters likely belong to a single cluster rather than multiple distinct clusters. By conducting this feasibility check, we can assess the suitability of the clustering and make informed decisions regarding the number of segments in the market based on the observed trends and differences in the cluster centers.

After performing the feasibility checks and determining that they have become false, the algorithm proceeds to the next step. In this step, the estimated distribution with the lowest fitting error from the stored vector is selected. The stored vector contains the estimated distributions and their corresponding fitting errors obtained during the iterative process. By selecting the estimated distribution with the lowest fitting error, we prioritize the distribution that best aligns with the actual data points. The number of elements in this selected estimated distribution vector serves as the estimated value for the number of segments ($\hat{n}$) in the market. This value represents the algorithm's estimation of the underlying segmentation based on the preference dataset and the iterative process of applying the Distribution Estimation Algorithm, feasibility checks, and selection based on fitting error. By following this approach, the algorithm leverages the fitting error as a criterion for choosing the most suitable estimated distribution, which in turn provides an estimation for the number of segments in the market.

3.4.2 Simulation

In order to evaluate the performance of the proposed Number of Segments Estimation Algorithm in estimating the number of segments in the presence of deterministic valuations without any prior knowledge of the market structure, two simulation studies were conducted. These studies involved considering one distribution $\delta_i = [0.1, 0.2, 0.3, 0.4]$ and various values for m and m'.

In the first simulation study, both columns in the converted dataset were used for clustering. The valuations considered for this study were $v_i = [18, 43, 56, 81]$. The objective was to investigate the effect of using different numbers of features for clustering and assess the algorithm's performance under such conditions. In the second simulation study, only the $\hat{y}$ values column in the converted dataset was utilized for clustering. The valuations considered for this study were $v_i = [8, 46, 56, 81]$. The purpose of this study was to explore the impact of non-uniformity in valuations on the performance of the algorithm. Both simulation studies consisted of 200 replications to ensure robust results. The outcomes of these simulation studies are presented in Tables 3.21 and 3.22. These tables provide a comprehensive overview of the algorithm's performance in estimating the number of segments under different scenarios.

By analyzing the results, we can assess the algorithm's effectiveness and gain insights into the impact of using different numbers of features for clustering and non-uniformity in valuations on the accuracy of the estimated number of segments.

Table 3.21 Probability of estimating the true number of segments using two feature vectors in the presence of deterministic valuatoins

		Probability of finding the true number of segments			Probability of finding the true number of segments
$m = 20$	$m' = 50$	0.53	$m = 50$	$m' = 20$	0.34
	$m' = 100$	0.67		$m' = 40$	0.43
	$m' = 150$	0.78		$m' = 60$	0.58
	$m' = 200$	0.89		$m' = 80$	0.79
	$m' = 250$	0.91		$m' = 100$	0.85
	$m' = 500$	0.98		$m' = 200$	0.98
	$m' = 1000$	1.00		$m' = 400$	1.00

In Table 3.21, the probability of accurately estimating the true number of segments using two feature vectors in the converted dataset is reported for different values of m and m'.

The results indicate that as the size of the dataset increases, the probability of correctly estimating the number of segments also increases. This can be attributed to the fact that with a larger dataset, the expected purchase probability function exhibits reduced fluctuations. The function maintains a more consistent and stable trend, enabling the algorithm to cluster the data points more accurately and estimate the parameters with higher precision.

Moreover, when the dataset size remains the same, by offering 20 distinct prices ($m = 20$), the probability of accurate estimation tends to be higher. This is due to the fact that each price is offered more frequently (higher m' values), resulting in a more robust and reliable expected purchase probability function. Consequently, the algorithm can estimate the parameters more accurately in such scenarios.

With a dataset size of 10,000, the probability of accurately estimating the number of segments is reported to be 98%. This high probability verifies the success of the proposed algorithm in accurately estimating the number of segments. Therefore, if the seller has the opportunity to obtain a larger dataset, it is recommended to do so as it increases the probability of accurately estimating the number of segments and improving the overall effectiveness of the algorithm.

Table 3.22 Probability of estimated number of segments using one feature vector in the presence of deterministic valuatoins

		Probability of estimated number of segments						
		$\hat{n}=1$	$\hat{n}=2$	$\hat{n}=3$	$\hat{n}=4$	$\hat{n}=5$	$\hat{n}=6$	$\hat{n}=7$
$m=20$	$m'=50$	0.000	0.000	0.000	0.020	0.085	0.330	0.565
	$m'=100$	0.000	0.000	0.000	0.090	0.315	0.345	0.250
	$m'=150$	0.000	0.000	0.000	0.255	0.420	0.280	0.045
	$m'=200$	0.000	0.000	0.000	0.330	0.490	0.160	0.020
	$m'=250$	0.000	0.000	0.000	0.470	0.445	0.080	0.005
	$m'=500$	0.000	0.000	0.000	0.765	0.225	0.010	0.000
	$m'=1000$	0.000	0.000	0.000	0.955	0.045	0.000	0.000
		Probability of estimated number of segments						
		$\hat{n}=1$	$\hat{n}=2$	$\hat{n}=3$	$\hat{n}=4$	$\hat{n}=5$	$\hat{n}=6$	$\hat{n}=7$
$m=50$	$m'=20$	0.000	0.000	0.000	0.000	0.000	0.015	0.985
	$m'=40$	0.000	0.000	0.000	0.000	0.020	0.085	0.895
	$m'=60$	0.000	0.000	0.000	0.000	0.035	0.120	0.845
	$m'=80$	0.000	0.000	0.000	0.000	0.130	0.225	0.645
	$m'=100$	0.000	0.000	0.000	0.010	0.285	0.270	0.435
	$m'=200$	0.000	0.000	0.000	0.180	0.565	0.240	0.015
	$m'=400$	0.000	0.000	0.000	0.760	0.220	0.020	0.000

Table 3.22 presents the probability of estimating different values as the number of segments ($\hat{n}$). Upon analysis, it is observed that when only one feature vector is considered, the probability of estimating the number of segments as 4 (which is the true value) is lower compared to the case where two feature vectors are used. Furthermore, with $m = 20$, these probabilities tend to be higher compared to the case where $m = 50$. This suggests that when a smaller number of distinct prices is offered, the algorithm is more likely to estimate the true number of segments accurately. In scenarios with smaller datasets, the estimated number of segments is generally greater than 4. However, as the size of the dataset increases, this estimate converges towards the true value of 4. This implies that with a larger dataset, the algorithm becomes more effective in accurately estimating the number of segments.

In summary, the results in Table 3.22 indicate that the probability of correctly estimating the number of segments is influenced by the number of feature vectors used for clustering, the number of distinct prices offered (m), and the size of the dataset. Employing two feature vectors tends to improve the accuracy of the estimation, and a larger dataset enhances the algorithm's ability to converge towards the true number of segments.

4. Online Learning

In the previous chapter, the focus was primarily on learning the unknown parameters of the problem using a full dataset. However, in this chapter, the objective shifts towards maximizing the reward of the problem. To achieve this, an additional component called online learning is introduced to the offline learning process. The online learning phase follows offline learning meaning we initialize online learning with the estimates obtained in the offline learning phase. In online learning, we no longer have access to the entire dataset upfront. Instead, customers arrive individually, and upon their arrival, we offer them a price and receive feedback regarding their purchase preference (i.e., whether they buy or not). Consequently, each time a customer arrives, we obtain a data point (x, y) consisting of the offered price x and the customer's buying preference y. We incorporate this data point into our original dataset, update the converted dataset, and subsequently re-estimate the parameters. This iterative process, performed for a specific number of iterations (n_{iter}), constitutes online learning. By incorporating online learning into the overall learning process, the research aims to enhance reward maximization. This iterative approach allows for continuous adaptation and updating of the estimated parameters based on real-time customer interactions and feedback. The integration of online learning offers a more dynamic and adaptive approach to reward optimization.

The key distinction between offline and online learning lies in the incremental nature of data acquisition in online learning, where the dataset gradually expands with the arrival of each customer. Additionally, unlike offline learning, where prices are uniformly offered within the range of $(0, 100)$, online learning adopts a more strategic and structured approach to price offerings.

4.1 Deterministic Valuations

In the first scenario, the problem is investigated in the presence of deterministic valuations. It is assumed that the number of segments (n) and their distribution (δ_i) are known, while the customers' valuations (v_i) remain unknown. In the context of online learning, there are two main objectives to consider. The first objective is to improve the estimates of the unknown parameters (customers' valuations in this scenario) that were obtained from the offline learning process. By incorporating real-time customer interactions and feedback, online learning allows for continuous refinement and enhancement of these parameter estimates. The second objective is to maximize the achieved reward during the online learning phase by offering different prices.

To obtain more accurate estimates of customers' valuations, it is not necessary to offer prices uniformly across the entire price range. Instead, we should focus on offering prices in the vicinity of estimated valuations. The rationale behind this lies in the fact that obtaining more accurate valuation estimates requires additional information within the neighborhood of valuations to distinguish between clusters. Since, for each element (x, y) in the preference dataset, we consider only the k nearest data points to calculate $\hat{\hat{y}}$ values, it suffices to offer prices within that neighborhood. By doing so, the offered prices can influence the $\hat{\hat{y}}$ values of the estimated valuations, leading to a more precise estimation. Regarding the objective of reward maximization, it is logical to offer customers the estimated valuations. For a given segment i, when prices are offered in the interval $(v_{i-1}, v_i]$, all customers within that segment will make a purchase since the offered prices are lower than their valuations. Higher prices result in increased revenue; thus, offering v_i to customers maximizes revenue.

Considering these factors, our approach for price offerings entails presenting one of the estimated valuations $\hat{v}_i$ to incoming customers. In the online learning phase, we employ the epsilon-greedy technique. At each iteration, there is a probability ϵ of offering the estimated valuation that yields the highest reward, while with a probability of $1 - \epsilon$, we randomly offer one of the other estimated valuations to the incoming customer.

4.1.1 Proposed Online Algorithm 1

The proposed algorithm is as follows:

- Provided with dataset of size T, estimate the valuations in offline mode by Algorithm 1 ($\hat{v}_i^{initial}$)

- $\hat{v}_i = \hat{v}_i^{initial}$
- For n_{iter} iterations:
    - Offer price x to the incoming customer
        - Calculate the revenue yielded by each $\hat{v}_i$
        - With probability ϵ offer the $\hat{v}_i$ that yields the highest revenue
        - With probability $1-\epsilon$ offer one of the other $\hat{v}_i$'s randomly
    - Getting the customers' feedback y
    - Add the new data point (x,y) to the existing dataset
    - Re-calculate the $\hat{\hat{y}}$ values of the data points in the converted dataset
    - Re-cluster
    - Re-estimate the customers' valuations and replace the $\hat{v}_i$'s with the new estimates

The initial step involves utilizing a dataset comprising T data points to derive an initial estimate of valuations ($\hat{v}_i^{initial}$) by offline learning using Algorithm 1. Throughout this process, we make certain assumptions, namely that the order and distribution of segments are known in advance, and that valuations are deterministic. Upon obtaining the initial valuation estimates, we transition to the online learning phase.During each iteration of the online learning phase, we offer a price to the incoming customer. To determine these prices, we first calculate the revenue generated by the latest valuation estimates ($\hat{v}_i$) using the following equation:

$$revenue_i = ((1 - \sum_{j=1}^{i-1} \delta_j) \times \hat{v}_i) \quad i = 1,...,n \tag{4.1}$$

In which δ_i is the probability of customers being in segment i, and $\hat{v}_i$ is the estimated valuation of segment i.

Subsequently, we adopt the epsilon-greedy approach, where with probability ϵ, we offer the $\hat{v}_i$ that yields the highest $revenue_i$, and with probability $1-\epsilon$, we randomly select one of the other $\hat{v}_i$'s. The offered price (x) is presented to the customer, and we collect their feedback (y). Next, we incorporate the newly acquired data point (x,y) into our existing dataset and recalculate the $\hat{\hat{y}}$ values for each element in the dataset, resulting in the creation of the new converted dataset. We then re-cluster the data points, employing the method described in Algorithm 1. Finally, based on the clusters identified, we re-estimate the valuations. This iterative process is repeated for n_{iter} iterations to obtain the final estimated valuations ($\hat{v}_i^{final}$).

4.1.2 Simulation

In order to evaluate the performance of Online Algorithm 1 in terms of valuations' accuracy and the average reward achieved during the online learning phase, a comprehensive simulation study will be conducted. This study will encompass various market structures to ensure a thorough assessment. The selected settings for evaluating the algorithm's performance are presented in Table 4.1. The simulation study focuses on a scenario where the dataset undergoes two conversions, with the outliers eliminated using the IQR method. We will investigate three distributions: increasing distribution, decreasing distribution, and random distribution. To begin, a warm-up phase will be initiated, consisting of 1000 data points to obtain initial estimates of customers' valuations. Subsequently, the online learning phase will proceed for 1000 iterations to improve these estimates and observe the average reward collected during this phase. Different values of ϵ will be considered to examine the impact of this hyperparameter on the algorithm's performance and the resulting outcomes. To ensure a robust evaluation of the algorithm's performance across diverse scenarios, each setting will undergo 100 replications. This rigorous approach guarantees a comprehensive assessment of the algorithm's performance in various scenarios.

Table 4.1 Setting of the simulation study for estimating the valuations by online learning in the presence of deterministic valuations and known distribution

Number of segments	n	4
Number of data points	T	1000
Number of iterations in online learning phase	n_{iter}	1000
Number of nearest neighbors	k	50
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Minimum probability for random distribution	R	0.15
Epsilon	ϵ	[0.5, 0.6, 0.7]

Table 4.2 Simulation results for initial and final estimated valuations by online learning in the presence of deterministic valuations and known distribution

	$\delta_i = [0.1, 0.2, 0.3, 0.4]$										
	Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
	$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.5$	20.200	45.017	54.933	79.908	20.039	45.000	54.999	80.000	0.133	28.913	40.500
$\epsilon = 0.6$	20.419	45.176	54.931	79.870	20.055	45.000	54.587	80.000	0.620	30.176	40.500
$\epsilon = 0.7$	19.806	44.994	54.835	79.982	20.315	45.000	53.201	80.000	1.671	31.764	40.500
	$\delta_i = [0.4, 0.3, 0.2, 0.1]$										
	Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
	$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.5$	20.082	44.783	55.394	79.551	20.000	45.000	54.997	79.988	0.048	15.651	27.000
$\epsilon = 0.6$	20.007	45.038	54.858	79.342	20.000	45.000	54.741	79.944	0.297	16.654	27.000
$\epsilon = 0.7$	20.213	44.822	55.004	79.800	20.011	45.000	53.545	79.945	1.333	17.513	27.000
	$\delta_i =$ **Random with** $R = 0.15$										
	Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
	$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.5$	20.182	45.047	55.074	80.079	20.000	45.000	54.949	80.000	0.118	18.316	29.025
$\epsilon = 0.6$	20.099	45.008	55.149	79.865	20.000	45.000	54.592	79.998	0.563	19.019	28.350
$\epsilon = 0.7$	19.818	44.761	55.222	79.768	20.042	45.000	53.235	79.997	1.560	20.841	29.925

The simulation results, representing the median values over 100 replications, are presented in Table 4.2. The first four columns provide the initial estimated valuations ($\hat{v}_i^{initial}$) obtained through offline learning. Subsequently, the following columns present the final estimates of valuations ($\hat{v}_i^{final}$) after applying online learning.

The "Error%" metric indicates the average absolute percent deviation between the final estimates and the true valuations. It is calculated as follows:

$$\frac{1}{n}\sum_{i=1}^{n}\left(\left|\frac{\hat{v}_i^{final} - v_i}{v_i}\right|\right) \tag{4.2}$$

The "Average Reward" metric corresponds to the collected reward during the online learning phase. In this phase, the seller offers a price to incoming customers, and if a customer purchases the product, the seller earns a reward. The reported reward values in the results represent the average of rewards collected during the online learning phase:

$$reward_{average} = \frac{1}{n_{iter}}\sum_{j=1}^{n_{iter}} x^j \times y^j \tag{4.3}$$

In which x^j is the offered price to the j^{th} incoming customer in the online learning phase, and y^j is the customer's buying preference.

The "Expected Reward" metric refers to the theoretical reward that is expected to be earned in the long run. It is calculated based on the true parameters of the problem:

$$reward_{expected} = \max_{i=1,...,n} \{(1 - \sum_{j=1}^{i-1} \delta_j) \times v_i\} \tag{4.4}$$

As the epsilon value increases, there is a corresponding increase in the estimation error. This can be attributed to the fact that a higher epsilon value emphasizes the estimated valuation with the highest reward, generating more data points in its vicinity. Consequently, more information is generated for this particular valuation, leading to a more accurate estimation. However, the improvement in the estimation of other valuations is limited. Conversely, a lower epsilon value allows for the improvement of all valuations, resulting in a decrease in estimation error.

Furthermore, an increase in the epsilon value leads to an increase in the average collected reward during the online learning phase, bringing it closer to the expected reward. This effect is observed because a higher epsilon value results in offering the price with the highest reward more frequently over multiple iterations, thereby increasing the average collected reward.

It is evident that there exists a trade-off between estimation error and the average collected reward. Therefore, the choice of the epsilon value is dependent on the seller's objective. If the seller aims to achieve more accurate valuations, a low epsilon value should be selected. Conversely, if the goal is to maximize the collected reward, increasing the epsilon value would be more suitable.

In other words, with low values of epsilon, the seller focuses on exploration, actively seeking to gather more information and improve the estimation of valuations. This exploration allows for a more comprehensive understanding of the valuations and reduces the estimation error. On the other hand, high values of epsilon prioritize exploitation, where the seller capitalizes on the valuation with the highest reward. By exploiting this high-reward valuation, the seller can maximize their collected reward. However, this emphasis on exploitation may result in less improvement in the estimation of other valuations, leading to a higher estimation error. In summary, low epsilon values promote exploration and lead to improved valuation estimates, while high epsilon values prioritize exploitation and maximize the collected reward.

4.1.3 Asymmetric penalty for estimation errors

An important problem encountered in the online learning phase in the presence of deterministic product valuations is that the penalty for estimation errors is not symmetrical. This asymmetry can be explained with an example: When the product valuations of the segments (v_i) are $[20, 45, 55, 80]$, respectively, and the distribution of the segments (δ_i) is $[0.1, 0.2, 0.3, 0.4]$, the strategy that maximizes the total reward is $P^* = 45$ (Calculated by Eq. 2.1). When this price is offered, the expected reward will be $(45) \times (0.9) = 40.5$ (Calculated by Eq. 4.4), since all customers will buy except the first segment. If this value is calculated as 44 as a result of the estimation error, when the price offered to the customer is 44, the expected reward will be $(44) \times (0.9) = 39.6$, since all customers will buy again except the first segment; if the estimate is calculated as 46 and the price offered to the customer is 46, this time the expected reward will decrease to $(46) \times (0.7) = 32.2$, since all customers in the second segment will lose together with the first segment. As observed, the estimation error for these two cases are similar, but the consequences of this error differ significantly. The reward lost due to the asymmetry between the valuations' estimation carries substantial importance.

This asymmetry created by the same amount of estimation error has led to the update of the exploitation step, where the price expected to bring the highest reward in the online learning phase will be proposed. In the case of full exploitation ($\epsilon = 1$), at each iteration of the online learning phase, the price to be offered is the estimated valuation that brings the highest reward. Let's denote this value as $price_{estimated}$. The recommendation at this stage for overcoming the estimations error is to offer a price α units below the $price_{estimated}$ to the incoming customer at each iteration of the online learning phase. We denote this offered price to the customers as $price_{offered}$:

$$price_{offered} = price_{estimated} - \alpha \tag{4.5}$$

This α value functions as a hyperparameter in our algorithm.

4.1.4 Simulation

In order to investigate the impact of the new hyperparameter α on the reward collected during the online learning phase, a simulation study will be conducted. The simulation settings will be similar to the previous simulation study discussed in Section 4.1.2. However, in this study, we will increase the values for the hyperparameter ϵ to assess its impact on approaching the expected reward. For the hyperparameter α, a value of 0.5 will be considered. Additionally, we will examine cases where α is not applied in order to observe its effect on the collected reward. By comparing the results between the cases with and without α, we can assess the significance of this hyperparameter in optimizing the reward collection process during the online learning phase. The selected settings for evaluating the algorithm's performance are presented in Table 4.3.

Table 4.3 Setting of the simulation study for estimating the valuations by online learning with applying alpha hyperparameter in the presence of deterministic valuations and known distribution

Number of segments	n	4
Number of data points	T	1000
Number of iterations in online learning phase	n_{iter}	1000
Number of nearest neighbors	k	50
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Minimum probability for random distribution	R	0.15
Epsilon	ϵ	[0.90, 0.95, 0.99]
Alpha	α	[0.0, 0.5]

The simulation results, representing the median values over 100 replications, are presented in Table 4.4. Once again, we observe that as the ϵ value increases, the estimation error also increases. In this study, the errors are higher compared to the previous study, and we have already discussed the reasons behind this observation. The increased errors can be attributed to the emphasis on exploitation rather than exploration, which limits the improvement in estimating valuations accurately. When considering the same ϵ value, applying α leads to significantly higher errors compared to the case where α is not applied. This result suggests that the interruption of the learning process by offering prices different than the estimated values hinders the accurate estimation of valuations.

Table 4.4 Simulation results for initial and final estimated valuations by online learning with applying alpha hyperparameter in the presence of deterministic valuations and known distribution

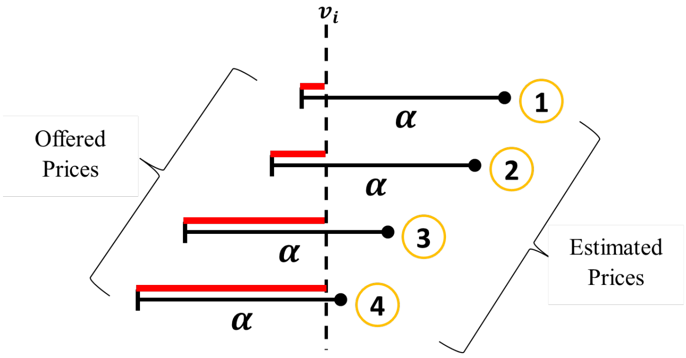
		$\delta_i = [0.1, 0.2, 0.3, 0.4]$										
		Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
		$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.90$	$\alpha = 0$	20.174	44.734	54.876	80.092	23.071	45.000	50.612	79.968	5.956	34.121	40.500
	$\alpha = 0.5$	20.187	44.852	54.886	79.989	40.384	44.991	55.297	80.470	25.834	38.224	40.500
$\epsilon = 0.95$	$\alpha = 0$	19.970	45.194	55.029	79.733	23.585	45.000	50.353	79.978	6.612	34.835	40.500
	$\alpha = 0.5$	20.365	45.181	54.966	79.830	40.544	45.044	55.300	80.367	26.078	38.726	40.500
$\epsilon = 0.99$	$\alpha = 0$	20.281	45.049	54.778	79.883	25.425	45.000	50.045	79.919	8.982	35.255	40.500
	$\alpha = 0.5$	20.128	45.103	55.006	79.985	40.811	45.000	55.148	80.023	26.839	39.291	40.500
		$\delta_i = [0.4, 0.3, 0.2, 0.1]$										
		Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
		$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.90$	$\alpha = 0$	20.263	45.014	54.983	79.579	23.265	45.000	50.523	79.658	6.500	19.475	27.000
	$\alpha = 0.5$	20.029	44.953	54.834	79.854	27.041	45.129	55.192	79.940	11.904	24.033	27.000
$\epsilon = 0.95$	$\alpha = 0$	20.084	45.100	54.980	79.823	24.581	45.000	50.303	79.838	8.639	19.878	27.000
	$\alpha = 0.5$	19.950	45.146	55.059	79.573	35.992	45.114	55.302	79.142	20.963	24.525	27.000
$\epsilon = 0.99$	$\alpha = 0$	20.052	45.068	54.959	79.420	26.158	45.000	50.237	79.686	10.484	20.252	27.000
	$\alpha = 0.5$	20.046	45.220	55.039	79.735	37.956	45.085	55.200	78.647	23.793	24.817	27.000
		$\delta_i =$ **Random with** $R = 0.15$										
		Initial Estimated Valuations				**Final Estimated Valuations**				**Error%**	**Average Reward**	**Expected Reward**
		$\hat{v}_1^{initial}$	$\hat{v}_2^{initial}$	$\hat{v}_3^{initial}$	$\hat{v}_4^{initial}$	$\hat{v}_1^{final}$	$\hat{v}_2^{final}$	$\hat{v}_3^{final}$	$\hat{v}_4^{final}$			
$\epsilon = 0.90$	$\alpha = 0$	20.215	44.949	55.358	79.829	21.892	45.000	50.865	79.933	4.451	23.166	30.825
	$\alpha = 0.5$	20.113	44.778	55.311	79.802	29.387	44.918	55.186	80.363	12.885	26.728	29.700
$\epsilon = 0.95$	$\alpha = 0$	20.028	44.945	55.187	79.708	23.516	45.000	50.518	79.989	7.116	23.364	30.150
	$\alpha = 0.5$	20.020	45.018	54.820	79.858	35.264	44.987	54.844	80.450	20.488	27.032	30.150
$\epsilon = 0.99$	$\alpha = 0$	20.113	44.876	54.976	79.711	24.453	45.000	50.430	79.808	8.631	22.698	28.575
	$\alpha = 0.5$	19.993	45.120	54.627	79.823	35.253	44.893	54.499	80.083	21.173	28.330	30.375

As the ϵ value increases, the collected reward moves closer to the expected reward. The rewards obtained in this study are higher than those in the previous study, which was expected as the emphasis is on exploitation rather than exploration. For the same ϵ value, the application of α leads to a significant increase in the average collected reward. This indicates that offering prices below the estimated valuations, as facilitated by α, contributes to enhancing the overall reward. Specifically, in the case of the increasing distribution with $\epsilon = 0.99$ and $\alpha = 0.5$, an average reward of 97.01% of the expected reward was achieved during the online learning phase.

4.1.5 Decaying Alpha

In the previous section, we observed that deducting a fixed value of α from the estimated price offered to customers proves beneficial, leading to higher rewards in the online learning process. However, this method presents a challenge. As we repeatedly re-estimate valuations during the online phase, our estimates become more accurate and approach the true valuations. Nevertheless, by subtracting a fixed value from these estimates to derive the offered price, we eventually fall short of the true valuations, resulting in a loss of rewards due to the underestimation. As depicted in Figure 4.1, the first and second estimates of v_i deviate significantly from its true value. Offering these estimates as prices to customers would lead to the loss of customers from segment i, consequently decreasing our reward. The third and fourth estimates of v_i are much closer to its true value, but still, if we were to offer these estimates as prices to customers, we would lose rewards due to the overestimation of the true value of v_i. However, by deducting α from the estimated values, we can potentially gain rewards since the offered prices would be lower than the valuations of customers in segment i. Nonetheless, after deducting α, the disparity between the offered price and the true value (represented by red lines) remains considerable for the third and fourth estimates, resulting in a partial loss of reward. If we could reduce these differences, we would achieve greater rewards.

Figure 4.1 Effect of hyperparameter α on the obtained reward



Our proposed approach suggests that instead of utilizing a fixed value for α throughout the entire online learning phase, we decay the value of α as the estimates of a valuation approach its true value. This adjustment ensures that the offered price is sufficiently close to the true value, leading to increased rewards. To implement the decaying alpha approach, we examine the moving average of estimated prices. We posit that when the moving averages exhibit significant fluctuations, it indicates that the estimates are distant from the true value, prompting the algorithm to improve its estimation of a valuation towards its true value. In such cases, a larger value of α is required. Conversely, when the moving averages exhibit minimal fluctuation and maintain a steady trend, it is likely that the algorithm has converged to the true value of the valuation it seeks to estimate. Thus, we can decay the α value. Based on this rationale, by monitoring the moving averages of the estimated prices, we can decay the value of α whenever the moving averages maintain a steady trend. The $decay_{window}$ serves as a hyperparameter in our algorithm, defining the window within which we calculate the moving averages. Furthermore, we require a metric to determine if the trend of the moving averages remains stable. To address this, we introduce a new hyperparameter, $decay_{threshold}$. Whenever the difference between the moving averages is below this threshold, we can assume a steady trend among the moving averages. However, we cannot decay α whenever this difference is below the $decay_{threshold}$. Otherwise, α would quickly converge to zero, rendering the concept of decaying alpha meaningless. Moreover, the intuition behind decaying α based on the difference between moving averages is that the estimates become stable, implying that they are in proximity to the true values. However, it is possible for the difference to be small while the estimates are still far from the true values. This can occur when the algorithm becomes stuck in a different region rather than the vicinity of the true values. To overcome this limitation, we introduce another hyperparameter called $decay_{step}$. Instead of decaying α whenever the difference falls below the threshold, we reduce α if the differences remain below the threshold for a consecutive number of $decay_{step}$ times. This ensures that the estimates stabilize within a neighborhood before α is decayed. If the estimates change, we halt the reduction of α until they stabilize again. This approach prevents α from quickly converging to zero, allowing its effect to be traced during the learning process. Lastly, for decaying α, we introduce a $decay_{rate}$ hyperparameter, which determines the rate of decay. The process of decaying α can be summarized as follows:

- At each iteration $i > decay_{window}$ of the online learning phase:
  - Calculate the moving average of estimated prices (MA_i) through $decay_{window}$ window size
  - Calculate the absolute difference between two consecutive moving averages as:
    $diff = \frac{|MA_i - MA_{i-1}|}{MA_{i-1}}$
  - If $diff \leq decay_{threshold}$ for $decay_{step}$ iterations consecutively :
    Decay alpha by $decay_{rate}$

We initiate the computation of moving averages after a specified number of $decay_{window}$ iterations. Prior to this point, there are not enough estimates to calculate the moving averages. Additionally, since we are examining consecutive moving averages, we begin calculating the differences after the $decay_{window}$ iterations. Hence, we have established the condition $i > decay_{window}$. Subsequently, we compute the absolute differences between consecutive moving averages. If these differences consistently fall below the $decay_{threshold}$ for a consecutive number of $decay_{step}$ iterations, we proceed with the decay of alpha by a factor of $decay_{rate}$ rate. This approach guarantees that alpha is decayed only when the consecutive absolute differences satisfy the specified criterion.

4.1.6 Simulation

To examine the impact of decaying alpha on the collected reward in the online learning phase, a simulation study will be conducted. The hyperparameter α will be assigned an initial value of 0.5, which will progressively decrease using the decaying alpha method as the learning process advances. A value of 1 will be chosen for ϵ, representing full exploitation. The chosen settings for evaluating the algorithm's performance are outlined in Table 4.5. Each specific setting will be replicated 100 times to ensure robustness and reliable results.

Table 4.5 Setting of the simulation study during the online learning phase with applying decaying alpha in the presence of deterministic valuations and known distribution

Number of segments	n	4
Number of data points	T	1000
Number of iterations in online learning phase	n_{iter}	1000
Number of nearest neighbors	k	50
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Epsilon	ϵ	1
Alpha	α	0.5
Decay rate	$decay_{rate}$	[0.70, 0.80, 0.90, 0.95, 0.99]
Decay threshold	$decay_{threshold}$	[0.000, 0.010, 0.001]
Decay step	$decay_{step}$	50
Moving average window size	$decay_{window}$	[10, 25, 50, 100]

The results of the simulation study are presented in Table 4.6. The first column displays the median of the average rewards obtained during the online learning phase. The subsequent column represents the final value of α after undergoing multiple decays. The third column indicates the number of times α has been decayed. Upon analyzing the results, it is evident that the implementation of decaying alpha has led to an improvement in the collected reward, surpassing the rewards obtained in the previous simulation study. Notably, in the case of $decay_{window} = 100$, $decay_{rate} = 0.99$, and $decay_{threshold} = 0.000$, the collected reward reached a value of 39.927, which closely approximates the expected reward by 98.6%. This significant enhancement in rewards underscores the effectiveness of decaying alpha as a method to maximize rewards during the learning process.

Table 4.6 Simulation results for obtained average reward during online learning phase with applying decaying alpha in the presence of deterministic valuations and known distribution

		$\delta_i = [0.1, 0.2, 0.3, 0.4]$								
		$decay_{threshold} = 0.000$			$decay_{threshold} = 0.010$			$decay_{threshold} = 0.001$		
		Average Reward	**Final Alpha**	**#Alpha**	**Average Reward**	**Final Alpha**	**#Alpha**	**Average Reward**	**Final Alpha**	**#Alpha**
$decay_{window} = 10$	$decay_{rate} = 0.70$	36.682	0.000	57.000	38.941	0.000	20.000	38.567	0.000	25.000
	$decay_{rate} = 0.80$	37.413	0.000	74.000	39.406	0.006	20.000	39.320	0.002	26.000
	$decay_{rate} = 0.90$	39.211	0.002	53.000	39.714	0.061	20.000	39.590	0.032	26.000
	$decay_{rate} = 0.95$	39.639	0.083	35.000	39.766	0.179	20.000	39.753	0.132	26.000
	$decay_{rate} = 0.99$	39.694	0.389	25.000	39.640	0.409	20.000	39.846	0.385	26.000
$decay_{window} = 25$	$decay_{rate} = 0.70$	37.699	0.000	38.000	38.999	0.000	20.000	38.879	0.000	21.000
	$decay_{rate} = 0.80$	38.474	0.000	41.500	39.318	0.006	20.000	39.354	0.005	21.000
	$decay_{rate} = 0.90$	39.538	0.014	34.000	39.806	0.061	20.000	39.639	0.055	21.000
	$decay_{rate} = 0.95$	39.820	0.132	26.000	39.641	0.179	20.000	39.779	0.170	21.000
	$decay_{rate} = 0.99$	39.860	0.401	22.000	39.816	0.409	20.000	39.812	0.405	21.000
$decay_{window} = 50$	$decay_{rate} = 0.70$	38.634	0.000	25.000	38.970	0.001	19.000	39.049	0.001	19.000
	$decay_{rate} = 0.80$	39.055	0.001	26.500	39.359	0.007	19.000	39.379	0.007	19.000
	$decay_{rate} = 0.90$	39.538	0.036	25.000	39.681	0.068	19.000	39.653	0.068	19.000
	$decay_{rate} = 0.95$	39.766	0.162	22.000	39.749	0.189	19.000	39.780	0.189	19.000
	$decay_{rate} = 0.99$	39.731	0.409	20.000	39.808	0.413	19.000	39.712	0.413	19.000
$decay_{window} = 100$	$decay_{rate} = 0.70$	39.293	0.002	15.000	39.124	0.001	18.000	39.009	0.001	18.000
	$decay_{rate} = 0.80$	39.451	0.014	16.000	39.461	0.009	18.000	39.482	0.009	18.000
	$decay_{rate} = 0.90$	39.598	0.075	18.000	39.705	0.075	18.000	39.669	0.075	18.000
	$decay_{rate} = 0.95$	39.884	0.220	16.000	39.703	0.199	18.000	39.890	0.199	18.000
	$decay_{rate} = 0.99$	39.927	0.421	17.000	39.903	0.417	18.000	39.698	0.417	18.000

To explore the convergence of the collected average reward towards the expected reward, a further simulation study was conducted with an increased number of iterations during the online learning phase. The settings for this study are presented in Table 4.7. The results of this extended study can be found in Table 4.8.

Table 4.7 Setting of the simulation study during the online learning phase with applying decaying alpha in the presence of deterministic valuations and known distribution

Number of segments	n	4
Number of data points	T	1000
Number of iterations in online learning phase	n_{iter}	2000
Number of nearest neighbors	k	50
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Epsilon	ϵ	1
Alpha	α	[0.25, 0.50, 1.00]
Decay rate	$decay_{rate}$	[0.95, 0.99]
Decay threshold	$decay_{threshold}$	0.001
Decay step	$decay_{step}$	50
Moving average window size	$decay_{window}$	[50, 100]

Table 4.8 Simulation results for obtained average reward during a longer online learning phase (2000 iterations) with applying decaying alpha in the presence of deterministic valuations and known distribution

		$\delta_i = [0.1, 0.2, 0.3, 0.4]$								
		$\alpha = 0.25$			$\alpha = 0.50$			$\alpha = 1.00$		
		Average Reward	**Final Alpha**	**#Alpha**	**Average Reward**	**Final Alpha**	**#Alpha**	**Average Reward**	**Final Alpha**	**#Alpha**
$decay_{window} = 50$	$decay_{rate} = 0.95$	40.075	0.034	39.000	40.016	0.068	39.000	39.995	0.135	39.000
	$decay_{rate} = 0.99$	39.989	0.169	39.000	39.922	0.338	39.000	39.815	0.676	39.000
$decay_{window} = 100$	$decay_{rate} = 0.95$	40.058	0.036	38.000	39.899	0.071	38.000	39.977	0.142	38.000
	$decay_{rate} = 0.99$	40.110	0.171	38.000	39.983	0.341	38.000	39.676	0.683	38.000

The results indicate that with 2000 iterations of online learning, using $decay_{window} = 100$, $decay_{rate} = 0.99$, $decay_{threshold} = 0.001$, and an initial α value of 0.25, an average reward of 40.11 was achieved. This corresponds to 99% of the expected reward, demonstrating the algorithm's success in maximizing the seller's reward and its convergence towards the expected reward. These findings underscore the effectiveness of the algorithm in optimizing the reward and its ability to approach the desired target.

A further simulation study was conducted with an increased number of iterations during the online learning phase to demonstrate the convergence of the algorithm. The results of this study are presented in Table 4.9. After conducting the online learning for 5000 iterations, an average reward of 40.196 was obtained, which corresponds to 99.25% of the expected reward. Notably, this average reward is higher than the previously obtained results, confirming that the algorithm converges to the expected reward over the long run. This serves as strong evidence of the algorithm's ability to optimize rewards and achieve convergence towards the desired outcome.

Table 4.9 Simulation results for obtained average reward during a longer online learning phase (5000 iterations) with applying decaying alpha in the presence of deterministic valuations and known distribution

	$\delta_i = [0.1, 0.2, 0.3, 0.4]$		
	$\alpha = 0.25$		
	Average Reward	**Final Alpha**	**#Alpha**
$decay_{rate} = 0.99$	40.196	0.093	98.000
$decay_{window} = 100$			
$decay_{threshold} = 0.001$			

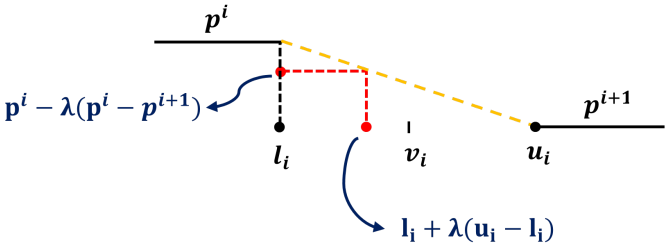
4.2 Probabilistic Valuations, non-overlapping intervals

In Section 3.1.7, the concept of probabilistic valuations was examined, focusing on the estimation of valuations through offline learning. In this section, our objective is to employ online learning in conjunction with probabilistic valuations, with the aim of exploring its impact on the collected reward during the online learning phase. Specifically, we investigate probabilistic valuations with non-overlapping intervals where the intervals within which valuations change are distinct intervals without overlapping with each other. It should be noted that probabilistic valuations introduce alterations to the conventional calculation of expected reward. In the forthcoming section, we will delve into the methodology of computing the expected reward in the context of probabilistic valuations.

4.2.1 Expected Reward in the presence of probabilistic valuations

In our previous discussions, we defined p^i as the purchase probability for cluster i and p^{i+1} as the purchase probability for the subsequent cluster, $i+1$, in the context of deterministic valuations. In the case of deterministic valuations, where we observe sharp declines from one cluster to the next, the purchase probabilities can be computed based on the offered prices and p^i values. However, for probabilistic valuations, these declines exhibit a gradual pattern, as illustrated in Figure 4.2.

Figure 4.2 Expected reward in the presence of probabilistic valuations


p^i
$\mathbf{p}^i - \lambda(\mathbf{p}^i - p^{i+1})$
p^{i+1}
l_i
v_i
u_i
$\mathbf{l_i + \lambda(u_i - l_i)}$

In the presence of probabilistic valuations, the valuation v_i changes for each customer within the segment i, falling within the range of (l_i, u_i). Consequently, it becomes evident that the purchase probability for offered prices lower than l_i is p^i, while the purchase probability for prices greater than u_i is p^{i+1}, as customers from segment i will be lost when faced with prices exceeding u_i. The question then arises: what is the purchase probability for prices offered between l_i and u_i?

To address this question, we assume a linear decrease in purchase probability between p^i and p^{i+1} for prices within the range of l_i and u_i, as depicted by the yellow dashed line in the Figure 4.2. To incorporate this linear characteristic, we introduce a parameter λ, where $\lambda \in (0,1)$, and represent the prices between the lower bound l_i and upper bound u_i as $l_i + \lambda \times (u_i - l_i)$ for segment i. By employing this price representation, the purchase probability for each of these price values can be expressed as $p^i - \lambda \times (p^i - p^{i+1})$. Consequently, the expected reward for each price within segment i can be calculated as $[l_i + \lambda \times (u_i - l_i)] \times [p^i - \lambda \times (p^i - p^{i+1})]$. Our aim is to maximize this expression, as it allows us to achieve the largest possible expected rewards. Expanding this expression results in a polynomial expression involving λ. By taking the derivative of this expression and equating it to zero, we can determine the value of λ_i^*, which is computed as follows:

$$\lambda_i^* = \frac{p^i(u_i - l_i) - l_i(p^i - p^{i+1})}{2(p^i - p^{i+1})(u_i - l_i)} \tag{4.6}$$

The value of λ_i^* provides us with the optimal parameter value, enabling us to compute the price and its corresponding purchase probability for segment i, thereby maximizing the reward within that specific segment. In order to calculate the expected reward in our problem, considering the presence of probabilistic valuations, we follow a three-step process. First, we determine the value of λ_i^* for each segment. Next, we calculate the reward for each segment based on the corresponding optimal parameter value. Finally, the maximum reward obtained across all segments represents the expected reward in our problem.

For the given case with $v_i = [20, 45, 55, 80]$, $\delta_i = [0.1, 0.2, 0.3, 0.4]$, and $p = 5\%$, the optimal values for λ are determined as follows: $\lambda^* = [0.25, 0, 0, 0]$. Subsequently, we can calculate the expected rewards for each segment. The expected rewards for the respective segments are as follows: $reward_{expected} = [19.0125, 38.4750, 36.5750, 30.4000]$. Hence, the expected reward for the overall problem is determined as 38.475.

4.2.2 Simulation

In order to assess the performance of the proposed Online Algorithm 1 and its proximity to the expected reward, a simulation study will be conducted in the presence of probabilistic valuations. The study will explore various settings, which are outlined in Table 4.10. By deploying the Online Algorithm 1 in these different settings, we aim to observe the achieved rewards and evaluate their proximity to the expected reward.

Table 4.10 Setting of the simulation study during the online learning phase with applying decaying alpha in the presence of probablistic valuations and known distribution

Number of segments	n	4
Number of data points	T	1000
Number of iterations in online learning phase	n_{iter}	1000
Number of nearest neighbors	k	50
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Epsilon	ϵ	1
Alpha	α	[0.0, 0.5]
Decay rate	$decay_{rate}$	[0.95, 0.99]
Decay threshold	$decay_{threshold}$	0.001
Decay step	$decay_{step}$	50
Moving average window size	$decay_{window}$	[50, 100]
Valuation percentage deviation	$p\%$	5%

Table 4.11 Simulation results for obtained average reward during online learning phase by applying decaying alpha in the presence of probabilistic valuations and known distribution

		$\delta_i = [0.1, 0.2, 0.3, 0.4]$									
		$\alpha = 0.0$					$\alpha = 0.5$				
		Average Reward	**Final Alpha**	**#Alpha**	**True interval Online% (avg)**	**True interval Offline% (avg)**	**Average Reward**	**Final Alpha**	**#Alpha**	**True interval Online% (avg)**	**True interval Offline% (avg)**
$decay_{window} = 50$	$decay_{rate} = 0.95$	35.749	0.000	0.000	0.963	0.840	36.862	0.189	19.000	0.938	0.760
	$decay_{rate} = 0.99$	35.749	0.000	0.000	0.963	0.840	37.107	0.413	19.000	0.885	0.810
$decay_{window} = 100$	$decay_{rate} = 0.95$	35.749	0.000	0.000	0.963	0.840	36.901	0.199	18.000	0.960	0.820
	$decay_{rate} = 0.99$	35.749	0.000	0.000	0.963	0.840	37.245	0.417	18.000	0.950	0.710

The simulation study results are presented in Table 4.11, which comprises 100 replications. In the first column, the median of the average rewards obtained during the online learning phase is displayed. The second column presents the final value of α, after multiple decays. The third column indicates the number of times α has been reduced throughout the study. Moving to the fourth column, it shows the percentage of time that the algorithm offers a point within the optimal interval during the online learning phase. The optimal interval refers to the range that yields the highest expected reward. In this specific case, the optimal interval is the second interval, denoted as [42.75, 47.25]. This metric assesses the algorithm's ability to make pricing decisions that align with the optimal range. Finally, the last column demonstrates the percentage of time that the estimated value of the valuation for the segment with the highest expected reward, as obtained from the offline learning phase, falls within the optimal interval. This metric evaluates the accuracy of the estimated valuation for the segment with the highest potential for reward.

The results indicate an increase in the percentage of time that the algorithm offers a price within the optimal interval during the online learning phase, which demonstrates an improvement in the estimation of valuations by the algorithm in this phase. The highest average reward achieved is 37.245, which corresponds to 96.8% of the expected reward (38.475). However, there is room for further improvement. In the current algorithm, the estimated valuations are represented by the middle points of the changing intervals. Yet, according to the derived formula 4.6, there may be another value within that range that yields the highest reward. Therefore, instead of offering the middle points, if we offer the value that maximizes the expected reward, we can potentially attain higher rewards during the online learning phase.

To accomplish this, we need to know the lower and upper bounds of the intervals within which the valuations are changing. In the upcoming section, we will develop methods to estimate these bounds. Subsequently, we will utilize the formula 4.6 to offer prices in a more structured manner during the online learning phase, enabling us to obtain higher rewards. This approach aims to refine the pricing strategy by incorporating valuation bounds.

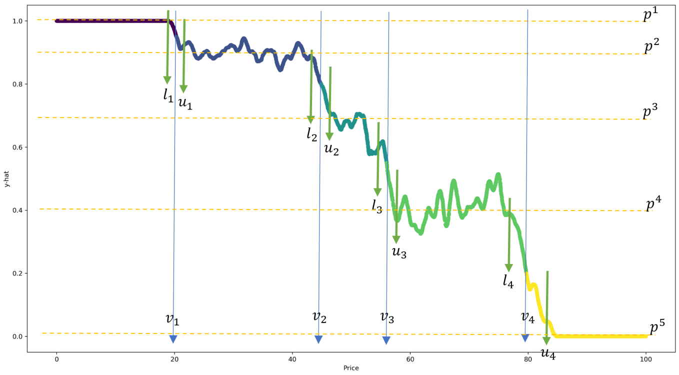
4.2.3 Proposed methods for estimating the lower and upper bounds of probabilistic valuations

In the context of probabilistic valuations, the valuation v_i for customers within segment i changes within the range $[l_i, u_i]$. In this section, our objective is to develop methods to estimate these bounds for each valuation. By accurately estimating the lower bound l_i and upper bound u_i, we can effectively capture the range of valuations for customers in segment i. These estimation methods will enable us to refine our understanding of customer valuations and inform the pricing strategy accordingly.

4.2.3.1 Method 1

Figure 4.3 showcases a distinct pattern where, as we approach the lower bound of interval i within which the valuation v_i changes probabilistically, there is a consistent trend up to l_i followed by a gradual decrease in the $\hat{\hat{y}}$ values of the prices up to a certain point, which represents the upper bound u_i of this interval. Beyond this point, we observe another steady trend in the $\hat{\hat{y}}$ values. The blue vertical lines indicate the middle points of the intervals or the valuations v_i, the green vertical lines show the corresponding lower and upper bounds of these valuations, and the orange horizontal lines represent the purchase probabilities p^i for clusters.

Figure 4.3 Shape of the expected purchase probability function in the presence of probabilistic valuations along with the lower and upper bounds of the valuations



Based on this observation, we propose a method to estimate the bounds of the intervals. Since each interval i comprises of two clusters i and $i+1$, We suggest starting from the middle point of interval i and calculating the differences between the purchase probabilities p^i and p^{i+1} and the $\hat{\hat{y}}$ values of the prices in this interval. As we move to the left of the middle point, these differences gradually decrease until they reach a certain point where they start to fluctuate. Similarly, as we move to the right of the middle point, the differences increase until they also reach a point of fluctuation. We believe these points of decrease and increase in the differences reflect the lower and upper bounds of the intervals, respectively. These points capture the underlying behavior of the probabilistic valuations.

Therefore, in our first method by analyzing the behavior of the differences between purchase probabilities p^i and $\hat{\hat{y}}$ values, we can identify the points at which these differences start to fluctuate, indicating the lower and upper bounds of the intervals. This approach aims to leverage the observed patterns to estimate the valuation bounds more accurately.

The proposed method for estimating the bounds is as follows:

- Identify the clusters and estimate the valuations for each segment based on the given dataset and proposed Algorithm 1.

- For each estimated valuation $\hat{v}_i$ within segment i:
  - Calculate two difference vectors as:
    - $differences_{decreasing} = p^i - \hat{\hat{y}}$ for data points in cluster i.
    - $differences_{increasing} = p^{i+1} - \hat{\hat{y}}$ for data points in cluster $i+1$.
  - Begin from the estimated valuation $\hat{v}_i$ and move towards the left. Compare the consecutive differences $differences_{decreasing}$ and verify that they are consistently decreasing. If at any point these differences cease to decrease, identify that point as the estimated lower bound $\hat{l}_i$ for valuation v_i.
  - Similarly, start from the estimated valuation $\hat{v}_i$ and move towards the right. Compare the consecutive differences $differences_{increasing}$ and ensure that they are consistently increasing. If these differences stop increasing at any point, recognize that point as the estimated upper bound $\hat{u}_i$ for valuation v_i.

In the process of comparing consecutive differences ($differences_j$ and $differences_{j+1}$) to estimate the upper and lower bounds, there may be instances where certain points exhibit fluctuations or outliers that do not align with the overall trend. To mitigate the influence of these points, we introduce a hyperparameter called $difference_{factor}$. The $difference_{factor}$ is multiplied by the first difference ($differences_j$) when conducting the comparison. By incorporating this factor, we aim to ensure that the subsequent differences ($differences_{j+1}$) are sufficiently larger or smaller than the initial difference, thus helping us identify the correct points corresponding to the upper and lower bounds. The introduction of the $difference_{factor}$ serves as a mechanism to filter out potential outliers or fluctuations, enabling us to focus on the points that reflect the underlying behavior of the valuation intervals more accurately. It adds an additional level of control and robustness to the estimation process.

4.2.3.2 Method 2

Our second method for estimating the lower and upper bounds involves examining the largest and smallest prices within clusters that are close to the corresponding purchase probabilities. We can express this method algorithmically as follows:

- Identify the clusters and estimate the valuations for each segment based on the given dataset and proposed Algorithm 1.
- For each estimated valuation $\hat{v}_i$ within segment i:
  - Estimate the lower bound $\hat{l}_i$ by finding the largest price within cluster i that is in close proximity to the corresponding purchase probability p^i.
  - Find the smallest price within cluster $i+1$ that is in close proximity to the purchase probability p^{i+1}, and assign it as the estimatedupper bound $\hat{u}_i$ for the valuation v_i.

To Estimate the lower and upper bounds accurately, we consider the differences between the buying probabilities and the $\hat{\hat{y}}$ values within each cluster i as $|p^i - \hat{\hat{y}}|$. To determine the prices that are close to the purchase probabilities, we introduce a hyperparameter called $price_{threshold}$. This hyperparameter serves as a measurement of the closeness between the $\hat{\hat{y}}$ values of the prices and the buying probabilities p^i in cluster i. Therefore, the largest and smallest prices in cluster i that satisfy the following inequality will be estimated as the lower and upper bounds of valuaiton v_i:

$$|p^i - \hat{\hat{y}}| \leq price_{threshold} \tag{4.7}$$

This method allows us to dynamically estimate the lower and upper bounds by considering the prices within the clusters that align closely with the purchase probabilities. By utilizing these specific prices, we can effectively define the boundaries of the valuation intervals for each segment.

4.2.3.3 Method 3

In addition to the previous two methods, we introduce a hybrid method that combines elements from both methods to achieve a balanced estimation of the bounds. This hybrid method involves taking the average of the estimated bounds obtained from methods 1 and 2. The steps for the hybrid method are as follows:

- Apply method 1: Estimate the lower and upper bounds for each valuation v_i as

$\hat{l}_i^{method1}$ and $\hat{u}_i^{method1}$ by comparing the consecutive differences and identifying the points where the differences stop decreasing or increasing, respectively.

- Apply method 2: Estimate the lower and upper bounds for each valuation v_i as $\hat{l}_i^{method2}$ and $\hat{u}_i^{method2}$ by examining the largest and smallest prices within clusters close to the corresponding purchase probabilities.
- Calculate the average of the lower bounds obtained from methods 1 and 2 to determine the hybrid lower bound for each valuation: $\hat{l}_i^{method3} = (\hat{l}_i^{method1} + \hat{l}_i^{method2})/2$
- Calculate the average of the upper bounds obtained from methods 1 and 2 to determine the hybrid upper bound for each valuation: $\hat{u}_i^{method3} = (\hat{u}_i^{method1} + \hat{u}_i^{method2})/2$.

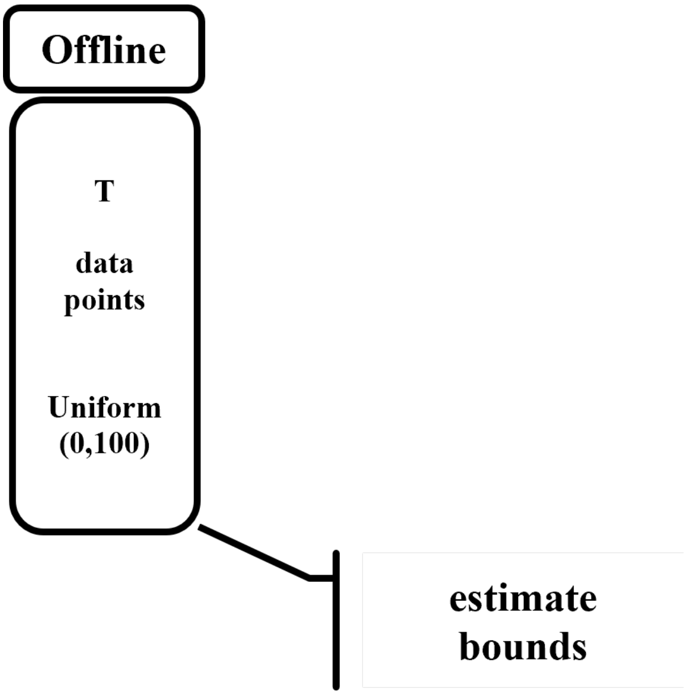
By averaging the bounds obtained from both methods, the hybrid method aims to strike a balance and leverage the strengths of each approach. This can result in a more robust and accurate estimation of the lower and upper bounds, considering the insights gained from both methods simultaneously.

4.2.4 Estimating the lower and upper bounds of probabilistic valuations by offline learning

Having developed methods for estimating the lower and upper bounds, we are now equipped to apply these methods to various datasets for learning the bounds.

In a similar manner to the offline learning approach for estimating the parameters of the problem, we can also apply the developed methods for estimating the bounds in offline mode. In this algorithm, denoted as "Offline Bounds Estimation" we assume that a complete dataset of size T is provided, generated uniformly within the range of $(0, 100)$. We subsequently employ the proposed methods on the entire dataset to estimate the bounds.

Figure 4.4 Estimating the lower and upper bounds of probabilistic valuations by Offline Bounds Estimation



4.2.5 Simulation

A simulation study was conducted to evaluate the efficacy of three developed methods in estimating the lower and upper bounds of probabilistic valuations in an offline mode, where a complete dataset of size T was provided. The experimental settings are summarized in Table 4.12. Various dataset sizes were considered to investigate the impact on estimation accuracy. Hyperparameter k was selected proportionally to the dataset size. The identification and elimination of outliers were accomplished using the IQR method. Additionally, two different values for valuation percentage deviation ($p\%$) were examined to assess the methods' performance across wider intervals. Each experimental setting was replicated 200 times.

Table 4.12 Setting of the simulation study for estimating the lower and upper bounds of probabilistic valuations by applying the three developed methods and offline bounds estimation

Number of segments	n	4
Number of data points	T	[1000, 5000, 10000]
Number of nearest neighbors	k	[50, 75, 100]
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Non-uniform distribution	δ_i	[0.1, 0.2, 0.3, 0.4]
Difference factor for comparing the consecutive differences	$difference_{factor}$	[1.05, 1.1]
Threshold for measuring the closeness of prices to the purchase probabilities	$price_{threshold}$	0.01
Valuation percentage deviation	$p\%$	[5%, 10%]

Table 4.13 simulation results for estimated lower and upper bounds of probabilistic valuations by applying the three developed methods and offline bounds estimation

		$p\% = 5, difference_{factor} = 1.05$								
		l_1	u_1	l_2	u_2	l_3	u_3	l_4	u_4	Error
		19	21	42.75	47.25	52.25	57.75	76	84	
		$\hat{l}_1$	$\hat{u}_1$	$\hat{l}_2$	$\hat{u}_2$	$\hat{l}_3$	$\hat{u}_3$	$\hat{l}_4$	$\hat{u}_4$	
$T = 1000, k = 50$	Method 1	15.771	23.719	41.106	48.746	50.505	59.222	74.165	87.602	2.218
	Method 2	17.242	23.296	41.674	48.795	50.412	59.770	73.251	85.775	1.882
	Method 3	16.434	23.645	41.349	49.019	50.300	59.749	73.198	86.764	2.237
$T = 5000, k = 75$	Method 1	18.204	21.124	42.999	46.839	52.563	57.490	76.396	85.085	0.454
	Method 2	18.896	20.980	42.989	46.994	52.323	57.639	76.288	84.113	0.150
	Method 3	18.542	21.089	42.948	46.993	52.442	57.625	76.310	84.628	0.282
$T = 10000, k = 100$	Method 1	18.520	21.068	43.040	46.879	52.631	57.256	76.567	84.583	0.404
	Method 2	19.106	20.910	43.024	47.016	52.459	57.397	76.444	83.899	0.226
	Method 3	18.813	21.059	43.017	46.950	52.557	57.354	76.463	84.266	0.281
		$p\% = 10, difference_{factor} = 1.1$								
		l_1	u_1	l_2	u_2	l_3	u_3	l_4	u_4	Error
		18	22	40.5	49.5	49.5	60.5	72	88	
		$\hat{l}_1$	$\hat{u}_1$	$\hat{l}_2$	$\hat{u}_2$	$\hat{l}_3$	$\hat{u}_3$	$\hat{l}_4$	$\hat{u}_4$	
$T = 1000, k = 50$	Method 1	15.368	24.157	39.679	49.157	49.227	61.316	71.230	91.090	1.363
	Method 2	17.018	23.283	40.212	48.870	49.557	61.399	71.546	88.950	0.693
	Method 3	16.139	23.804	39.884	49.034	49.463	61.592	71.367	90.035	1.068
$T = 5000, k = 75$	Method 1	17.510	21.904	41.461	48.516	50.617	59.638	73.422	88.909	0.855
	Method 2	18.320	21.612	41.357	48.746	50.408	59.972	72.815	87.706	0.608
	Method 3	17.933	21.761	41.518	48.651	50.525	59.690	73.210	88.351	0.696
$T = 10000, k = 100$	Method 1	17.821	21.773	41.458	48.341	50.741	59.304	73.135	88.474	0.821
	Method 2	18.417	21.541	41.446	48.405	50.466	59.475	72.938	87.546	0.788
	Method 3	18.133	21.682	41.469	48.293	50.656	59.409	73.069	88.023	0.746

Table 4.13 presents the median values for the estimated lower and upper bounds

obtained from the three developed methods by applying offline bounds estimation, accompanied by the corresponding estimation errors. The reported error is computed as the mean absolute error between the estimated and true values of the lower and upper bounds, using the following formula:

$$Error = \frac{1}{n}\sum_{i=1}^{n}(|\hat{l}_i - l_i| + |\hat{u}_i - u_i|) \tag{4.8}$$

Here, $\hat{l}_i$ and $\hat{u}_i$ denote the estimated lower and upper bounds of valuation i, while l_i and u_i represent the true values of the lower and upper bounds of the same valuation i. The error is averaged across n valuations.

The results demonstrate the estimation error decreases as the dataset size increases. This observation implies that the methods employed exhibit greater robustness in accurately estimating the lower and upper bounds when a larger number of data points are available. Specifically, when considering datasets comprising 5000 and 10000 data points, the estimation error remains below 1 unit for both scenarios involving $p = 5\%$ and $p = 10\%$. This outcome highlights the efficacy of the developed methods in successfully estimating the bounds.

The results obtained from the various offline and online learning methods demonstrate their effectiveness in successfully estimating the bounds, particularly when dealing with relatively large datasets. With accurate estimates of the bounds, we can now proceed to the next phase of the online learning process, which involves offering prices in a more rigorous manner to maximize the collected reward. To facilitate this, a new algorithm has been developed, which will be discussed in the upcoming section. This algorithm aims to leverage the estimation of bounds to enhance the pricing strategy, ultimately leading to increased reward collection.

4.2.6 Proposed Online Algorithm 2

In Online Algorithm 1, when dealing with probabilistic valuations, we estimate the valuations using conventional methods and consider them as the middle points of the intervals where the valuations change. These estimated valuations are then offered to the customers. However, through analysis of the expected reward in the presence of probabilistic valuations, we have observed that offering a price other than the middle point within the changing interval may lead to higher rewards. To determine the optimal price within each interval, we need to know the lower and upper bounds of these intervals. With the accurate estimation methods we have developed for these bounds, we can leverage Equation 4.6 to identify the price that maximizes the reward based on the purchase probabilities. Therefore, instead of offering the middle points, in Online Algorithm 2, we propose offering the price that brings the highest reward at each iteration of the online learning process. This approach aims to increase the collected reward throughout the online learning phase, moving closer to the expected reward. The Online Algorithm 2 is outlined as follows:

- Provided with dataset of size T, estimate the lower and upper bounds of valuations in offline mode by Offline Bounds Estimation $[\hat{l}_i^{initial}, \hat{u}_i^{initial}]$
- $[\hat{l}_i, \hat{u}_i] = [\hat{l}_i^{initial}, \hat{u}_i^{initial}]$
- For n_{iter} iterations:
  - Offer price x to the incoming customer
    - Find the optimal interval
    - With probability ϵ offer the price in the optimal interval that yields the highest reward
    - With probability $1-\epsilon$ offer a price uniformly distributed in the optimal interval
  - Getting the customers' feedback y
  - Add the new data point (x, y) to the existing dataset
  - Re-calculate the $\hat{y}$ values of the data points in the converted dataset
  - Re-cluster
  - Re-estimate the lower and upper bounds and replace the $[\hat{l}_i, \hat{u}_i]$'s with the new estimates

The proposed Online Algorithm 2 consists of both offline and online learning phases. In the offline learning part, where we are provided with a dataset of size T, we estimate the lower and upper bounds using Offline Bounds Estimation with the hybrid method to obtain initial estimates of the bounds $[\hat{l}_i^{initial}, \hat{u}_i^{initial}]$.

Moving on to the online learning phase, we aim to offer prices to customers in a way that maximizes the collected reward. At each iteration of online learning, the first step is to identify the optimal interval, which is the interval that contains the price with the highest reward. To determine the optimal interval, we calculate the $\hat{\lambda}_i^*$ value for each estimated interval $[\hat{l}_i, \hat{u}_i]$ using the following equation:

$$\hat{\lambda}_i^* = \frac{p^i(\hat{u}_i - \hat{l}_i) - \hat{l}_i(p^i - p^{i+1})}{2(p^i - p^{i+1})(\hat{u}_i - \hat{l}_i)} \tag{4.9}$$

By utilizing the estimated lower and upper bounds along with the calculated $\hat{\lambda}_i^*$ values, we can find the estimated optimal price $\hat{x}^i$ that maximizes the reward in interval i as:

$$\hat{x}^i = \hat{l}_i + \hat{\lambda}_i^* \times (\hat{u}_i - \hat{l}_i) \tag{4.10}$$

Once the estimated optimal price $\hat{x}^i$ is determined, we calculate the purchase probability $\hat{p}^i$ for that price as:

$$\hat{p}^i = p^i - \hat{\lambda}_i^* \times (p^i - p^{i+1}) \tag{4.11}$$

Subsequently, we calculate the expected reward for each interval as:

$$reward^i_{expected} = \hat{x}^i \times \hat{p}^i \tag{4.12}$$

The interval with the highest expected reward is then identified as the estimated optimal interval, denoted by $[\hat{l}_{optimal}, \hat{u}_{optimal}]$.

For offering prices to customers, we employ the epsilon-greedy method. With a probability of ϵ, we offer the price in the estimated optimal interval that brings the highest reward (already calculated for each interval). With a probability of $1-\epsilon$, we offer a price uniformly distributed within the estimated optimal interval $[\hat{l}_{optimal}, \hat{u}_{optimal}]$.

The offered price (x) is presented to the customer, and their feedback (y) is collected. This new data point (x, y) is incorporated into the existing dataset, and the $\hat{\hat{y}}$ values for each element in the dataset are recalculated, resulting in the creation of a new converted dataset. The data points are then re-clustered using the method described in Algorithm 2. Finally, based on the identified clusters, the lower and upper bounds are re-estimated using Offline Bounds Estimation with the hybrid method and replaced with $[\hat{l}_i, \hat{u}_i]$ values. This iterative process is repeated for n_{iter} iterations, allowing for continuous improvement and refinement of the pricing strategy based on customer feedback and the estimated bounds.

4.2.7 Simulation

A simulation study was conducted to assess the performance of Online Algorithm 2 in collecting rewards during the online learning phase with probabilistic valuations. The study considered different settings, including the initial value of k, which was set to 125, equivalent to 5% of the dataset size used in the offline learning phase. During the online learning phase, the value of k was adjusted proportionally to the increasing dataset size. Additionally, the effect of different ϵ values was examined to analyze their impact on the collected rewards. Additionally, the online learning phase was performed for a longer period to observe the convergence of the algorithm toward the expected reward. The detailed settings of this study are presented in Table 4.14.

Table 4.14 Setting of the simulation study for evaluating the Online Algorithm 2 in the presence of probabilistic valuations and known distribution

Number of segments	n	4
Number of data points	T	2500
Number of iterations in online learning phase	n_{iter}	[1000, 2500]
Number of nearest neighbors	k	[125,...,175]
Number of data conversion	$n_{conversion}$	2
Non-uniform valuations	v_i	[20, 45, 55, 80]
Non-uniform distribution	δ_i	[0.1, 0.2, 0.3, 0.4]
Epsilon	ϵ	[0.5, 0.7, 0.9, 1.0]
Difference factor for comparing the consecutive differences	$difference_{factor}$	1.05
Threshold for measuring the closeness of prices to the purchase probabilities	$price_{threshold}$	0.01
Valuation percentage deviation	$p\%$	5%

The results from the simulation study, as presented in Table 4.15, show the median of average rewards for different values of ϵ and n_{iter} based on 100 replications. It is observed that as the value of ϵ increases, the average reward collected during the online learning phase also increases. This is in line with expectations, as offering prices with higher rewards more frequently leads to higher average rewards. The highest reward achieved with 1000 iterations was 37.586, corresponding to 97.69% of the expected reward. Importantly, this outperforms the reward obtained by applying Online Learning 1, which was 37.245.

To conduct a longer online learning phase, the ϵ value is set to 1, as it yielded the highest average reward. With 2500 iterations, the average reward achieved was 37.809, corresponding to 98.27% of the expected reward. This demonstrates the convergence of the algorithm towards the expected reward. These findings highlight the effectiveness of estimating the bounds and offering prices based on the highest reward in achieving greater rewards during the online learning phase compared to offering middle points.

Table 4.15 Average rewards obtained by Online Algorithm 2 in the presence of probabilistic valuations and known distribution

		Average Reward	Expected Reward
$n_{iter} = 1000$	$\epsilon = 0.5$	35.883	38.475
	$\epsilon = 0.7$	36.705	
	$\epsilon = 0.9$	37.279	
	$\epsilon = 1.0$	37.586	
$n_{iter} = 2500$	$\epsilon = 1.0$	37.809	

4.2.8 Applying the Alpha hyperparameter in the presence of probabilistic valuations

In the context of deterministic valuations, it has been observed that incorporating a deduction of the α value from our estimated price leads to the attainment of higher rewards. This deduction helps mitigate the impact of overshooting the valuation estimates. However, when addressing probabilistic valuations, the nature of customers' valuations is inherently random and undergoes changes upon the arrival of each customer. Consequently, the deduction of the α value does not ensure that the offered price will fall below the customer's valuation. Nevertheless, in order to examine the impact of this hyperparameter on the average reward collected when dealing with probabilistic valuations, a simulation study was conducted. This study employed the same experimental setup as the previous simulation, encompassing

1000 online iterations, an ϵ value of 1, and varied values of α. The corresponding results are presented in Table 4.16.

Table 4.16 Average rewards obtained by Online Algorithm 2 by applying the hyperparameter Alpha in the presence of probabilistic valuations and known distribution

	Average Reward	Expected Reward
$\alpha = 0.00$	37.586	38.475
$\alpha = 0.25$	37.148	
$\alpha = 0.50$	36.693	

The results from the simulation study, as presented in Table 4.16, show the median of average rewards for different values of α based on 100 replications.

The results indicate that utilizing a non-zero value for α leads to diminished rewards. This outcome can be attributed to two primary reasons. Firstly, in this model, the objective is to identify the optimal price and offer it to customers. Deducting the α value results in offering lower prices, consequently yielding reduced rewards. Secondly, given the stochastic nature of customers' valuations, deducting α leads to offering prices that are occasionally lower than their valuations, but not consistently. Consequently, it can be inferred that the utilization of this hyperparameter in the presence of probabilistic valuations is not advantageous and results in decreased rewards.

In the previous algorithms developed, the process of converting the preference dataset involved calculating the average buying preferences, resulting in $\hat{y}$ or $\hat{\hat{y}}$ values. These values served as the basis for determining the offered prices during the online learning phase. However, we aim to introduce the concept of expected reward for prices and develop a new algorithm to offer prices to incoming customers based on the expected rewards associated with each price.

Let's consider the preference dataset provided as $\{(x^t, y^t)\}$, where x^t represents the price offered by the seller and y^t denotes the customer's feedback (1 for buying and 0 for not buying). To calculate the reward for each price x^t, we can compute $r^t = x^t * y^t$. If the customer buys the product ($y^t = 1$) the reward achieved by this purchase is as much as the offered price x^t ($r^t = x^t$), and if the customer does not buy the product ($y^t = 0$), there is no reward for the seller ($r^t = 0$). This allows us to create a new dataset, referred to as the reward dataset, in the form of $\{(x^t, y^t, r^t)\}$. Consequently, in order to offer prices to customers based on the rewards these prices bring, we require a conversion process.

In this new conversion process, instead of averaging the y values of the k nearest data points for each price x^t, we calculate the average of the r values of the k nearest data points for each price x^t. This provides us with the expected reward for each price ($\hat{r}^t$), effectively converting our dataset into $\{(x^t, y^t, \hat{r}^t)\}$ format. Through this conversion, we gain insight into which prices are more likely to yield higher rewards based on their expected values. Consequently, we can offer these prices to customers during the online learning phase to maximize the rewards achieved. In this section, we focus on probabilistic valuations characterized by non-overlapping distinct intervals within which these valuations change for incoming customers.

4.2.9 Proposed Reward Algorithm 1

Our proposed algorithm for offering prices to customers based on the expected rewards of the prices under the presence of probabilistic valuations and non-overlapping intervals is as follows:

- **Offline phase**
  - Provided with the preference dataset $\{(x^t, y^t)\}$ of size T, derive the reward dataset $\{(x^t, y^t, r^t)\}$ and convert it to obtain the converted dataset $\{(x^t, y^t, \hat{r}^t)\}$.
  - Find the optimal interval $[\hat{l}_{optimal}, \hat{u}_{optimal}]$ by estimating the lower and upper bounds of valuations via Offline Bounds Estimation and the hybrid method
  - Set the initial value for ϵ : $\epsilon = \epsilon_0$
- **Online phase**
  - For n_{iter} iterations:
    - Offer price x to the incoming customer
      - With probability ϵ offer the price in the optimal interval with the highest expected reward $\hat{r}$
      - With probability $1 - \epsilon$ offer a price uniformly distributed in the optimal interval
    - Getting the customers' feedback y

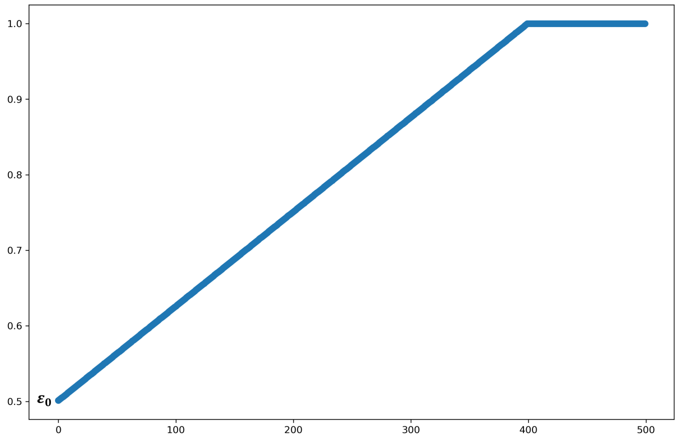
- Cacluate the reward r for the new data point (x,y)
- Add the new data point (x,y,r) to the existing reward dataset
- Re-calculate the $\hat{r}$ values of the data points in the converted dataset
- Increase the ϵ value

In the offline phase, we are provided with a preference dataset $\{(x^t,y^t)\}$ of size T. From this dataset, we derive the reward dataset $\{(x^t,y^t,r^t)\}$ by calculating the reward r^t for each price x^t. This reward dataset is converted using the KNN method as described earlier. The resulting converted dataset takes the form $\{(x^t,y^t,\hat{r}^t)\}$. These datasets will be utilized in the online learning phase for price offerings and updates. To offer prices in the online learning phase, it is crucial to determine the optimal interval where the price with the highest expected reward resides. This interval can be estimated based on the provided preference dataset using the same procedure as Online Algorithm 2. By following the steps outlined in that algorithm, we can identify the estimated optimal interval $[\hat{l}_{optimal},\hat{u}_{optimal}]$, which will be employed in the online learning phase.

The subsequent step in the offline phase is to set the initial value for ϵ. Similar to the previous online algorithms, this algorithm employs the epsilon-greedy technique. However, there is a difference in this algorithm: the ϵ value is not fixed throughout the online learning phase but increases as the learning progresses. The rationale behind using an increasing ϵ value is to ensure convergence to the price with the highest expected reward within the optimal interval during the online learning phase. Without such convergence, the algorithm may offer prices that do not maximize the reward, resulting in the average collected reward deviating from the expected reward. By starting with an initial ϵ value less than 1, the algorithm can explore the optimal interval, refine its estimates of expected rewards, and converge towards the price with the highest expected reward. Subsequently, by gradually increasing the ϵ value to 1 and maintaining it for a certain number of iterations while offering the price with the highest expected reward, the algorithm can exploit that price and attain more rewards, thereby increasing the average collected reward. In order to implement this increasing ϵ approach, we have considered two methods:

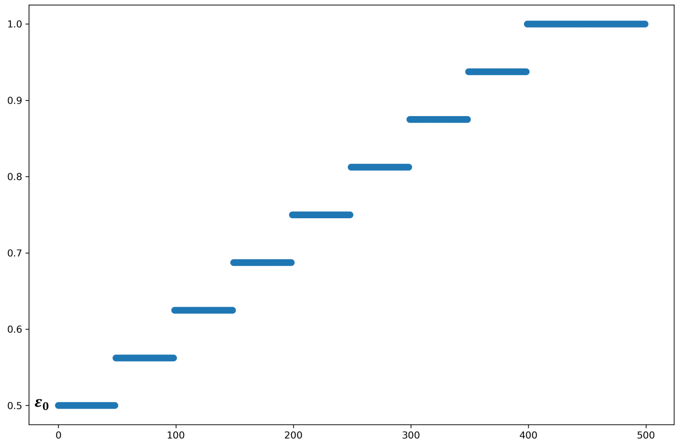
The first method involves initializing ϵ with an initial value (ϵ_0) and continuously increasing it during a specific number of iterations in the online learning phase until it reaches a value of 1. After reaching the value of 1, ϵ remains fixed for the remaining online learning iterations, allowing for the exploitation of the price with the highest expected reward.

Figure 4.5 Increasing Epsilon continuously



In the second method, we also begin with an initial value of ϵ (ϵ_0). However, instead of continuously increasing its value, we fix it for a certain number of iterations and then increment it by a specific value. We repeat this process, maintaining the new value of ϵ for a set number of iterations before increasing it again. This pattern continues until the ϵ value reaches 1. For the remaining duration of the online learning phase, we exploit the price with the highest expected reward.

Figure 4.6 Increasing Epsilon periodically



After establishing a fixed initial value for ϵ, we move to the online learning phase. Within this phase, we offer prices derived from the optimal interval obtained during the offline phase. With a probability of ϵ, we offer the price with the highest expected reward $\hat{r}$, while with a probability of $1 - \epsilon$, we offer a price uniformly distributed within the optimal interval. Within this algorithm, it is necessary to offer prices within the estimated optimal interval. However, it is possible that these estimates do not encompass the prices with the highest rewards. Consequently, by offering prices within such interval, it may be challenging to achieve high rewards. To mitigate this issue, we introduce a new hyperparameter called the *scale*. This scale value is used to widen the interval within which prices will be offered. For instance, if the value of this hyperparameter is s, and the estimated lower and upper bounds of the optimal interval are $\hat{l}_{optimal}$ and $\hat{u}_{optimal}$, prices will be offered in the range of $(\hat{l}_{optimal}(1-s)),(\hat{u}_{optimal}(1+s))$.

Upon receiving feedback (y) from incoming customers, we calculate the corresponding reward (r). Subsequently, we incorporate this new data point (x,y,r) into the existing reward dataset and recompute the $\hat{r}$ values for the prices within the converted dataset. This iterative process is repeated for a predetermined number of iterations (n_{iter}).

4.2.10 Simulation

To assess the effectiveness of the proposed Reward Algorithm 1 in achieving higher rewards, a simulation study was conducted. Similar to previous online algorithms, the value of k in this algorithm is proportional to the size of the dataset and increases during the online learning phase. Specifically, we use 5% of the dataset size as the value for k. In the offline phase, where we are provided with the preference dataset, it undergoes two conversions to determine the estimated optimal interval. The IQR method is utilized to identify and eliminate outliers from the dataset during this process. However, when converting the reward dataset to calculate the expected rewards, we perform the conversion only once and do not consider outliers. The complete setting of this simulation study can be found in Table 4.17, which outlines the specific settings and hyperparameters used for the evaluation. Each setting undergoes 100 replications of the simulation.

Table 4.17 Setting of the simulation study for evaluating the Reward Algorithm 1 in the presence of probabilistic valuations, non-overlapping intervals, and known distribution

Number of segments	n	4
Number of data points	T	2500
Number of iterations in online learning phase	n_{iter}	5000
Number of nearest neighbors	k	[125,...,375]
Number of data conversion	$n_{conversion}$	[1, 2]
Non-uniform valuations	v_i	[20, 45, 55, 80]
Non-uniform distribution	δ_i	[0.1, 0.2, 0.3, 0.4]
Initial epsilon	ϵ_0	[0.00, 0.25, 0.50, 0.75, 1.00]
Difference factor for comparing the consecutive differences	$difference_{factor}$	1.05
Threshold for measuring the closeness of prices to the purchase probabilities	$price_{threshold}$	0.01
Scale factor for widening the optimal interval	$scale$	0.05
Valuation percentage deviation	$p\%$	5%

The results of the simulation study are presented in Table 4.18, which includes the median of average rewards collected during the online learning phase, as well as the median of average rewards collected during the last 1000 iterations of the online learning phase, for different initial epsilon (ϵ_0) values. We used 80% of the online learning phase (4000 iterations) for increasing the ϵ value to explore the optimal interval, and in the final 20% of the online learning (1000 iterations), its value was fixed to 1 for exploiting the prices with the highest rewards. It is observed that increasing the initial epsilon value leads to higher average rewards during the online learning phase. There is a relatively small difference in the rewards obtained using the two different adaptive epsilon methods. However, when considering the average rewards during the last 1000 iterations of the online learning phase, an interesting pattern emerges. Initially, the average reward increases as epsilon increases. However, there comes a point where further increases in epsilon lead to a decrease in the average reward. This indicates that full exploitation ($\epsilon = 1$) does not always result in the highest reward. Instead, starting with an epsilon value less than 1 allows for exploration, and gradually increasing epsilon allows the algorithm to converge towards the optimal price, leading to higher rewards. The highest average reward obtained during the last 1000 iterations is 38.117, achieved with an initial epsilon value of 0.75 ($\epsilon_0 = 0.75$). This corresponds to 99.07% of the expected reward

(38.475). These results demonstrate the success of the algorithm in estimating prices with high rewards and increasing the revenue for the seller.

Table 4.18 Average rewards obtained by Reward Algorithm 1 in the presence of probabilistic valuations, non-overlapping intervals, and known distribution

	Continous Epsilon		**Periodic Epslion**	
	Average Reward	**Average Reward during last 1000 iteration**	**Average Reward**	**Average Reward during last 1000 iteration**
$\epsilon_0 = 0.00$	36.964	37.895	36.949	38.046
$\epsilon_0 = 0.25$	37.246	37.868	37.231	38.072
$\epsilon_0 = 0.50$	37.487	38.065	37.458	38.080
$\epsilon_0 = 0.75$	37.730	38.117	37.693	37.965
$\epsilon_0 = 1.00$	37.861	37.738	37.975	37.988

4.3 Probabilistic Valuations, overlapping intervals

In the previous section, we examined the scenario where customers' valuations exhibited distinct intervals without any overlap. In such cases, accurate estimation of bounds, identification of the optimal interval, and offering prices within that interval based on their expected rewards proved successful. Consequently, we achieved high average rewards that closely aligned with the expected reward. However, in this section, our objective is to investigate probabilistic valuations characterized by intervals that are no longer distinct, but rather exhibit points of overlap. This situation arises for instance when customers' valuations, such as $[20, 45, 55, 80]$, deviate within an interval of 15%. Consequently, the second interval ($[38.25, 51.75]$) and the third interval ($[46.75, 63.25]$) overlap with each other. Within this context, our focus is to assess the performance of the developed methods for estimating bounds and to determine an effective approach for offering prices that maximizes the achieved reward.

To assess the effectiveness of Method 3 (Hybrid Method) in estimating the lower and upper bounds of probabilistic valuations $v_i = [20, 45, 55, 80]$ with a deviation of 15% ($p\% = 15\%$), we conducted a simulation study with various settings. The study involved 100 replications, and the obtained results are presented in Table 4.19.

Table 4.19 The estimated lower and upper bound of probabilistic valuations where the intervals overlap

			$\delta_i = [0.1, 0.2, 0.3, 0.4]$								
			l_1	u_1	l_2	u_2	l_3	u_3	l_4	u_4	$\hat{l}_3 - \hat{u}_2$
			17	23	38.25	51.75	46.75	63.25	68	92	
			$\hat{l}_1$	$\hat{u}_1$	$\hat{l}_2$	$\hat{u}_2$	$\hat{l}_3$	$\hat{u}_3$	$\hat{l}_4$	$\hat{u}_4$	
$T = 2500$	$k = 62$	$difference_{factor} = 1.05$	17.207	22.942	40.290	48.248	50.272	60.948	70.996	91.642	1.885
		$difference_{factor} = 1.10$	16.953	22.584	39.535	48.595	49.520	61.749	69.927	92.008	0.119
		$difference_{factor} = 1.15$	16.822	22.878	39.082	48.801	49.554	61.936	69.333	92.018	0.132
	$k = 83$	$difference_{factor} = 1.05$	16.378	23.746	39.036	48.688	49.413	61.955	68.681	92.677	0.191
		$difference_{factor} = 1.10$	16.517	23.538	38.609	48.821	49.176	62.598	69.275	92.479	0.147
		$difference_{factor} = 1.15$	16.364	23.213	38.430	49.010	49.255	63.146	68.263	92.550	0.117
	$k = 125$	$difference_{factor} = 1.05$	15.085	23.980	37.895	48.879	49.089	64.514	67.379	93.782	0.183
		$difference_{factor} = 1.10$	15.249	24.093	37.133	48.886	49.062	63.543	67.363	93.972	0.204
		$difference_{factor} = 1.15$	14.990	24.789	37.441	48.786	48.955	64.122	67.807	93.925	0.182
$T = 5000$	$k = 125$	$difference_{factor} = 1.05$	16.595	22.624	39.060	48.713	49.128	62.586	69.155	92.444	0.183
		$difference_{factor} = 1.10$	16.613	22.930	38.756	48.865	49.148	62.304	68.497	92.420	0.163
		$difference_{factor} = 1.15$	16.591	23.154	38.612	49.073	49.334	62.719	68.506	92.444	0.163
	$k = 166$	$difference_{factor} = 1.05$	16.056	23.312	38.392	48.903	49.174	63.088	67.904	93.003	0.206
		$difference_{factor} = 1.10$	16.035	23.724	38.137	48.746	49.017	63.409	67.817	92.826	0.188
		$difference_{factor} = 1.15$	16.000	23.669	38.212	48.796	49.053	63.109	67.848	93.028	0.215
	$k = 250$	$difference_{factor} = 1.05$	14.944	24.359	37.483	48.905	49.197	63.786	67.059	94.080	0.263
		$difference_{factor} = 1.10$	14.878	24.529	36.787	48.827	49.093	64.518	67.103	94.219	0.250
		$difference_{factor} = 1.15$	14.914	25.103	36.774	48.855	49.079	65.051	66.591	94.151	0.246

The reported values in Table 4.19 represent the medians of the estimated bounds obtained under different settings. In this particular case, where the second and third intervals overlap, we also provide the difference between the lower bound of the third interval and the upper bound of the second interval $\hat{l}_3 - \hat{u}_2$.

Upon examination, it becomes apparent that Method 3 (hybrid method) demonstrates success in estimating the majority of lower and upper bounds, with the exception of u_2 and l_3. For these specific bounds, the estimated values closely approximate each other, as evidenced by the small difference $(\hat{l}_3 - \hat{u}_2)$. In fact, these estimates correspond to the midpoints between the true values of u_2 and l_3. The presence of such minimal differences indicates the detection of overlapping intervals. Leveraging this observation, we propose a novel algorithm for price offering, which takes into account the expected rewards while considering the presence of overlapping intervals.

4.3.1 Proposed Reward Algorithm 2

In order to address the challenges posed by probabilistic valuations and overlapping intervals, we introduce a novel reward-based algorithm. This algorithm builds upon

Reward Algorithm 1 by incorporating modifications that involve identifying and merging overlapping intervals. The proposed algorithm is outlined as follows:

- **Offline phase**
  - Provided with the preference dataset $\{(x^t, y^t)\}$ of size T, derive the reward dataset $\{(x^t, y^t, r^t)\}$ and convert it to obtain the converted dataset $\{(x^t, y^t, \hat{r}^t)\}$.
  - Find the optimal interval $[\hat{l}_{optimal}, \hat{u}_{optimal}]$ by estimating the lower and upper bounds of valuations via Offline Bounds Estimation and the hybrid method
  - Check for overlapping intervals. If there are overlapping intervals, merge them (merged interval)
  - Find the interval for offering prices during online learning (price interval):
    - If there is a merged interval:
      - If the optimal interval is within the merged interval:
        price interval = merged interval
      - Otherwise:
        price interval = optimal interval
    - If there is no merged interval:
      - price interval = optimal interval
  - Set the initial value for ϵ : $\epsilon = \epsilon_0$
- **Online phase**
  - For n_{iter} iterations:
    - Offer price x to the incoming customer
      - With probability ϵ offer the price in the price interval with the highest expected reward $\hat{r}$
      - With probability $1 - \epsilon$ offer a price uniformly distributed in the price interval
    - Getting the customers' feedback y
    - Cacluate the reward r for the new data point (x, y)

* Add the new data point (x,y,r) to the existing reward dataset
* Re-calculate the $\hat{r}$ values of the data points in the converted dataset
* Increase the ϵ value

The first two steps of the offline phase closely resemble those of Reward Algorithm 1, encompassing the derivation of the reward dataset, converted dataset, and identification of the optimal interval.

Subsequently, we try to detect overlapping intervals. As previously discussed, the presence of a small difference between the upper and lower bounds of consecutive intervals serves as an indicator of overlap. Leveraging this concept, we introduce the $interval_{threshold}$ hyperparameter to establish a criterion for determining the magnitude of a "small" difference. Following the estimation of bounds, if the difference between the upper bound of one interval and the lower bound of the subsequent interval falls below the defined threshold, we infer the presence of overlap and merge these intervals. Through this iterative process for all intervals, we obtain a unified interval known as the merged interval.

Next, we embark on identifying the interval suitable for price offerings to customers during the online learning phase. Diverging from Reward Algorithm 1, where the offering interval consistently aligns with the optimal interval, Reward Algorithm 2 introduces variations. Specifically, if a merged interval exists, we verify whether the estimated optimal interval falls within this merged interval. If it does, we utilize the merged interval for price offerings during online learning. In the case that the estimated optimal interval lies outside the merged interval or if no merged interval is present, we revert to using the estimated optimal interval for price offerings during online learning.

The subsequent steps of Reward Algorithm 2 closely resemble those of Algorithm 1, involving price offerings within the determined interval, a reward-based and epsilon-greedy approach for price selection, customer feedback integration, dataset updates, recalculation of expected rewards, and iterative execution for a specified number of iterations (n_{iter}).

4.3.2 Simulation

In order to evaluate the efficacy of the proposed Reward Algorithm 2 in achieving higher rewards when dealing with overlapping intervals, a simulation study was conducted. Similar to the previous simulation study, the value of parameter k in this algorithm is directly proportional to the size of the dataset and is incrementally adjusted during the online learning phase. In the offline phase, where we have access to the preference dataset, two initial values for k were considered: 62 and 125, which correspond to 2.5% and 5% of the dataset size during this phase, respectively. During the offline phase, the preference dataset undergoes two conversions to determine the estimated lower and upper bounds. The IQR method is employed to detect and remove outliers from the dataset during this estimation process. However, when converting the reward dataset to calculate the expected rewards, the conversion is performed only once and outliers are not taken into consideration. The values of the $difference_{factor}$ and $scale$ hyperparameters are determined in proportion to the percentage deviation of valuations. To increment the ϵ value during the online learning phase, the first method is employed, continuously increasing ϵ. The complete configuration of this simulation study is presented in Table 4.20, which provides a comprehensive overview of the specific settings and hyperparameters employed for the evaluation. Each setting underwent 100 replications of the simulation.

Table 4.20 Setting of the simulation study for evaluating the Reward Algorithm 2 in the presence of probabilistic valuations, overlapping intervals, and known distribution

Number of segments	n	4
Number of data points	T	2500
Number of iterations in online learning phase	n_{iter}	5000
Number of nearest neighbors	k	[62,125]
Number of data conversion	$n_{conversion}$	[1, 2]
Non-uniform valuations	v_i	[20, 45, 55, 80]
Non-uniform distribution	δ_i	[0.1, 0.2, 0.3, 0.4]
Initial epsilon	ϵ_0	[0.00, 0.25, 0.50, 0.75]
Difference factor for comparing the consecutive differences	$difference_{factor}$	1.15
Threshold for measuring the closeness of prices to the purchase probabilities	$price_{threshold}$	0.01
Scale factor for widening the optimal interval	$scale$	0.15
Valuation percentage deviation	$p\%$	15%
Threshold for detecting overlapping intervals	$interval_{threshold}$	2

The results of the simulation study are presented in Table 4.21, which includes the median of average rewards collected during the online learning phase, the median of average rewards collected during the last 1000 iterations of the online learning phase, and the percentage of instances where the right interval (i.e., the interval containing the optimal price with the highest reward) was identified, for various initial epsilon (ϵ_0) values. Consistent with the previous simulation study, 80% of the online learning phase (4000 iterations) was allocated for increasing the value of epsilon (ϵ) to explore the optimal interval. In the remaining 20% of the online learning phase (1000 iterations), ϵ was fixed at 1 to exploit prices yielding the highest rewards. The highest average reward attained during the final 1000 iterations was 36.119, achieved when employing an initial epsilon value of 0.50 ($\epsilon_0 = 0.50$). This corresponds to 99.50% of the expected reward (36.30). These outcomes demonstrate that the integration of intervals and the offering of prices within the merged interval lead to the attainment of high rewards, closely approximating the expected rewards. Moreover, the algorithm is successful in identifying the right interval for offering prices. Furthermore, these results affirm the effectiveness of the proposed Algorithm 2 in increasing revenue for the seller.

Table 4.21 Average rewards obtained by Reward Algorithm 2 in the presence of probabilistic valuations, overlapping intervals, and known distribution

	initial $k = 62$			**initial $k = 125$**		
	Average Reward	**Average Reward during last 1000 iteration**	**right interval percentage**	**Average Reward**	**Average Reward during last 1000 iteration**	**right interval percentage**
$\epsilon_0 = 0.00$	33.745	35.993	1.000	29.596	35.953	1.000
$\epsilon_0 = 0.25$	34.352	35.997	1.000	31.146	36.005	1.000
$\epsilon_0 = 0.50$	34.739	36.119	1.000	32.698	35.818	1.000
$\epsilon_0 = 0.75$	35.253	35.910	1.000	34.442	35.819	1.000

4.4 Realized Revenue when there is no prior knowledge regarding the market structure

In this chapter, we explored the rewards that can be achieved when we have knowledge regarding the number of segments (n) and their distribution (δ_i) in the market. Now, our focus is on assessing the rewards attainable when dealing with the lowest level of foreknowledge regarding the market structure, where all parameters (n, v_i, δ_i) are unknown.

With the algorithms developed in the previous chapter, we can estimate the number of segments (n) and their distribution (δ_i), allowing us to subsequently estimate the valuations (v_i). By estimating all the relevant parameters in the problem, we can utilize Equation 2.1 to find the estimated optimal price that maximizes revenue.

As observed in the results of Section 3.4, number of segments cannot be estimated accurately all the time. Yet, we claim that even with a wrong estimate of the n, the optimal price we estimate based on the estimated values of the parameters of the problem achieves rewards close to the expected reward. Our main focus in this section is to investigate this.

Once we have the estimated optimal price, we can compare the revenue achieved by this estimated optimal price with the revenue achieved by the true optimal price. This comparison allows us to gauge the extent to which we can approach the true revenue using the developed algorithms in the previous chapter, in scenarios where all parameters of the problem are initially unknown. This analysis provides insights into the performance and effectiveness of the developed algorithms in estimating the optimal price and achieving revenue close to the true optimal revenue, even when confronted with limited foreknowledge about the market structure. It demonstrates the potential for these algorithms to guide decision-making and revenue optimization when facing uncertain market conditions.

4.4.1 Simulation

In order to observe the revenues achieved by the estimated optimal prices obtained through the developed algorithms, several simulation studies were conducted. The first simulation study was conducted in the presence of deterministic valuations $v_i = [8, 46, 56, 81]$ using one feature vector in the converted dataset for clustering. This study involved considering different distributions and dataset sizes. Additionally, the hyperparameter α introduced in Section 4.1.3 was applied in this simulation to investigate its effect on the achieved reward. The value considered for alpha was $\alpha = \frac{100}{2m}$. Each setting of the simulation was replicated 200 times to ensure reliable results and account for variability. The objective of these simulation studies was to assess the performance and effectiveness of the developed algorithms in estimating optimal prices and achieving revenues. By analyzing the obtained results, insights can be gained into the ability of the algorithms to approach the true revenue in various scenarios and the impact of different factors, such as distributions, dataset sizes, and the hyperparameter α, on the achieved reward.

In the reported results, the median of six values is presented, including:

$\hat{n}$: This represents the estimated number of segments in the market. It is obtained through the Number of Segments Estimation Algorithm, which estimates the underlying segmentation based on the available data.

$\hat{x}$: This denotes the estimated optimal price based on the estimated number of segments ($\hat{n}$) and their distribution ($\hat{\delta}_i$). The calculation of $\hat{x}$ can be performed as follows:

$$\hat{x} = \arg \max_{i=1,...,\hat{n}} \{(1 - \sum_{j=1}^{i-1} \hat{\delta}_j) \times \hat{v}_i\} \tag{4.13}$$

Realized revenue by $\hat{x}$: This represents the revenue achieved if $\hat{x}$ is offered to customers in the real market structure. To calculate this, first we find the $\hat{x}$ such that $v_i < \hat{x} \leq v_{i+1}$. This ensures that $\hat{x}$ falls within the price range between two consecutive valuations (v_i and v_{i+1}). then we calculate the purchase probability of customers when $\hat{x}$ is offered as $1 - \sum_{j=1}^{i} \delta_j$. Finally, the realized revenue is calculated as:

$$revenue_{realized} = (1 - \sum_{j=1}^{i} \delta_j) * \hat{x} \tag{4.14}$$

The true optimal price ($x^{optimal}$) and true revenue refer to the optimal price and revenue achieved in the real market structure. These values are determined based on the actual market structure.

The deviation percentage between the realized revenue and the true revenue is a metric that indicates how close the realized revenue is to the true revenue. It is calculated as $\frac{(revenue_{realized} - revenue_{true})}{revenue_{true}} * 100$. This metric provides a quantitative measure of the relative difference between the revenue achieved in the simulation study (realized revenue) and the revenue that would be achieved in the real market (true revenue). A smaller deviation percentage indicates a closer approximation of the true revenue by the realized revenue. The deviation percentage allows for an assessment of the accuracy and effectiveness of the developed algorithms in estimating and approaching the true revenue in the absence of complete knowledge about the market structure and parameters.

Table 4.22 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of deterministic valuations and distribution $\delta_i = [0.1, 0.2, 0.3, 0.4]$

			$\delta_i = [0.1, 0.2, 0.3, 0.4]$														
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	52.50	33.25	46.00	41.40	-19.69	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	55.00	34.30	46.00	41.40	-17.15
		$\alpha = 2.5$	7.00	50.00	35.00	46.00	41.40	-15.46			$\alpha = 1$	7.00	52.00	34.30	46.00	41.40	-17.15
	$m' = 100$	$\alpha = 0$	6.00	52.50	33.25	46.00	41.40	-19.69		$m' = 40$	$\alpha = 0$	5.00	45.00	22.40	46.00	41.40	-45.89
		$\alpha = 2.5$	6.00	47.50	38.50	46.00	41.40	-7.00			$\alpha = 1$	7.00	51.00	34.30	46.00	41.40	-17.15
	$m' = 150$	$\alpha = 0$	5.00	50.00	33.25	46.00	41.40	-19.69		$m' = 60$	$\alpha = 0$	7.00	51.00	34.30	46.00	41.40	-17.15
		$\alpha = 2.5$	5.00	45.00	40.50	46.00	41.40	-2.17			$\alpha = 1$	7.00	50.50	34.30	46.00	41.40	-17.15
	$m' = 200$	$\alpha = 0$	5.00	47.50	33.25	46.00	41.40	-19.69		$m' = 80$	$\alpha = 0$	7.00	50.00	32.90	46.00	41.40	-20.53
		$\alpha = 2.5$	5.00	45.00	40.50	46.00	41.40	-2.17			$\alpha = 1$	7.00	48.00	35.70	46.00	41.40	-13.77
	$m' = 250$	$\alpha = 0$	5.00	47.50	33.25	46.00	41.40	-19.69		$m' = 100$	$\alpha = 0$	6.00	49.00	32.90	46.00	41.40	-20.53
		$\alpha = 2.5$	5.00	45.00	40.50	46.00	41.40	-2.17			$\alpha = 1$	6.00	47.00	37.45	46.00	41.40	-9.54
	$m' = 500$	$\alpha = 0$	4.00	47.50	33.25	46.00	41.40	-19.69		$m' = 200$	$\alpha = 0$	5.00	47.00	32.90	46.00	41.40	-20.53
		$\alpha = 2.5$	4.00	45.00	40.50	46.00	41.40	-2.17			$\alpha = 1$	5.00	46.00	41.40	46.00	41.40	0.00
	$m' = 1000$	$\alpha = 0$	4.00	47.50	33.25	46.00	41.40	-19.69		$m' = 400$	$\alpha = 0$	4.00	47.00	32.90	46.00	41.40	-20.53
		$\alpha = 2.5$	4.00	45.00	40.50	46.00	41.40	-2.17			$\alpha = 1$	4.00	46.00	41.40	46.00	41.40	0.00

Table 4.22 presents the simulation results for distribution $\delta_i = [0.1, 0.2, 0.3, 0.4]$, revealing several important observations. Firstly, when the α hyperparameter is not applied, a significant difference is observed between the realized revenue and the true revenue. In some cases, this difference even increases with larger dataset sizes. This disparity can be attributed to the overshooting of the optimal price, which leads to the loss of a portion of customers and subsequently decreases the overall reward.

However, when the α hyperparameter is applied, the deviation percentage between the realized revenue and the true revenue is equal to or less than the case where α is not applied. This demonstrates the effectiveness of applying the alpha hyperparameter in reducing the disparity between the realized and true revenue. Additionally, as the size of the dataset increases, the deviation percentage decreases and converges to 0. This highlights the positive impact of dataset size on the accuracy of the realized revenue.

Furthermore, it is worth noting that in some cases, even when the estimated number of segments is not the true value, the deviation percentage remains very small. This suggests that, despite an incorrect estimation of the number of segments, the price offered is still close to the optimal price. Consequently, the developed algorithms can be employed for price offering purposes.

Table 4.23 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of deterministic valuations and distribution $\delta_i = [0.4, 0.3, 0.2, 0.1]$

			$\delta_i = [0.4, 0.3, 0.2, 0.1]$														
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	45.00	15.75	46.00	27.60	-42.93	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	43.50	16.20	46.00	27.60	-41.30
		$\alpha = 2.5$	7.00	42.50	22.50	46.00	27.60	-18.48			$\alpha = 1$	7.00	40.50	16.20	46.00	27.60	-41.30
	$m' = 100$	$\alpha = 0$	6.00	45.00	15.00	46.00	27.60	-45.65		$m' = 40$	$\alpha = 0$	7.00	46.00	16.20	46.00	27.60	-41.30
		$\alpha = 2.5$	6.00	45.00	25.50	46.00	27.60	-7.61			$\alpha = 1$	7.00	44.50	22.50	46.00	27.60	-18.48
	$m' = 150$	$\alpha = 0$	5.00	47.50	14.25	46.00	27.60	-48.37		$m' = 60$	$\alpha = 0$	7.00	47.00	15.90	46.00	27.60	-42.39
		$\alpha = 2.5$	5.00	45.00	25.50	46.00	27.60	-7.61			$\alpha = 1$	7.00	45.00	25.80	46.00	27.60	-6.52
	$m' = 200$	$\alpha = 0$	5.00	47.50	14.25	46.00	27.60	-48.37		$m' = 80$	$\alpha = 0$	7.00	46.00	20.40	46.00	27.60	-26.09
		$\alpha = 2.5$	5.00	45.00	27.00	46.00	27.60	-2.17			$\alpha = 1$	7.00	45.50	26.40	46.00	27.60	-4.35
	$m' = 250$	$\alpha = 0$	5.00	47.50	14.25	46.00	27.60	-48.37		$m' = 100$	$\alpha = 0$	7.00	46.00	21.60	46.00	27.60	-21.74
		$\alpha = 2.5$	5.00	45.00	27.00	46.00	27.60	-2.17			$\alpha = 1$	7.00	45.00	25.80	46.00	27.60	-6.52
	$m' = 500$	$\alpha = 0$	4.00	47.50	14.25	46.00	27.60	-48.37		$m' = 200$	$\alpha = 0$	5.00	47.00	14.10	46.00	27.60	-48.91
		$\alpha = 2.5$	4.00	45.00	27.00	46.00	27.60	-2.17			$\alpha = 1$	5.00	46.00	27.00	46.00	27.60	-2.17
	$m' = 1000$	$\alpha = 0$	4.00	47.50	14.25	46.00	27.60	-48.37		$m' = 400$	$\alpha = 0$	4.00	47.00	14.10	46.00	27.60	-48.91
		$\alpha = 2.5$	4.00	45.00	27.00	46.00	27.60	-2.17			$\alpha = 1$	4.00	46.00	27.60	46.00	27.60	0.00

Table 4.23 displays the simulation results for distribution $\delta_i = [0.4, 0.3, 0.2, 0.1]$. Similar to the previous results, the impact of the α hyperparameter is evident in reducing the deviation percentage between the realized revenue and the true revenue.

As the dataset size increases, the deviation percentage decreases and approaches 0 when α is applied. This signifies the effectiveness of the α hyperparameter in aligning the realized revenue closer to the true revenue.

Furthermore, it is notable that the difference in deviation percentage between the cases where α is applied and not applied can be significant for certain dataset sizes. This discrepancy arises due to the overshooting of the optimal price and the characteristics of the distribution itself. In such cases, the application of the α hyperparameter plays a crucial role in reducing the deviation and improving the realized revenue.

Table 4.24 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of deterministic valuations and random distribution with minimum probability $R = 0.1$

			Random distribution with $R = 0.1$														
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	47.50	15.84	46.00	27.60	-38.15	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	46.00	15.45	46.00	26.91	-36.18
		$\alpha = 2.5$	7.00	45.00	21.73	46.00	29.67	-17.96			$\alpha = 1$	7.00	46.00	16.08	46.00	27.60	-39.04
	$m' = 100$	$\alpha = 0$	6.00	47.50	15.45	46.00	26.91	-37.43		$m' = 40$	$\alpha = 0$	7.00	47.00	16.96	46.00	27.37	-30.89
		$\alpha = 2.5$	6.00	45.00	22.08	46.00	27.14	-13.04			$\alpha = 1$	7.00	48.00	20.52	46.00	30.07	-28.26
	$m' = 150$	$\alpha = 0$	5.00	47.50	19.88	46.00	31.05	-29.52		$m' = 60$	$\alpha = 0$	7.00	48.00	19.10	46.00	28.06	-23.93
		$\alpha = 2.5$	5.00	45.00	25.20	46.00	28.98	-7.61			$\alpha = 1$	7.00	47.00	19.35	46.00	27.14	-24.97
	$m' = 200$	$\alpha = 0$	5.00	47.50	16.15	46.00	28.29	-38.28		$m' = 80$	$\alpha = 0$	7.00	47.50	18.70	46.00	28.52	-29.48
		$\alpha = 2.5$	5.00	45.00	24.30	46.00	28.06	-3.50			$\alpha = 1$	7.00	48.00	19.37	46.00	27.01	-19.03
	$m' = 250$	$\alpha = 0$	5.00	47.50	15.10	46.00	28.06	-39.09		$m' = 100$	$\alpha = 0$	7.00	47.00	19.12	46.00	27.37	-25.79
		$\alpha = 2.5$	5.00	45.00	23.50	46.00	26.45	-2.17			$\alpha = 1$	7.00	46.00	21.73	46.00	28.06	-13.31
	$m' = 500$	$\alpha = 0$	4.00	47.50	14.25	46.00	26.22	-39.58		$m' = 200$	$\alpha = 0$	5.00	47.00	16.94	46.00	26.68	-30.79
		$\alpha = 2.5$	4.00	45.00	26.10	46.00	26.68	-2.17			$\alpha = 1$	5.00	46.00	23.84	46.00	28.29	-4.35
	$m' = 1000$	$\alpha = 0$	4.00	47.50	14.25	46.00	28.98	-40.13		$m' = 400$	$\alpha = 0$	4.00	47.00	13.82	46.00	26.22	-40.72
		$\alpha = 2.5$	4.00	45.00	26.78	46.00	27.37	-2.17			$\alpha = 1$	4.00	46.00	27.06	46.00	28.52	0.00

Table 4.24 presents the simulation results for a random distribution with a minimum probability of $R = 0.1$. The results align with the previous findings, reinforcing the observations regarding the impact of deploying the α hyperparameter and the dataset size on the deviation percentage and the convergence of the realized revenue to the true revenue.

As observed in the previous simulations, when α is applied and the dataset size increases, the deviation percentages become smaller, indicating a closer approximation of the realized revenue to the true revenue. This highlights the effectiveness of the α hyperparameter in mitigating deviations and improving the accuracy of the revenue estimation.

By considering these findings, it is evident that the developed algorithms, when combined with the α hyperparameter and larger datasets, can yield more accurate and reliable estimates of the revenue in scenarios where a random distribution with a minimum probability is present. These insights emphasize the importance of these factors in achieving better performance and more precise revenue predictions.

In the second simulation study, probabilistic valuations with 5% deviations were considered, and the focus was on assessing the realized revenues with minimal foreknowledge regarding the market structure. The objective was to determine the impact of deploying the α hyperparameter on improving these revenues. The simulation results for different distributions are presented in Tables 4.25, 4.26, and 4.27.

These tables provide insights into the realized revenues achieved in the presence of probabilistic valuations and with the least degrees of knowledge about the market structure. By analyzing the results, it can be observed whether deploying the α hyperparameter leads to improvements in the realized revenues.

The findings from these simulation studies enable a comprehensive understanding of the performance of the developed algorithms in scenarios with probabilistic valuations. They shed light on the effectiveness of the algorithms and the significance of the α hyperparameter in enhancing the achieved revenues when dealing with limited foreknowledge and probabilistic valuations.

Table 4.25 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of probabilistic valuations and distribution $\delta_i = [0.1, 0.2, 0.3, 0.4]$

			$\delta_i = [0.1, 0.2, 0.3, 0.4]$															
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	50.00	34.90	43.70	39.33	-11.26	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	53.00	34.30	43.70	39.33	-12.79
		$\alpha = 2.5$	7.00	47.50	35.00	43.70	39.33	-11.01			$\alpha = 1$	7.00	52.00	34.30	43.70	39.33	-12.79
	$m' = 100$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 40$	$\alpha = 0$	7.00	50.00	35.56	43.70	39.33	-9.59
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	49.00	35.49	43.70	39.33	-9.77
	$m' = 150$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 60$	$\alpha = 0$	7.00	49.00	35.56	43.70	39.33	-9.59
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	48.00	35.70	43.70	39.33	-9.23
	$m' = 200$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 80$	$\alpha = 0$	7.00	47.00	35.56	43.70	39.33	-9.59
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	46.00	36.80	43.70	39.33	-6.43
	$m' = 250$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 100$	$\alpha = 0$	7.00	47.00	35.56	43.70	39.33	-9.59
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	46.00	36.80	43.70	39.33	-6.43
	$m' = 500$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 200$	$\alpha = 0$	7.00	45.00	37.96	43.70	39.33	-3.49
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	44.00	39.03	43.70	39.33	-0.77
	$m' = 1000$	$\alpha = 0$	7.00	47.50	34.90	43.70	39.33	-11.26		$m' = 400$	$\alpha = 0$	7.00	45.00	37.96	43.70	39.33	-3.49
		$\alpha = 2.5$	7.00	45.00	37.96	43.70	39.33	-3.49			$\alpha = 1$	7.00	44.00	39.03	43.70	39.33	-0.77

Table 4.25 displays the simulation results for distribution $\delta_i = [0.1, 0.2, 0.3, 0.4]$. Unlike the deterministic case, where the deviation percentage is generally smaller when the α hyperparameter is deployed, in some cases of m and m', the deviation percentage is greater when α is used in the presence of probabilistic valuations. This discrepancy can be attributed to the nature of probabilistic valuations, where the values can vary for customers within a segment. Consequently, simply deducting a fixed value of α does not guarantee that the offered price falls below the optimal price, leading to potential deviations in the realized revenue.

However, it is important to note that with a large dataset, the difference between the realized revenue and the true revenue is not significant. This suggests that, even though the deviation percentage may be higher when deploying the α hyperparameter, the overall impact on revenue is not substantial.

Considering these results, it is advisable for sellers to gather large datasets whenever possible. Larger datasets enable more accurate estimations and help minimize the deviation between the realized revenue and the true revenue, especially in scenarios involving probabilistic valuations. These findings emphasize the importance of dataset size in obtaining reliable revenue estimations when dealing with limited foreknowledge and probabilistic valuations.

Table 4.26 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of probabilistic valuations and distribution $\delta_i = [0.4, 0.3, 0.2, 0.1]$

			$\delta_i = [0.4, 0.3, 0.2, 0.1]$														
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	47.50	16.73	43.70	26.22	-36.20	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	45.00	16.20	43.70	26.22	-38.22
		$\alpha = 2.5$	7.00	45.00	23.18	43.70	26.22	-11.58			$\alpha = 1$	7.00	45.50	15.60	43.70	26.22	-40.50
	$m' = 100$	$\alpha = 0$	6.00	47.50	16.73	43.70	26.22	-36.20		$m' = 40$	$\alpha = 0$	7.00	47.00	18.08	43.70	26.22	-31.03
		$\alpha = 2.5$	7.00	45.00	23.18	43.70	26.22	-11.58			$\alpha = 1$	7.00	46.00	20.70	43.70	26.22	-21.05
	$m' = 150$	$\alpha = 0$	6.00	47.50	16.73	43.70	26.22	-36.20		$m' = 60$	$\alpha = 0$	7.00	47.00	18.08	43.70	26.22	-31.03
		$\alpha = 2.5$	6.00	45.00	23.18	43.70	26.22	-11.58			$\alpha = 1$	7.00	46.00	20.70	43.70	26.22	-21.05
	$m' = 200$	$\alpha = 0$	6.00	47.50	16.73	43.70	26.22	-36.20		$m' = 80$	$\alpha = 0$	7.00	47.00	18.08	43.70	26.22	-31.03
		$\alpha = 2.5$	6.00	45.00	23.18	43.70	26.22	-11.58			$\alpha = 1$	7.00	46.00	20.70	43.70	26.22	-21.05
	$m' = 250$	$\alpha = 0$	6.00	47.50	16.73	43.70	26.22	-36.20		$m' = 100$	$\alpha = 0$	7.00	47.00	18.08	43.70	26.22	-31.03
		$\alpha = 2.5$	6.00	45.00	23.18	43.70	26.22	-11.58			$\alpha = 1$	7.00	44.00	23.10	43.70	26.22	-11.90
	$m' = 500$	$\alpha = 0$	6.00	42.50	25.50	43.70	26.22	-2.75		$m' = 200$	$\alpha = 0$	7.00	45.00	23.18	43.70	26.22	-11.58
		$\alpha = 2.5$	6.00	40.00	24.00	43.70	26.22	-8.47			$\alpha = 1$	7.00	44.00	25.54	43.70	26.22	-2.60
	$m' = 1000$	$\alpha = 0$	6.00	42.50	25.50	43.70	26.22	-2.75		$m' = 400$	$\alpha = 0$	7.00	45.00	23.18	43.70	26.22	-11.58
		$\alpha = 2.5$	6.00	40.00	24.00	43.70	26.22	-8.47			$\alpha = 1$	7.00	44.00	25.54	43.70	26.22	-2.60

Table 4.27 Realized revenue by applying the Number of Segments Estimation Algorithm in the presence of probabilistic valuations and random distribution with minimum probability $R = 0.1$

			Random distribution with $R = 0.1$														
			$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue				$\hat{n}$	Estimated Optimal Price ($\hat{x}$)	Realized Revenue by $\hat{x}$	Optimal Price $x^{optimal}$	True Revenue by $x^{optimal}$	Deviation percentage between True Revenue & Realized Revenue
$m = 20$	$m' = 50$	$\alpha = 0$	7.00	47.50	18.76	43.70	27.09	-25.63	$m = 50$	$m' = 20$	$\alpha = 0$	7.00	48.50	16.06	43.70	25.35	-31.42
		$\alpha = 2.5$	7.00	45.00	21.59	43.70	25.78	-16.01			$\alpha = 1$	7.00	48.00	18.20	43.70	26.66	-30.22
	$m' = 100$	$\alpha = 0$	7.00	47.50	18.90	43.70	26.22	-19.91		$m' = 40$	$\alpha = 0$	7.00	49.00	18.46	43.70	26.00	-22.43
		$\alpha = 2.5$	6.00	45.00	23.45	43.70	26.66	-12.53			$\alpha = 1$	7.00	46.00	21.46	43.70	27.09	-19.34
	$m' = 150$	$\alpha = 0$	6.00	47.50	19.43	43.70	26.66	-19.58		$m' = 60$	$\alpha = 0$	7.00	47.00	21.30	43.70	27.09	-18.18
		$\alpha = 2.5$	6.00	45.00	24.43	43.70	27.09	-8.47			$\alpha = 1$	7.00	48.00	19.80	43.70	25.13	-20.20
	$m' = 200$	$\alpha = 0$	6.00	47.50	19.42	43.70	26.22	-17.38		$m' = 80$	$\alpha = 0$	7.00	47.00	20.40	43.70	26.00	-18.53
		$\alpha = 2.5$	6.00	45.00	23.30	43.70	25.35	-8.47			$\alpha = 1$	7.00	46.00	20.70	43.70	26.22	-14.66
	$m' = 250$	$\alpha = 0$	6.00	47.50	21.65	43.70	26.88	-13.78		$m' = 100$	$\alpha = 0$	7.00	47.00	22.18	43.70	26.66	-15.62
		$\alpha = 2.5$	6.00	45.00	23.26	43.70	26.44	-8.47			$\alpha = 1$	7.00	46.00	23.67	43.70	26.22	-10.76
	$m' = 500$	$\alpha = 0$	6.00	47.50	21.23	43.70	24.47	-12.50		$m' = 200$	$\alpha = 0$	7.00	46.00	23.23	43.70	26.22	-13.76
		$\alpha = 2.5$	6.00	45.00	25.20	43.70	27.53	-8.47			$\alpha = 1$	7.00	46.00	22.85	43.70	25.35	-8.63
	$m' = 1000$	$\alpha = 0$	6.00	47.50	24.14	43.70	27.31	-8.25		$m' = 400$	$\alpha = 0$	7.00	45.00	24.19	43.70	27.53	-12.47
		$\alpha = 2.5$	6.00	45.00	23.48	43.70	25.56	-8.47			$\alpha = 1$	6.00	44.00	25.84	43.70	26.88	-3.38

The same observations can be made for the results in Tables 4.26, and 4.27. In both cases, applying the α hyperparameter does not consistently result in improved realized revenues. The impact of α can vary depending on the specific distribution and dataset size.

However, it is evident that using large datasets consistently leads to higher realized revenues that are closer to the true revenue. The availability of more data points allows for more accurate estimations and reduces the deviation between the realized revenue and the true revenue.

These findings highlight the significance of dataset size in obtaining reliable revenue estimates when dealing with probabilistic valuations and limited foreknowledge. While the impact of the α hyperparameter may vary, sellers are advised to focus on gathering large datasets to enhance the accuracy of revenue predictions and approach the true revenue more closely.

In summary, the algorithm developed for estimating the number of segments and their distribution demonstrates successful results in accurately estimating these parameters and providing prices that yield revenue close to the expected revenue. Even in cases where the number of segments is not estimated accurately, the estimated optimal price based on the estimated parameters of the problem yields rewards pretty close to the expected reward. For the case of deterministic valuations, the deployment of the hyperparameter α improves the realized revenue. Additionally, utilizing large datasets leads to a convergence of the realized revenue towards the true revenue. These findings emphasize the importance of considering α and gathering extensive datasets to enhance revenue estimation accuracy in scenarios with deterministic valuations. On the other hand, for the case of probabilistic valuations, the impact of the α hyperparameter on the realized revenue is not consistently positive. However, employing large datasets consistently brings the realized revenue closer to the true revenue. This highlights the significance of utilizing extensive datasets to achieve more accurate revenue estimates in scenarios involving probabilistic valuations. Overall, the combination of deploying α, especially in deterministic valuations, and utilizing large datasets is crucial for improving revenue estimation accuracy and approaching the true revenue. These findings provide valuable insights for decision-makers in pricing strategies and underscore the importance of data-driven approaches in revenue optimization.

5. Conclusion

In chapters 3 and 4, several algorithms were developed to address the estimation of unknown parameters and reward maximization in the pricing problem.

In the offline learning phase, the research began with a high level of prior knowledge about the market structure, including known numbers of segments, their distributions, and their order. When dealing with deterministic valuations, the proposed Offline Algorithm 1 successfully estimated customers' valuations with low error rates, especially when the preference dataset had a reasonable size and outliers were eliminated through double conversion. Additionally, smart learning, where the first and last valuations were learned separately, resulted in better estimation. In the case of probabilistic valuations, the proposed algorithm accurately estimated valuations, but applying smart learning in cases of wide valuation intervals did not lead to accurate estimates. For wide intervals, it was better to estimate valuations without applying smart learning.

As the level of prior knowledge decreased, the research focused on the scenario where the order of segment distributions was unknown. The proposed method for finding the correct order by trying all possible orders and applying Offline Algorithm 1 was effective in identifying the correct order with high probabilities.

The investigation then moved to the scenario where segment distributions were unknown, and only the number of segments was known. The proposed Offline Algorithm 2 successfully estimated customers' valuations and segment distributions, but only when these parameters followed a uniform distribution. In cases with non-uniform valuations and distributions, the research proposed offering discrete prices instead of continuous prices to estimate the unknown parameters. The Distribution Estimation Algorithm was introduced for estimating segment distributions and subsequently customers' valuations. The results showed successful estimations with low error, and larger datasets resulted in lower estimation errors, suggesting the collection of large datasets whenever possible.

Finally, the research addressed scenarios with the least prior knowledge about the market structure, where all parameters were unknown. The Number of Segments Estimation Algorithm was proposed to estimate the number of segments, distributions, and customers' valuations based on discrete prices. The proposed algorithm generally succeeded in estimating these parameters. In cases where the algorithm did not accurately capture the true number of segments, the estimated optimal prices based on the estimated parameters still yielded rewards close to the expected levels.

In the online learning phase with the objective of reward maximization, several algorithms were developed to achieve this goal.

For deterministic valuations, Online Algorithm 1 was proposed to offer prices during the online learning phase. It was observed that offering prices lower than the estimated price to customers resulted in increased rewards during this phase. This strategy of offering lower prices allowed for more customer purchases and higher revenue generation.

For probabilistic valuations, two scenarios were investigated. In the first scenario, involving non-overlapping intervals, offering prices lower than the estimated prices did not increase the reward. Instead, offering prices within the estimated intervals with the highest expected reward brought higher rewards. Methods were developed to estimate the lower and upper bounds of these intervals, and a formula was derived to offer prices within these estimated bounds, leading to achieving high rewards close to the expected levels. This approach allowed for effective price setting, considering the range of valuations and maximizing revenue potential.

In situations where probabilistic valuations had overlapping intervals, it was observed that merging the overlapping intervals and offering prices within this merged interval achieved higher rewards. Moreover, a new algorithm was developed, focusing on the expected reward of the prices rather than their expected purchase probability. This approach proved successful in achieving high rewards by offering prices within the merged intervals, effectively accounting for the overlapping nature of the valuations.

Overall, the developed algorithms and strategies demonstrated their effectiveness in maximizing rewards during the online learning phase, regardless of the type of valuations present. These findings highlight the importance of adaptive pricing strategies that consider the uncertainty in customer valuations and leverage real-time feedback to optimize revenue. The proposed algorithms offer practical and effective solutions for online pricing decisions in dynamic markets.

BIBLIOGRAPHY

Accenture, I. (2016). Personalization pulse check. https://www.accenture.com/t00010101T000000Z__w__/ar-es/_acnmedia/PDF-37/Accenture-Personalization-Pulse-Check-Infographic.pdf. Accessed: 2021-09-02.

Aoki, M. (1973). On a dual control approach to the pricing policies of a trading specialist. In *5th Conference on Optimization Techniques Part II 5*, (pp. 272–282). Springer.

Aviv, Y. & Pazgal, A. (2006). Pricing of short life-cycle products through active learning.

Baetge, J., Bolenz, G., Ballwieser, W., Hömberg, R., & Wullers, P. (1975). Dynamic price policies in monopolistic competition. In *Modern Trends in Cybernetics and Systems. Proceedings of the Third International Congress of Cybernetics and Systems, Bucharest, Romania*, volume 1, (pp. 401–417).

Ban, G.-Y. & Keskin, N. B. (2021). Personalized dynamic pricing with machine learning: High-dimensional features and heterogeneous elasticity. *Management Science*, *67*(9), 5549–5568.

Besbes, O. & Zeevi, A. (2009). Dynamic pricing without knowing the demand function: Risk bounds and near-optimal algorithms. *Operations research*, *57*(6), 1407–1420.

Besbes, O. & Zeevi, A. (2012). Blind network revenue management. *Operations research*, *60*(6), 1537–1550.

Billstroem, F. & Thore, S. (1964). Simulation experiments with dynamic price strategies in monopoly theory. *Econometric model building. Essays on the Causal Chain Approach*, 297–321.

Birge, J. R., Chen, H., & Keskin, N. B. (2021). Markdown policies for demand learning with forward-looking customers. *Available at SSRN 3299819.*

Broder, J. & Rusmevichientong, P. (2012). Dynamic pricing under a general parametric choice model. *Operations Research*, *60*(4), 965–980.

Bu, J., Simchi-Levi, D., & Xu, Y. (2020). Online pricing with offline data: Phase transition and inverse square law. In *International Conference on Machine Learning*, (pp. 1202–1210). PMLR.

Carvalho, A. X. & Puterman, M. L. (2005). Learning and pricing in an internet environment with binomial demands. *Journal of Revenue and Pricing Management*, *3*, 320–336.

Cheung, W. C., Simchi-Levi, D., & Wang, H. (2017). Dynamic pricing and demand learning with limited price experimentation. *Operations Research*, *65*(6), 1722–1731.

Chong, C.-Y. & Cheng, D. (1975). Multistage pricing under uncertain demand. In *Annals of Economic and Social Measurement, Volume 4, number 2* (pp. 311–323). NBER.

Clifford, S. (2012). Shopper alert: Price may drop for you alone. https://www.nytimes.com/2012/08/10/business/supermarkets-try-customizing-prices-for-shoppers.html. Accessed: 2021-09-02.

Clive, H., Terry, H., & Tim, P. (2004). Scoring points: How tesco is winning customer loyalty.

Davenport, T. H., Mule, L. D., & Lucker, J. (2011). Know what your customers want before they do. *harvard Business review*, *89*(12), 84–92.

Den Boer, A. V. (2014). Dynamic pricing with multiple products and partially specified demand distribution. *Mathematics of operations research*, *39*(3), 863–888.

den Boer, A. V. & Zwart, B. (2014). Simultaneously learning and optimizing using controlled variance pricing. *Management science*, *60*(3), 770–783.

eMarketer (2023). Global retail ecommerce forecast 2023. `https://www.insiderintelligence.com/content/global-retail-ecommerce-forecast-2023`. Accessed: 2023-09-02.

Eren, S. S. & Maglaras, C. (2010). Monopoly pricing with limited demand information. *Journal of revenue and pricing management*, *9*, 23–48.

Farnham, A. (2013). Prices now pegged to your buying history at some markets. `https://abcnews.go.com/Business/supermarkets-introduce-personalized-pricing/story?id=21010246`. Accessed: 2021-09-02.

Ferrari-Trecate, G. & Muselli, M. (2002). A new learning method for piecewise linear regression. In *Artificial Neural Networks—ICANN 2002: International Conference Madrid, Spain, August 28–30, 2002 Proceedings*, (pp. 444–449). Springer.

Ferreira, K. J., Simchi-Levi, D., & Wang, H. (2018). Online network revenue management using thompson sampling. *Operations research*, *66*(6), 1586–1602.

Gallego, G. & Van Ryzin, G. (1994). Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management science*, *40*(8), 999–1020.

Gupta, R. & Pathak, C. (2014). A machine learning framework for predicting purchase by online customers based on dynamic pricing. *Procedia Computer Science*, *36*, 599–605.

Hartigan, J. A. & Wong, M. A. (1979). Algorithm as 136: A k-means clustering algorithm. *Journal of the royal statistical society. series c (applied statistics)*, *28*(1), 100–108.

Javanmard, A. & Nazerzadeh, H. (2019). Dynamic pricing in high-dimensions. *The Journal of Machine Learning Research*, *20*(1), 315–363.

Kleinberg, R. & Leighton, T. (2003). The value of knowing a demand curve: Bounds on regret for online posted-price auctions. In *44th Annual IEEE Symposium on Foundations of Computer Science, 2003. Proceedings.*, (pp. 594–605). IEEE.

Kramer, O. & Kramer, O. (2013). K-nearest neighbors. *Dimensionality reduction with unsupervised nearest neighbors*, 13–23.

Kwon, H. D., Lippman, S. A., & Tang, C. S. (2012). Optimal markdown pricing strategy with demand learning. *Probability in the Engineering and Informational Sciences*, *26*(1), 77–104.

Lazear, E. P. (1984). Retail pricing and clearance sales. Technical report, National Bureau of Economic Research.

Lim, A. E. & Shanthikumar, J. G. (2007). Relative entropy, exponential utility, and robust dynamic pricing. *Operations Research*, *55*(2), 198–214.

Lobo, M. S. & Boyd, S. (2003). Pricing and learning with uncertain demand. In *INFORMS revenue management conference*. Citeseer.

Miao, S. & Chao, X. (2021). Dynamic joint assortment and pricing optimization with demand learning. *Manufacturing & Service Operations Management*, *23*(2), 525–545.

Miao, S. & Wang, Y. (2021). Network revenue management with nonparametric demand learning:\sqrt {T}-regret and polynomial dimension dependency. *Available at SSRN 3948140*.

Nguyen, D. (1984). The monopolistic firm, random demand, and bayesian learning. *Operations Research*, *32*(5), 1038–1051.

Nguyen, D. (1997). *Pricing Under Uncertainty*, (pp. 23–57). Boston, MA: Springer US.

Qu, H., Ryzhov, I. O., & Fu, M. C. (2013). Learning logistic demand curves in business-to-business pricing. In *2013 Winter Simulations Conference (WSC)*, (pp. 29–40). IEEE.

Ramezani, S., Bosman, P. A., & La Poutré, H. (2011). Adaptive strategies for dynamic pricing agents. In *2011 IEEE/WIC/ACM International Conferences on Web Intelligence and Intelligent Agent Technology*, volume 2, (pp. 323–328). IEEE.

Ross, B. (2013). Canned goods-not canned prices with personalized pricing. `https://canadiangrocer.com/canned-goods-not-canned-prices-personalized-pricing`. Accessed: 2021-09-02.

Sass, T. R. (1988). A note on optimal price cutting behavior under demand uncertainty. *The review of economics and statistics*, 336–339.

Shakya, S., Kern, M., Owusu, G., & Chin, C. M. (2012). Neural network demand models and evolutionary optimisers for dynamic pricing. *Knowledge-Based Systems*, *29*, 44–53.

Shakya, S., Oliveira, F., & Owusu, G. (2009). Analysing the effect of demand uncertainty in dynamic pricing with eas. In *Research and Development in Intelligent Systems XXV: Proceedings of AI-2008, The Twenty-eighth SGAI International Conference on Innovative Techniques and Applications of Artificial Intelligence*, (pp. 77–90). Springer.

Thore, S. (1964). Price strategies of an" ignorant" monopolist. *Econometric model building. Essays on the Causal Chain Approach. North Holland Publishing Co., Amsterdam.*

Tukey, J. W. (1977). Exploratory data analysis addision-wesley. *Reading, Ma*, *688*, 581–582.

Wruck, E. G. (1989). *Dynamic pricing implications of uncertainty about demand.* Cornell University.