Markdowns in E-Commerce Fresh Retail: A Counterfactual Prediction and Multi-Period Optimization Approach

Junhao Hua, Ling Yan, Huan Xu, Cheng Yang
Alibaba Group
Hangzhou, Zhejiang, China
{junhao.hjh,yanling.yl,huan.xu,charis.yangc}@alibaba-inc.com

ABSTRACT

While markdowns in retail have been studied for decades in traditional business, nowadays e-commerce fresh retail brings much more challenges. Due to the limited shelf life of perishable products and the limited opportunity of price changes, it is difficult to predict sales of a product at a counterfactual price, and therefore it is hard to determine the optimal discount price to control inventory and to maximize future revenue. Traditional machine learning-based methods have high predictability but they can not reveal the relationship between sales and price properly. Traditional economic models have high interpretability but their prediction accuracy is low. In this paper, by leveraging abundant observational transaction data, we propose a novel data-driven and interpretable pricing approach for markdowns, consisting of counterfactual prediction and multi-period price optimization. Firstly, we build a semi-parametric structural model to learn individual price elasticity and predict counterfactual demand. This semi-parametric model takes advantage of both the predictability of nonparametric machine learning model and the interpretability of economic model. Secondly, we propose a multi-period dynamic pricing algorithm to maximize the overall profit of a perishable product over its finite selling horizon. Different with the traditional approaches that use the deterministic demand, we model the uncertainty of counterfactual demand since it inevitably has randomness in the prediction process. Based on the stochastic model, we derive a sequential pricing strategy by Markov decision process, and design a two-stage algorithm to solve it. The proposed algorithm is very efficient. It reduces the time complexity from exponential to polynomial. Experimental results show the advantages of our pricing algorithm, and the proposed framework has been successfully deployed to the well-known e-commerce fresh retail scenario - Freshippo.

CCS CONCEPTS

• **Information systems** → **Online shopping**; • **Applied computing** → *Online auctions*; **Forecasting**; **Marketing**; *Multi-criterion optimization and decision-making*; *Consumer products*.

KEYWORDS

Online marketing, dynamic pricing, counterfactual prediction, demand learning, multi-period optimization

1 INTRODUCTION

In a fresh retail scenario, the freshness of goods is the main concern of the customers. Many perishable products, such as vegetables, meat, milk, eggs, bread, have a limited shelf life. To provide fresh and high-quality goods, it is very important to control inventory. If goods have not been sold out before their expiry date, the retailer will have a substantial loss. To maximize the total profit, promotional markdown is a common approach for e-commerce fresh retails. However, typically, the retailer does not know what is the best discount price and often chooses an empirical discount on goods (such as 30% off, 50% off), which is usually not the optimal solution.

In this paper, we consider the e-commerce retailer that its fresh store has two channels for selling goods: the *normal channel*, where goods are sold by no-discount retail price, and the *markdown channel*, where customers can buy goods by discount under the condition that their total purchase has reached a certain amount. In particular, we consider the well-known e-commerce fresh retail - Freshippo[1]. As shown in Figure 1, customers can either buy goods in normal channel with the retail price or buy them in markdown channel with the discount price. In order to maximize the profit, the retailer needs to answer two questions. First, can goods be sold out with the retail price before its expiry date? Second, if not, what is the optimal discount price for promotional markdown to ensure the goods being sold out while maximizing the profit? The first problem is about sales forecasting [9, 23], and the second problem is about price-demand curve fitting and price optimization [8, 10, 11, 28]. This paper focuses on the second problem. Specifically, we consider the multi-period dynamic pricing problem of a perishable product sold in markdown channel over its shelf life. Our aim is to learn price-demand model rightly and optimize price over a multi-period time horizon dynamically.

---

[1]Freshippo (https://www.freshhema.com/) is an online-to-offline (O2O) service platform provided by Alibaba group. It takes the online shopping experience to bricks-and-mortar stores. It integrates the online and offline stores, uses artificial intelligence technology and operations research methods to provide customers best shopping experience and build ecological industrial chain.
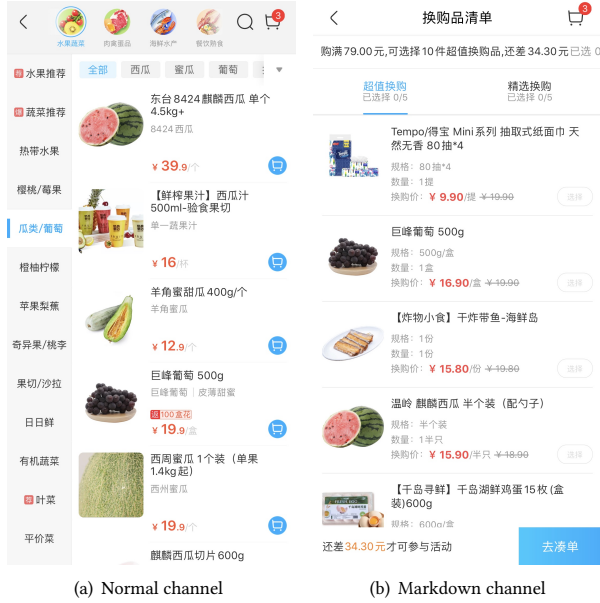
Figure 1: An example of normal channel and markdown channel on Freshippo.

Figure 2: Our framework for markdowns in fresh retail.

The main challenge of demand learning is that the price of most of goods are changed infrequently for fairness, and some even never be changed. It is hard to predict the sales of a product at the discount price which is not observed before. In the literature of causality, this problem is called counterfactual inference [25–27], where the price is the treatment/intervention of the markdown and the sales is the outcome of the markdown. We are interested in studying how the outcome changes with different values of the intervention. For example, suppose we observe the sales of a product with price A and B, we aim to predict the sales of a product with price C, which is counterfactual. The seller can use the price experimentation for demand learning, however the randomized control trail is too costly and has a risk of price discrimination. Alternatively, we focus on the observational study for learning price-demand curve. In other words, we attempt to learn price-demand function from the observational data. It is infeasible to learn individual demand function of a product using only its own data due to the limited price change. Nevertheless, it is possible to jointly learn the price-elasticity of plenty of products with shared parameters. Therefore, we collect abundant feature and historical transaction data of all kinds of products, possibly in different stores. To learn the demand function, a naive approach is using machine learning-based predictive model. It treats price as one of input features and treats sales as label, and fits the model by minimizing factual error. However, the small factual error does not mean a small counterfactual error since the counterfactual distribution may be very different with factual distribution [22, 32]. Besides, when features are correlated with the price (such as historical sales), the importance of price will be largely dominated. Moreover, since
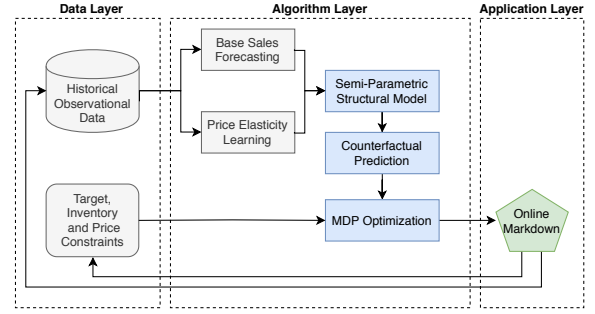
most of the machine learning models are complex or even black-box, it is difficult to reveal the true relationship between price and demand. Therefore, most of ML models have weak interpretability.

To tackle with this problem meanwhile making use of rich features, we propose a novel *data-driven semi-parametric structural model* to capture the price-demand curves. This semi-parametric model establishes a connection between black-box machine learning model and economic model, where the role of ML model is to predict the baseline demand, and the role of economic model is to build a parametric and explainable price-demand relationship, namely price elasticity. To learn price elasticity of individual product, we make use of the shared information among products and propose a multi-task learning algorithm. Based on this framework, the learned elasticity is stable and the counterfactual prediction is reliable.

Using the counterfactual prediction results, the aim of price optimization is to choose the best discount under some business constraints to maximize the overall profit. Note that a fresh retail usually has many stores in a region. For example, Freshippo has more than 50 stores in Shanghai. To avoid price discrimination, the discounts of the same product in different stores within the same region should be all equal. Therefore, to optimize the discount price, we need to take all stores in a region into consideration. Besides, since both demand and inventory of a product would be changed with different stages of product life time horizon, the retailer would like to take *dynamic pricing* strategy for markdowns. Moreover, note that the forecasting results obtained by counterfactual prediction will inevitably have randomness as the variance and bias occurred in the learning procedure. To improve the robustness of the algorithm, it is better to model the demand uncertainty. For these reasons, we split the life cycle of a product into multiple periods and optimize the discount price of each period to maximize the overall profit. We present a *multi-period joint price optimization* formulation using Markov decision process. Since the number of the state grows exponentially with the number of stores, traditional dynamic programming method suffers from the curse of dimensionality. To solve this problem, we develop an efficient two-stage algorithm. It reduces the time complexity from exponential to polynomial. To demonstrate the effectiveness of our approach, we present both offline experiment and online A/B testing at Freshippo. It is shown that our approach has remarkable improvements compared to the manual pricing strategy.

The main contributions are summarized as follows:

- We propose a new learning-and-pricing framework based on counterfactual prediction and multi-period optimization for markdowns of perishable goods, especially for e-commerce fresh retails, as shown in Figure 2.
- We propose a *semi-parametric* structural model that taking advantage of both <mark>predictability and interpretability</mark> for predicting counterfactual demand, and present a *multi-period* dynamic pricing approach which takes the demand uncertainty into consideration, and finally design a very efficient two stage pricing algorithm to solve it.
- We successfully apply the proposed pricing algorithm into the real-world e-commerce fresh retail. The results show great advantage.

2 RELATED WORK

This paper addresses the important data science problem, markdown optimization, which focuses on the dynamic pricing problem where the price of a perishable product is adjusted over its finite selling horizon. It is a variant of revenue management (RM), and has been studied in areas of marketing, economics, and operations research [29]. Most of studies address the static price optimization problem using descriptive models [4, 5, 33]. We focus on multi-period dynamic pricing optimization with demand uncertainty using the prescriptive model.

In the literature, some researchers have developed pricing decision support tools for retailers. For example, a multiproduct pricing tool is implemented for fast-fashion retailer Zara for its markdown decisions during clearance sales [6]. For recommending promotion, the researchers present randomized pricing strategies by incorporating some new features into electronic commerce [34]. In [10], the researchers develop a multiproduct pricing algorithm that incorporates reference price effects for Rue La La to offer extremely limited-time discounts on designer apparel and accessories.

In the context of machine learning, some regression methods have been applied for demand learning. Most of existing works are based on linear regression with different linear demand models for its ease of use and interpretation, such as strict linear, multiplicative, exponential and double-log model [17]. Other works consider the market share of products and use choice model to model demands of products, such as multinomial logit model, nested logit model [21]. Some researchers use semi- and non-parametric models for demand learning. In [24], the authors use SVM to evaluate retailer's decisions about temporary price discounts. In [16], the authors use multilayer perceptrons to estimate store-specific sales response functions. However, these methods can not work well when the price of a product is seldom changed in the historical data, which is common in real-world scenario.

Some recent approaches use machine learning methods for causal inference, such as trees [1, 2] and deep neural networks [22, 32]. But these methods are designed for <mark>binary treatment</mark>. In [13], the authors presents a deep model for characterize the relationship between <mark>continuous treatment</mark> and outcome variables in the presence of instrument variables (IV). However, the IV itself is hard to be identified, which limits the scope of its application.
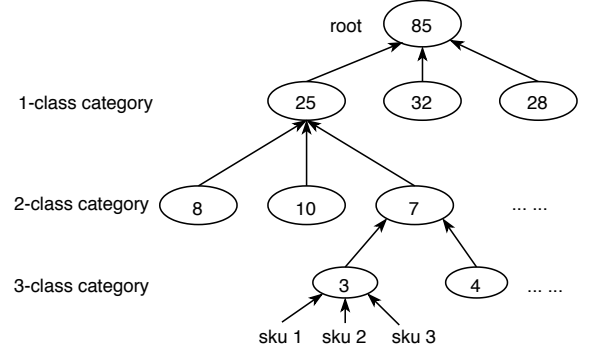
Figure 3: An example of the data aggregation structure by categories. The number in a circle represents the number of the observed data aggregated in that category.

In the literature of price optimization, some researchers consider the large scale pricing problem of multiple products and propose algorithms based on network flow [18], semi-definite programming relaxation [19] and robust quadratic programming [35]. In [36], the authors consider pricing problem in multi-period setting, and use robust optimization to find adaptable pricing policy. However, this model is designed for monopoly and oligopoly and is not suitable for e-commerce fresh retail scenario.

3 PROBLEM FORMULATION

Let us consider an e-commerce retail who wants to sell out N products by markdowns. These products are possibly sold in different stores in a region. Our task is to offer the optimal discount price for these products to maximize the overall profit of all stores.

Instead of using the absolute value of price, we use the relative value, i.e. the percentage of retail price $d \in [0, 1]$, as the decision variable (treatment/intervention) of markdowns, as the range of the percentage discount is fixed and finite while the price itself is infinite. Thus, we can jointly learn price elasticity of different products who have different price magnitudes. Let us denote Y_i^{obs} as the average sales of product i at the discount d_i in markdown channel in the past days. We collect a set of observable covariate features $x_i \in \mathbb{R}^n$, including categories, holidays, event information, inherent properties and historical sales of products and shops, etc. In particular, we denote categorical feature as $L_i \in \{0, 1\}^m$, and assume there are three level categories.

The proposed pricing algorithm consists of two steps: counterfactual prediction and price optimization. In next two sections, we present these two steps respectively.

4 COUNTERFACTUAL PREDICTION

The key of pricing decision making is to accurately predict the demand of products at different discount prices. The difficult is that these prices may never be observed before. In other words, they are counterfactual [15, 30]. The aim of counterfactual prediction is to predict the expectation of demand/sales $\mathbb{E}[Y_i|\text{do}(d_i), x_i]$ under the intervention d_i and the condition $X = x_i$, where the do($\cdot$) is the do operator [26].

4.1 Semi-Parametric Structural Model

In this subsection, we present a semi-parametric structural model to predict the counterfactual demand. It works under the common simplifying assumption of "no unmeasured confounding" [14]. Specifically, we make the following assumption:

ASSUMPTION 1 (UNCONFOUNDEDNESS). *The treatment d_i is conditionally independent with potential outcomes $Y(do(d_i))$ given the observed features x_i, i.e., $d_i \perp\!\!\!\perp Y_i(do(d_i))|x_i$.*

This ensure the identifiability of the casual effect when the complete data on treatment d_i, outcome Y_i and covariates x_i are available. However, in the real-world scenarios, it is almost impossible to collect the complete data for a single product i, especially when treatment d_i is continuous. In most cases, the discount on a product only has one or two different values in the historical transaction data. Therefore, it is impossible to fit the price-demand curve and compute individual causal effect using the observational data of a single product.

To solve this problem, we use the data aggregation technique. We aggregate data of all products by using the *category* information and learn the causal effect of each product *jointly*. As shown in Figure 3, a high-level category has many low-level categories, and a lowest-level category has many different products/SKUs (stock keeping unit). Note that a higher level category has more SKUs, but the difference and variance between SKUs also become larger.

Borrowing the ideas from marginal structural model [31], we propose a semi-parametric structural model to learn individual casual effect. In detail, we assume the outcome Y_i is structurally determined by d_i, x_i and L_i as

$$\mathbb{E}[\ln(Y_i/Y_i^{nor})|d_i, L_i] = g(d_i; L_i, \theta) + h(d_i^o, x_i) - g(d_i^o; L_i, \theta), \quad (1)$$

where Y_i^{nor} is the average sales of product i in normal channel in recent weeks, and d_i^o is the average discount of product i in the markdown channel in recent weeks. We treat Y_i^{nor} as normalized factor, and the quantity Y_i/Y_i^{nor} represents the ratio of the sales with a discount price in markdown channel to the sales with the retail price in normal channel. This quantity is much more useful compared with the absolute value of sales, since sales may have different magnitude with different products.

The function $g(d_i; L_i, \theta)$ is the parametric price-elasticity model of product i and its parameter $\theta \in \mathbb{R}^{m+1}$ represents the price elasticity vector, which is shared by all products. The function $h(d_i^o, x_i)$ is nonparametric preditive model whose role is to predict the log ratio of sales at the base discount d_i^o. Note that let $d_i = d_i^o$, we have

$$\mathbb{E}[\ln(Y_i/Y_i^{nor})|d_i^o] = h(d_i^o, x_i). \quad (2)$$

Model (1) is semi-parametric model since it consists of the parametric model $g(\cdot)$ and the non-parametric model $h(\cdot)$. It provides both high interpretability through parametric econometric model and high predictability through nonparametric machine learning model. We respectively present these two models in the following.

4.2 Base Sales Forecasting

The fundamental building block $h(d_i^o, x_i)$ aims to forecast the sales at base discount d_i^o using the observational data $\{x_{i,t}, Y_{i,t}^{obs}\}$, where t represents the index of data points. A naive approach may treat the discount as one of input variables, and train a predictive model using observational data, then predict the counterfactual outcome by varying the discount, i.e., $E[Y_i|do(d_i)] = h(d_i, x_i)$. However, this method does not work. Because transactional features, such as historical sales, are affected by the historical discount. If we do an intervention on the discount d_i, i.e. varying the discount, we also need to change the relevant historical features $x_i(d_i)$ which is impossible. To resolve this problem, we dissociate the treatment from the sales-related features, and divide model into two parts as presented in model (1). We emphasize that the sales forecasting module only predicts the sales at an average of historical discounts d_i^o, i.e.,

$$h(\cdot) \leftarrow \{x_{i,t}, Y_{i,t}^{obs}/Y_{i,t}^{nor}\}_{t=1,...,T, i=1,...,N},$$
$$\ln(Y_i^o/Y_{i,T+1}^{nor}) = h(d_i^o, x_{i,T+1}), i = 1, \ldots, N. \quad (3)$$

We call (d_i^o, Y_i^o) as base discount and base sales pair of product i. Many fundamental non-parametric predictive model can be applied to learn $h(\cdot)$, such as deep neural networks [20] and boosting tree model [7].

4.3 Price Elasticity Model

In the second part, we aim to learn the average price sensitivity of customers to products. We propose a double-log structural nested mean model,

$$g(d_i; L_i, \theta) = \mathbb{E}[\ln(Y_i/Y_i^{nor})] = (\theta_1 + \theta_2^T L_i) \ln d_i + c, \quad (4)$$

where $\theta_2 \in \mathbb{R}^m$, $\theta = [\theta_1, \theta_2^T]^T$, c is an intercept parameter and L_i is a compound of three one-hot variables, namely

$$L_i = [\underbrace{0, \ldots, 1, 0}_{category1}, \underbrace{0, 1, \ldots, 0}_{category2}, \underbrace{0, \ldots, 1, 0}_{category3}]^T. \quad (5)$$

Since the coefficient of treatment d_i is modified by L_i, we call L_i as effect modifiers. By exponential transformation, this model can also be written as a constant price elasticity model, i.e.,

$$Y_i = Y_i^{nor} e^c d_i^{\theta_1 + \theta_2^T L_i}, \forall i, \quad (6)$$

where $\theta_1 + \theta_2^T L_i$ is price elasticity of demand, which is structured by three different level categories. In other words, the price elasticity of each individual SKU is a summation of a common coefficient θ_1 and the elasticity of each category. As we have stated, the aggregated data within high level category has large sample size but also has large variation, while the aggregated data within low level category has small variation but also has small sample size. Instead of learning elasticity within a single category, this model simultaneously learns price elasticity of all category-levels which is more flexible.

To estimate the price elasticity of demand, we can use the mean squared error criterion with entire samples. However, in the real world e-commerce retail scenario, the data is generated by streaming. A better alternative is to update the parameters in an online fashion. Therefore, based on recursive least squares, we update the parameters by minimizing the following objective function:

$$\min_{\theta, c} \sum_{i=1}^{N} \sum_{j=1}^{t} \tau^{t-j} || \ln \frac{Y_{i,j}}{Y_{i,j}^{nor}} - \theta^T \hat{L}_i \ln d_{i,j} - c ||_2^2 + \lambda ||\theta||_2^2, \quad (7)$$

Period				1	2	3	4	5	6
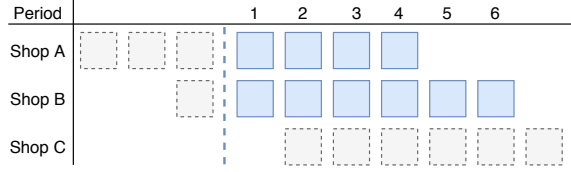
Figure 4: An example of the activated shops and index of multiple periods. The dotted blue vertical line represents the current time. The dotted grey blocks represent the inactive days, and solid blue block represent the activate days.

where $\hat{L}_i = [1, L_i^T]^T$ is the augmented variable and $\lambda > 0$ is a pre-defined regularization coefficient, $0 < \tau \leq 1$ is a forgetting factor which gives exponentially less weight to older samples. The optimal solution of (7) can be represented by two sufficient statistics, which are then can be easily updated in an online fashion.

4.4 Counterfactual Demand Prediction

Suppose we have obtained the observational data from time 1 to $t-1$, our aim become to predict the counterfactual demand of product i of time t with different discounts. Let us denote the base discount and base sales of product i predicted by the sales forecasting module as $(d_{i,t}^o, Y_{i,t}^o)$, and denote the estimated parameter of price elasticity model as $\hat{\theta}_t$. Substituting (3) and (4) into semi-parametric model (1), we can predict the counterfactual demand as

$$\ln Y_{i,t}(d_i) = \hat{\theta}_t^T \hat{L}_i(\ln d_i - \ln d_{i,t}^o) + \ln Y_{i,t}^o. \tag{8}$$

Varying the discount from 0 to 1, we can predict the counterfactual sales using (8). In the next section, we use these predicted results for optimizing discount price.

5 MULTI-PERIOD PRICE OPTIMIZATION

Once obtaining counterfactual demands, we can formulate a price optimization problem to maximize the overall profit. We neglect the substitution effect and assume that different products are independent. Thus, we can consider price optimization for only one product and omit the subscript i for notational simplicity.

Suppose there are $|\mathcal{J}|$ shops in a region. Each shop $j \in \mathcal{J}$ has B_j items of product A in stock and they want to sold out in T_j days. If it is not sold out, it will be thrown away for ensuring the freshness of goods and leads to the economic loss. We treat one day as a period, and the number of total periods is $T_{\max} := \arg\max_j T_j$. In different periods, product A has possibly different demands. Taking the targets of all shops into consideration, we aim to optimize the price jointly and dynamically. An example of the activated shops and the index of periods is illustrated in Figure 4.

Let us denote the retail price of product A as p_0. It is the price sold in the normal channel and is fixed. Let us denote d_t as the percentage discount at period t, and the corresponding discount price is $p_t = p_0 d_t$. We use the sales forecasting algorithm presented in the subsection 4.2 to predict the sales in normal channel for all shops and all periods, denoted as $Z_{jt}, \forall j, t$. We use the semi-parameteric model (8) to predict the counterfactual sales in markdown channel in period j with the discount d_t, denoted as $Y_{jt}(d_t)$.

Since minimizing the waste loss is also a target of the retailer, we also take it into consideration. Together with the target of maximizing the overall profit, we formulate the following price optimization problem:

$$\max_{d_1,\dots,d_T} \sum_{j \in \mathcal{J}} \left(\sum_{t=1}^{T_j} p_0 d_t Y_{jt}(d_t) - w_j [B_j - \sum_{t=1}^{T_j} (Y_{jt}(d_t) + Z_{jt})]^+ \right)$$

$$\text{s.t.} \sum_{t=1}^{T_j} (Y_{jt}(d_t) + Z_{jt}) \leq B_j, j \in \mathcal{J},$$

$$lb_{jt} \leq p_t \leq ub_{jt}, t = 1,\dots,T_j, j \in \mathcal{J}, \tag{9}$$

where $w_j > 0$ is the weight of waste loss and $[\cdot]^+$ is the non-negative operator. The variables lb_{jt} and ub_{jt} are respectively minimum and maximum value of discount for shop j at period t. Since both Z_{jt} and B_j are independent with decision variables $\{d_1,\dots,d_T\}$, we can simplify (9) as

$$\max_{d_1,d_2,\dots,d_T} \sum_{j \in \mathcal{J}} \sum_{t=1}^{T_j} (p_0 d_t + w_j) Y_{jt}(d_t)$$

$$\text{s.t.} \sum_{t=1}^{T_j} (Y_{jt}(d_t) + Z_{jt}) \leq B_j, j \in \mathcal{J} \tag{10}$$

$$lb_{jt} \leq d_t \leq ub_{jt}, t = 1,\dots,T_j, j \in \mathcal{J},$$

where the first constraint describes the interaction among decision variables $\{d_1, d_2, \cdots, d_T\}$.

## 5.1 MDP Model

The continuous optimization w.r.t discount is indeed not convex. However, in this paper, we do not optimize the discount directly. Considering that the candidates of the discount is finite, we transform it into discrete optimization problem instead. Moreover, note that the learned price-sales function $Y_{jt}(d_t)$ and the predicted normal sales Z_{jt} inevitably have the randomness due to the variance and bias occurred in the learning and prediction process. In other words, the parameters in the optimization problem (10) have uncertainty. Therefore, we use Markov decision process (MDP) to model this decision-making problem.

Without loss of generality, we assume the actual sales of shop j in markdown channel and normal channel are respectively a_{jt}^y and a_{jt}^z at period t. Let us define the state as the inventory in stock, i.e., the state $s_{j,t}$ represents the inventory of product A at shop j at the beginning of period t. The initial state of shop j is equal to the target sales or equivalently initial inventory B_j. The state $s_{j,t+1}$ is equal to the state $s_{j,t}$ subtracting both sales in the markdown channel and normal channel at period t, and the state $s_{j,T}$ will not be negative at the last period. Mathematically,

$$s_{j,1} = B_j,$$
$$s_{j,t+1} = s_{j,t} - a_{jt}^y - a_{jt}^z, \ t < T_j, \tag{11}$$
$$s_{j,t+1} = 0, \ T_j \leq t \leq T_{\max}.$$

Note that the value of state is non-increasing, i.e., $s_{j,1} \geq \cdots \geq s_{j,T_j+1} = 0$. We define a set of finite candidates of the percentage
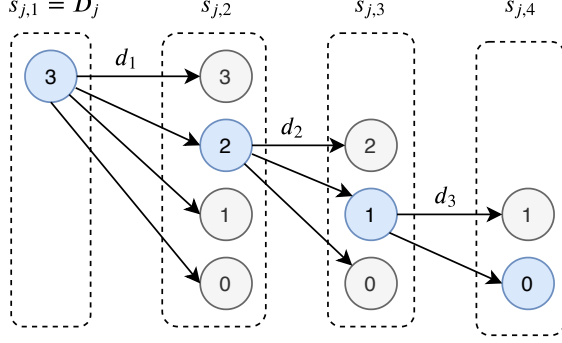
Figure 5: An example of the Markov decision process of shop j with 4 states. Circles represent the states and the arrows represent the possible transition directions with action d_t. For simplicity, we only draw the arrows of blue circles.

discount as $\mathcal{D} = \{d^1, d^2, \ldots, d^M\}$. This discount set is the set of actions that the decision maker chooses from.

To modeling the *uncertainty* of sales forecasting algorithm, we dig into the historical data of the predicted sales and actual sales, and find that the sales distribution mostly follows *Poisson* distribution. In detail, we assume the sales distribution of the markdown channel follows *Poisson* distribution with the parameter $Y_{jt}(d_t)$, and sales distribution of the normal channel follows *Poisson* distribution with parameter Z_{jt}. Since the expectation of the Poisson distribution is equal to its parameter, we have implicitly assumed that $Y_{jt}(d_t)$ and Z_{jt} are unbiased estimation of a_{jt}^y and a_{jt}^z, respectively.

Let us define $a_{jt} := a_{jt}^y + a_{jt}^z$ as the total sales of product A in both channels. Since a_{jt} can not be greater than inventory $s_{j,t}$, we have

$$P(a_{jt}) = \begin{cases} Poi(a_{jt}|\lambda_{jt}), & 0 \le a_{jt} < s_{j,t} \\ 1 - Q(s_{j,t} - 1, \lambda_{jt}), & a_{jt} = s_{j,t}, \end{cases} \quad (12)$$

where $\lambda_{jt} := Y_{jt}(d_t) + Z_{jt}$, $Poi(\cdot)$ is Poisson distribution and $Q(\cdot, \cdot)$ is the regularized Gamma function.

Using (11) and (12), we can obtain the *state-transition probability* that action d_t in state $s_{j,t}$ will lead to $s_{j,t+1}$:

$$P(s_{j,t+1}|s_{j,t}, d_t)$$
$$= \begin{cases} Poi(s_{j,t} - s_{j,t+1}|Y_{jt}(d_t) + Z_{jt}) & 0 < s_{j,t+1} \le s_{j,t} \\ 1 - Q(s_{j,t} - 1, Y_{jt}(d_t) + Z_{jt}) & s_{j,t+1} = 0. \end{cases} \quad (13)$$

The *expected immediate reward* received after transitioning from state $s_{j,t}$ to state $s_{j,t+1}$ due to action d_t is

$$R(s_{j,t}, d_t, s_{j,t+1}) = (p_0 d_t + w_j)[s_{j,t} - s_{j,t+1} - Z_{jt}]^+. \quad (14)$$

The optimization problem (10) now becomes to choose a policy $\pi(\cdot)$ that will maximize the total rewards,

$$\sum_{j \in \mathcal{J}} \sum_{t=1}^{T_j} R(s_{j,t}, d_t, s_{j,t+1}), \quad d_t = \pi(s_t), \quad (15)$$

where $s_t := \{s_{1,t}, \ldots, s_{J,t}\}$, and d_t is the action given by policy $\pi(s_t)$. To illustrate this process vividly, we give an example of Markov decision process with four states in Figure 5.

5.2 Two-stage Algorithm

Given state transition probability $P(\cdot)$ in (13) and reward function $R(\cdot)$ in (14) for an MDP, we can seek the optimal policy through dynamic programming with Bellman equation. However, since the dimension of the state space grows exponentially as the increase of the number of the shops, a primitive backward induction method is very time-consuming. We propose a two-stage optimization algorithm to simplify dynamic programming, consisting of a separate backward induction and a joint optimization.

5.2.1 Separate backward induction. The Q function of each shop j is updated separately using greedy policy by backward induction from period T_j to period 2, i.e.,

$$Q(s_{j,t}, d_t) = \sum_{s_{j,t+1}=0}^{s_{j,t}} P(s_{j,t+1}|s_{j,t}, d_t)(R(s_{j,t}, d_t, s_{j,t+1})$$
$$+ \max_{d_{t+1} \in \mathcal{D}} Q(s_{j,t+1}, d_{t+1})), t = T_j, \ldots, 2, j \in \mathcal{J}, \quad (16)$$

where the initial value is $Q(s_{j,T_j+1}, \cdot) = 0, s_{j,T_j} = 0, \forall j$. This is the bellman equation for each single shop, and it can be updated by dynamic programming efficiently by finding substructure of computation.

5.2.2 Joint optimization. We update the Q function using the backward induction separately for each shop until the last step. In the last step (i.e., at period 1), we can *jointly* optimize the SKU price since there is only one state, i.e., $s_1 = \{B_1, B_2, \cdots, B_J\}$. The joint optimization equation can be written as follows,

$$d_1^* := \arg \max_{d_t \in \mathcal{D}} \sum_{j \in \mathcal{J}} Q(s_{j,1} = B_j, d_1), \quad (17)$$

where

$$Q(s_{j,1} = B_j, d_1) = \sum_{s_{j,2}=0}^{s_{j,1}} P(s_{j,2}|s_{j,1}, d_1)$$
$$\cdot (R(s_{j,1}, d_1, s_{j,2})d + \max_{d_2 \in \mathcal{D}} Q(s_{j,2}, d_2)). \quad (18)$$

Thus, we obtain the optimal price $p_1^* = p_0 d_1^*$ and the expected total reward $V(s_1) = \sum_{j \in \mathcal{J}} Q(B_j, d_1^*)$. Since the pricing algorithm is performed daily as soon as the new data arrived for each new day, it will recompute the optimal price for the first period which is satisfied the price constraint, i.e., the discount price of all shops are equal. Therefore, it is not necessary that the obtained prices of different shops are not equal from period 2 to the last period.

For clarity, the pricing algorithm for markdowns in fresh retail is summarized in Algorithm 1, where we refill the subscript i of all variables to represent product i for completeness.

5.3 Time Complexity

In this subsection, we analyze the time complexity of the proposed two-stage algorithm. There are at most $B_j^2 * M$ iterations to compute $Q(s_t, d_t)$, where M is the number of actions. So the time complexity of two-stage algorithm is $O(T_{\max} * M * B_{\max}^2 * |\mathcal{J}|)$, where $B_{\max} := \arg \max_j B_j$. In comparison, note that the state space for a primitive backward induction method is vector space, and the number of state is $\prod_{j \in \mathcal{J}} B_j$. The time complexity for this method is $O(T_{\max} * M * B_{\max}^{2|\mathcal{J}|})$, which grows exponentially with the increase of the

Algorithm 1 Pricing algorithm for markdowns of e-commerce retails

Input: $\lambda, \tau, \omega, \boldsymbol{x}_{i,\cdot}, Y_{i,\cdot}^{obs}, Y_{i,\cdot}^{nor}, d_{i,\cdot}^{o}, \forall i.$
Output: $p_{i,t}^{*}, \forall i, \forall t.$

1: Pretrain $h(\cdot)$ and $g(\cdot)$ using historical observational data.
2: **for** $t = 1, 2, \ldots$ **do**
3:    Receive new data $\boldsymbol{x}_{i,t}, Y_{i,t}^{obs}, Y_{i,t}^{nor}, d_{i,t}^{o}, \forall i.$
4:    Update $h(\cdot)$ and predict $(d_{i,t}^{o}, Y_{i,t}^{o})$ using (3).
5:    Update $\hat{\boldsymbol{\theta}}_{t}$ using (7).
6:    **for** $i = 1, \ldots, N$ **do**
7:       **for** $k = 1, 2, \ldots, T_{i,\max} - t + 1$ **do**
8:          Predict Z_{ijk} and $Y_{ijk}(d_{ik}), \forall d_{ik} \in \mathcal{D}_{i}.$
9:       **end for**
10:        Optimize $d_{i,1}^{*}$ using (16) and (17).
11:    **end for**
12:    $p_{i,t}^{*} = p_{i,0} d_{i,1}^{*}.$
13: **end for**

number of the shops. It is easy to see that our algorithm is much more efficient than the traditional dynamic programming algorithm when $|\mathcal{J}|$ is large. In practice, our algorithm can be further pruned by reducing redundant computation.

5.4 Extension

The *counterfactual prediction and optimization* framework is quite general and can be applied into other scenarios of e-commerce platform:

- In fresh retail scenario, this approach can also be applied for the markdown of daily-fresh goods, which has only one day shelf life. It aims to boost sales by cutting the price before the end of the day, such as only two hours left.
- In the marketing scenario, it can be applied for personalized coupon assignment task, which aims to maximize the overall Return on Investment (ROI) of the platform by assigning the different coupons to different users with budget constraint.
- In the customer service scenario, it can be applied for personalized compensation pay-outs task, which aims to maximize the satisfaction rate of customers by offering the optimal pay-outs for users with the budget constraint.

In above three examples, the treatment is price, coupon and compensation pay-outs respectively, the outcome is the sales of product, the conversion rate and satisfaction rate of the users respectively, and the optimization objective is overall profit, ROI and satisfaction rate respectively. To solve these problems, we first build the causal relationship between treatment and outcome based on semi-parametric model, and then optimize the objective with budget or inventory constraints. It is worth noting that we have presented a marketing budget allocation framework for the second scenario in our previous work [37]. While, in this paper, the proposed counterfactual prediction-based framework is more general and can be applied into more scenarios.

6 EXPERIMENT

To evaluate the performance of the proposed approach, we apply the markdown pricing algorithm in a real-world e-commerce fresh retail - Freshippo. To reduce cost and stimulate consumption, Freshippo will give a discount for the product whose sales performance is not satisfactory. It will be sold in markdown channel, where customers can buy goods by discount if their total purchase reached a certain amount. Our aim is to help the retailer of Freshippo to decide which optimal discount price of a product to be set to maximize the overall profit.

6.1 Offline Experiment

To optimize the price with business constraints, the first step and the key step is to learn the price-demand curve as described in Section 4. We use the offline transaction data of Freshippo to train and evaluate the proposed semi-parametric structural demand model. We collect observational data in markdown channel over more than 100 fresh stores across 10 different cities in China over 6 months. There are about 11000+ SKUs in total (we omit the exact number for privacy-preserving). The detail of feature extraction is provided in Supplement.

Learning model. Using the extracted features, we use XGBoost [7] as our base sales forecasting model. The deep neural network is also recommended when there are sufficient data samples for training [3, 12]. Using the predicted base-discount and sales as inputs, we solve the objective (7) by recursive least squares algorithm, and the price elasticity θ is updated in the recursion fashion. In the Freshippo scenario, the price elasticity is daily updated once the new transaction data is collected and features are automatically extracted.

Evaluation metric. To evaluate the predictive ability of our semiparametric model, we predict the sales of the next day in markdown channel with the actual discount of that day. The results are measured in Relative Mean Absolute Error (RMAE), defined as:

$$\text{RMAE} = \frac{\sum_{i=1}^{N} |Y_i - \hat{Y}_i|}{\sum_{i=1}^{N} Y_i}, \tag{19}$$

where N is the number of the test samples, Y_i is the ground truth sales and $\hat{Y}_i$ is the predicted sales.

Parameter settings. To tun the model hyperparameters, we split data into training, validation and test data according to the time: first 65% for training, next 15% for validation and last 20% for testing. For price elasticity model, we empirically set the forgetting factor $\tau = 0.95$, the regularization parameter $\lambda = 0.5$.

Comparison algorithm. We compare the proposed model with the classical boost tree model and deep model.

- XGBoost. This powerful boost tree model has been applied in many industrial applications. For counterfactual prediction task, we treat the discount variable as one of its input features, and predict the outcome by varying discount with other features fixed. For the sake of fairness, its hyperparameters are set the same as our base sales forecasting model.
- DeepIV [13]. This method use deep model to characterize the relationships between treatment and outcome in the
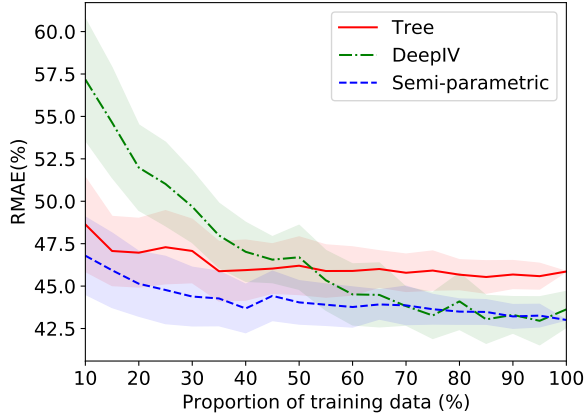
Figure 6: The comparisons between tree model, deep model, and our semi-parametric model under different proportion of training data.

presence of instrument variables. We use the average price of third-level category as the instrument variables, and use Gaussian distribution for continuous treatment.

Results: We first evaluate the prediction error of each algorithm with different proportion of training data. As shown in Figure 6, the deep model has poor performance with small training data. The tree model is relatively insensitive with the data size but its performance is poor even when there are more data available. Our proposed semi-parametric model achieve the best RMAE performance and its variance is reduced with the increase of the number of training data. To further illustrate the interpretability of the proposed model, we plot four price-sales curves of randomly chosen SKUs in a random store in Figure 7. The curve learned by the tree model has unpredictable jitter, and its outcome keeps unchanged in a large range of discount (0.5-0.6, 0.8-1.0). The curve learned by deepIV is almost a line, which indicates the sales is independent with the price. Both tree and deep model can not correctly reveal the price-sales relationship and the inferred results are not credible. While, our semi-parametric model is much smoother and reveals a monotonous relationship between price and sales. The variation between close prices is small, which is consistent with the intuition.

6.2 Online A/B Testing

We evaluate our markdown approach in a online fresh retail, Freshippo, as shown in Figure 1. Traditionally, the retailer of Freshippo uses a manual pricing strategy that chooses an empirical discount, such as 30% off or 50% off. Although the manual pricing strategy is simple, it is the fruit of experience of operations for many years and it works in most cases. The target products, which are required for markdowns, are provided by the inventory control system of Freshippo. The target sales and min-max price constraints are given by fresh operational staff. Once got the signal of markdowns, the intelligence marketing system of Freshippo will call our pricing algorithm to offer the optimal discount. Our algorithm has been applied in the fresh stores in four regions, Beijing, Shanghai, Hangzhou and Shenzhou.

Table 1: Target completion rate and GMV improvement

	TCR_{nor}	TCR_{md}	TCR_{total}	GMV_IMP
Operations	34.25%	46.68%	80.93%	101.49%
Ours	38.92%	53.20%	**92.12%**	**118.63%**

For fair comparison, we design the online A/B testing. The products required to be taken a discount are randomly assigned to the operation's manual approach and our markdown approach.

Evaluation metrics. To evaulate the performance of the proposed algorithm, we use the target completion rate (TCR) as the metric, defined as

$$\text{TCR}_{nor} = \frac{\sum_{i=1}^{N} \#\text{SALES}_{i,nor}}{\sum_{j=1}^{N} B_i},$$

$$\text{TCR}_{md} = \frac{\sum_{i=1}^{N} \#\text{SALES}_{i,md}}{\sum_{j=1}^{N} B_i}, \tag{20}$$

where $\#\text{SALES}_{i,nor}$ and $\#\text{SALES}_{i,md}$ denote the sales in the normal channel and the sales in markdown channel, respectively. The target completion rate assesses the effect on clearance of markdowns. Besides, we also evaluate the ratio of Gross Merchandise Volume (GMV) between normal channel and makdown channel, namely

$$\text{GMV\_IMP} = \frac{\sum_{i=1}^{N} d_i p_i \#\text{SALES}_{i,md}}{\sum_{i=1}^{N} p_i \#\text{SALES}_{i,nor}}. \tag{21}$$

This metric evaluates the GMV improvement after markdowns.

Results . The A/B testing is carried out about 2 months. As shown in Table1, the target completion rate of operation's group is about 34.25% before markdown, and this number goes to 80.93% after markdowns. While the target completion rate of our group is about 38.92% before markdown, and this number goes to 92.12% after markdown. It indicates that our approach can better achieve the clearance target than the maunal approach. Besides, the GMV improvement of our pricing algorithm is higher about 17.14% than that of maunal approach. This result shows that our approach can help retailer obtain the higher GMV/profit.

In summary, our pricing algorithm is easy to explan as it has interpretable price-sales curve, and it is intelligent as it can automatically offer optimal price with high profit in consideration of complex business constraints, such as inventory constraint.

7 CONCLUSION

In this paper, we present a novel pricing framework for markdowns in e-commerce fresh retails, consisting of counterfactual prediction and multi-period price optimization. Firstly, we propose a data-driven semi-parametric structural demand model, which has both predictability and interpretability. The proposed demand model reveals the relationship between price and sales and can be used for predicting counterfactual demand with different prices. The proposed counterfactual model has a lower model complexity and a clear causal structure, therefore it is much more interpretable than traditional ML and deep model. Secondly, we take the demand uncertainty into consideration, and present a dynamic pricing strategy of perishable products in a multi-period setting. An efficient
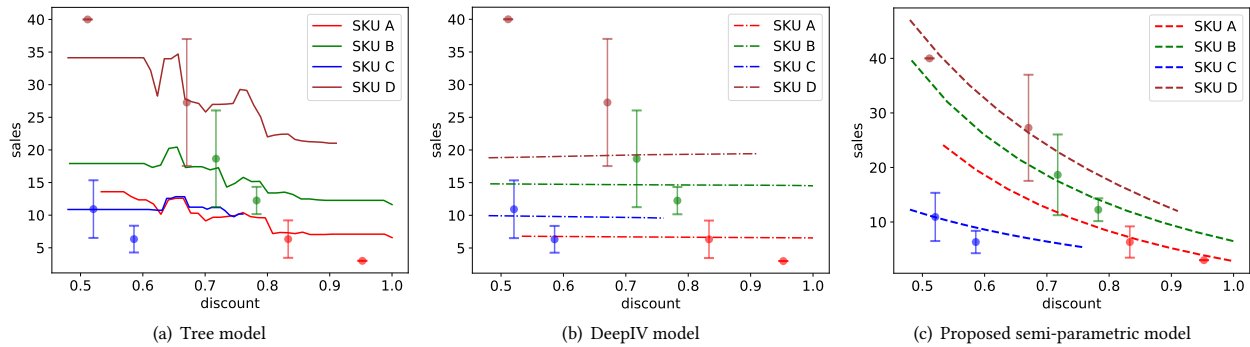
Figure 7: Price-sales curves of four randomly chosen SKUs. The error bar denotes the mean and the standard deviation of historical observed sales.

two-stage algorithm is proposed for solving the MDP problem. It reduces the time complexity from exponential to polynomial. Finally, we apply our framework into a real-world e-commerce fresh retail. Both offline experiment and online A/B testing show the effectiveness of our approach.

REFERENCES

[1] Susan Athey and Guido Imbens. 2016. Recursive partitioning for heterogeneous causal effects. *Proceedings of the National Academy of Sciences* 113, 27 (2016), 7353–7360.
[2] Susan Athey, Julie Tibshirani, Stefan Wager, et al. 2019. Generalized random forests. *The Annals of Statistics* 47, 2 (2019), 1148–1178.
[3] Yoshua Bengio, Aaron Courville, and Pascal Vincent. 2013. Representation learning: A review and new perspectives. *IEEE transactions on pattern analysis and machine intelligence* 35, 8 (2013), 1798–1828.
[4] Robert C Blattberg and Edward I George. 2010. Shrinkage estimation of price and promotional elasticities: Seemingly unrelated equations. In *Perspectives On Promotion And Database Marketing: The Collected Works of Robert C Blattberg*. World Scientific, 145–156.
[5] Ruth N Bolton and Venkatesh Shankar. 2003. An empirically derived taxonomy of retailer pricing and promotion strategies. *Journal of Retailing* 79, 4 (2003), 213–224.
[6] Felipe Caro and Jérémie Gallien. 2012. Clearance pricing optimization for a fast-fashion retailer. *Operations Research* 60, 6 (2012), 1404–1422.
[7] Tianqi Chen and Carlos Guestrin. 2016. Xgboost: A scalable tree boosting system. In *Proceedings of the 22nd acm sigkdd international conference on knowledge discovery and data mining*. ACM, 785–794.
[8] Adam N Elmachtoub and Paul Grigas. 2017. Smart" predict, then optimize". *arXiv preprint arXiv:1710.08005* (2017).
[9] Zhi-Ping Fan, Yu-Jie Che, and Zhen-Yu Chen. 2017. Product sales forecasting using online reviews and historical sales data: A method combining the Bass model and sentiment analysis. *Journal of Business Research* 74 (2017), 90–100.
[10] Kris Johnson Ferreira, Bin Hong Alex Lee, and David Simchi-Levi. 2016. Analytics for an online retailer: Demand forecasting and price optimization. *Manufacturing & Service Operations Management* 18, 1 (2016), 69–88.
[11] Marshall Fisher, Santiago Gallino, and Jun Li. 2018. Competition-based dynamic pricing in online retailing: A methodology validated with field experiments. *Management Science* 64, 6 (2018), 2496–2514.
[12] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. 2016. *Deep learning*. MIT press.
[13] Jason Hartford, Greg Lewis, Kevin Leyton-Brown, and Matt Taddy. 2017. Deep IV: A flexible approach for counterfactual prediction. In *Proceedings of the 34th International Conference on Machine Learning-Volume 70*. JMLR. org, 1414–1423.
[14] Miguel A Hernan and James M Robins. 2010. Causal inference.
[15] Jake M Hofman, Amit Sharma, and Duncan J Watts. 2017. Prediction and explanation in social systems. *Science* 355, 6324 (2017), 486–488.
[16] Harald Hruschka. 2006. Relevance of functional flexibility for heterogeneous sales response models: a comparison of parametric and semi-nonparametric models. *European journal of operational research* 174, 2 (2006), 1009–1020.
[17] Jian Huang, Mingming Leng, and Mahmut Parlar. 2013. Demand functions in decision modeling: A comprehensive survey and research directions. *Decision Sciences* 44, 3 (2013), 557–609.

[18] Shinji Ito and Ryohei Fujimaki. 2016. Large-scale price optimization via network flow. In *Advances in Neural Information Processing Systems*. 3855–3863.
[19] Shinji Ito and Ryohei Fujimaki. 2017. Optimization beyond prediction: Prescriptive price optimization. In *Proceedings of the 23rd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*. ACM, 1833–1841.
[20] Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. 2015. Deep learning. *nature* 521, 7553 (2015), 436.
[21] Hongmin Li and Woonghee Tim Huh. 2011. Pricing multiple products with the multinomial logit and nested logit models: Concavity and implications. *Manufacturing & Service Operations Management* 13, 4 (2011), 549–563.
[22] Christos Louizos, Uri Shalit, Joris M Mooij, David Sontag, Richard Zemel, and Max Welling. 2017. Causal effect inference with deep latent-variable models. In *Advances in neural information processing systems*. 6446–6456.
[23] Shaohui Ma, Robert Fildes, and Tao Huang. 2016. Demand forecasting with high dimensional data: The case of SKU retail sales forecasting with intra-and inter-category promotional information. *European Journal of Operational Research* 249, 1 (2016), 245–257.
[24] María Pilar Martínez-Ruiz, Alejandro Mollá-Descals, MA Gómez-Borja, and José Luis Rojo-Álvarez. 2006. Evaluating temporary retail price discounts using semiparametric regression. *Journal of Product & Brand Management* (2006).
[25] Stephen L Morgan and Christopher Winship. 2015. *Counterfactuals and causal inference*. Cambridge University Press.
[26] Judea Pearl. 2009. *Causality*. Cambridge university press.
[27] Judea Pearl et al. 2009. Causal inference in statistics: An overview. *Statistics surveys* 3 (2009), 96–146.
[28] Georgia Perakis and Anshul Sood. 2006. Competitive multi-period pricing for perishable products: A robust optimization approach. *Mathematical Programming* 107, 1-2 (2006), 295–335.
[29] Robert Lewis Phillips. 2005. *Pricing and revenue optimization*. Stanford University Press.
[30] Mattia Prosperi, Yi Guo, Matt Sperrin, James S Koopman, Jae S Min, Xing He, Shannan Rich, Mo Wang, Iain E Buchan, and Jiang Bian. 2020. Causal inference and counterfactual prediction in machine learning for actionable healthcare. *Nature Machine Intelligence* 2, 7 (2020), 369–375.
[31] James M Robins, Miguel Angel Hernan, and Babette Brumback. 2000. Marginal structural models and causal inference in epidemiology.
[32] Uri Shalit, Fredrik D Johansson, and David Sontag. 2017. Estimating individual treatment effect: generalization bounds and algorithms. In *International Conference on Machine Learning*. PMLR, 3076–3085.
[33] Garrett J van Ryzin. 2005. Models of demand. In *The Oxford Handbook of Pricing Management*.
[34] Jianghua Wu, Ling Li, and Li Da Xu. 2014. A randomized pricing decision support system in electronic commerce. *Decision Support Systems* 58 (2014), 43–52.
[35] Akihiro Yabe, Shinji Ito, and Ryohei Fujimaki. 2017. Robust Quadratic Programming for Price Optimization.. In *IJCAI*. 4648–4654.
[36] Lei Zhang et al. 2006. *Multi-period pricing for perishable products: Uncertainty and competition*. Ph.D. Dissertation. Massachusetts Institute of Technology.
[37] Kui Zhao, Junhao Hua, Ling Yan, Qi Zhang, Huan Xu, and Cheng Yang. 2019. A Unified Framework for Marketing Budget Allocation. In *Proceedings of the 25th ACM SIGKDD International Conference on Knowledge Discovery & Data Mining*. 1820–1830.

A SUPPLEMENT

A.1 Implementation

In order to improve the reproducibility, we present more details about the implementation.

A.1.1 Software versions. The details of programming languages, software packages and frameworks used in the experiments and deployed system are as follow:

- Language: Python 3.6.10, SQL.
- Packages: XGoost 1.3.1, NumPy 1.16.6, Pandas 1.0.3, Tensorflow 1.14.0, DeepIV[2].
- Frameworks: MaxCompute.

A.1.2 Feature extraction. We first extract the features of both products and stores. The raw features includes brand, sku, department, categories, weekend, holiday information, sales channel, promotional event, page views, users views, historical discounts and historical sales, etc. Secondly, we use the data aggregation process to create new features. Specifically, we aggregate the sales with different time period, such as by week and by holiday, and aggregate the sales with different clusters, such brand, category, store, sku, channel of sales. Thirdly, we impute the missing data by using the averaged value or by nearest neighbor matching. Other methods can also be applied, such as EM algorithm, interpolation and matrix completion, etc. Finally, one-hot representation is carried out for the sparse features, such as one-level, second-level, and third-level categories.

A.1.3 Other tips.

- The normalized factor Y_i^{nor} is the average sales of product i by definition. However, this quantity may be not very stable. In practice, the average sales of level-2 or level-3 category that the product belongs to can be used for normalization.
- The division operation is sensitive to the small value. It is better to add a small quantity before do division operation, e.g., $\frac{a}{b} \to \frac{a}{b+1}$.

---

[2]https://github.com/jhartford/DeepIV