Making smart recommendations for perishable and stockout products*

SINAN SEYMEN, Northwestern University, USA

ANNA-LENA SACHS, Lancaster University, United Kingdom

EDWARD C. MALTHOUSE, Northwestern University, USA

Food waste and stockouts are widely recognized as an important global challenge. While inventory management aims to address these challenges, the tools available to inventory managers are often limited and the usefulness of their decisions is dependent on demand realizations, which are not within their control. Recommender systems (RS) can influence and direct customer demand, e.g., by sending personalized emails with promotions for different items. We propose a novel approach that combines the opportunities provided by RS with inventory management considerations. Under the assumption that there is a known set of customers to receive a promotion consisting of k items, we use mixed-integer programming (MIP) to allocate recommended items across customers taking both individual preferences and the current state of inventory with uncertainties into account. Our approach can solve problems with both stochastic supply (inventory and perishability) and demand. We propose heuristics to improve scalability and compare their performance with the optimal solution using data from an online grocery retailer. The goal is to target the right set of customers who are likely to purchase an item, while simultaneously considering which items are prone to expire or be out-of-stock soon. We show that creating recommendation lists exclusively considering user preferences can be counterproductive to users due to possible excessive stockouts. Similarly, focusing only on the retailer can be counterproductive to retailer sales due to the number of expired products that can be considered lost income. We thus avoid the loss of customer goodwill due to stockouts and reduce waste by selling inventory before it expires.

CCS Concepts: • **Information systems** → **Personalization**; **Recommender systems**.

Additional Key Words and Phrases: Recommender systems, mixed-integer programming, perishability, multi-objective optimization

1 INTRODUCTION

With about one-third of food lost or wasted, avoiding food waste has become a global challenge due to its environmental and economic impact [9]. One out of four calories of food intended for consumption was wasted in 2009 [38]. Additionally, 28% of the items sold by retailers are wasted because they expire without other flaws [35]. Reasons for waste include spillage, spoilage, and a significant decrease in the quality of the product. Wasted food emits significant amounts of greenhouse gas to the air [50], deforests the Amazon [40], and causes significant negative effects on the climate with very little in return.

Retail food waste occurs when supply exceeds demand. A simple solution is for retailers to order less and thus decrease supply. This, however, increases the number of stockouts and consequently reduces customer goodwill [7]. While the stockout problem has been studied for many years [30], it has become more critical recently because of supply chain disruptions due to extreme weather events, port congestion, labor relations [24], COVID-19 lockdowns, manufacturing delays and human errors (e.g., Suez Canal being blocked) [54]. Increasing inventory can cause higher holding costs and lower clearance sale pricing, which results in lower overall revenue for the retailer [49].

Although the decision maker has a notion of what to expect in the future, customer demand and whether they would buy a recommended item is not deterministic. Consequently, inventory levels are also stochastic since the decision

Presented at the MORS workshop held in conjunction with the 16th ACM Conference on Recommender Systems (RecSys), 2022, in Seattle, USA.

Authors' addresses: Sinan Seymen, Northwestern University, Evanston, Illinois, USA; Anna-Lena Sachs, Lancaster University, Lancaster, United Kingdom, a.sachs@lancaster.ac.uk; Edward C. Malthouse, Northwestern University, Evanston, Illinois, USA, ecm@northwestern.edu.

maker does not know in advance how much demand will occur on a specific day and how much stock will be left at the end of the day. Additionally, inventory levels might not be recorded correctly, or a set of items might arrive sooner or later than expected, causing the inventory to fluctuate unexpectedly. Some soon-to-perish items might go bad sooner or later than expected according to the storage conditions and the type of item [28]. We propose a novel solution using recommender systems (RS) that uses a personalized top-k list to direct demand toward items that will perish soon and away from items that are nearing stockout while accounting for the above-mentioned uncertainties. Therefore, we make recommendations that not only consider consumer preferences, but also the status of the available stock. Thus, we show how to use RS to manage inventory, integrating inventory management, RS, and promotions literatures.

In the RS literature, earlier works mostly focused solely on user preferences. More recent works started incorporating the needs of the item providers, the RS platform/system itself, and other stakeholders. Multistakeholder recommender systems (MRS) are introduced as systems that include the objectives of parties other than the users. These objectives may conflict with each other, and solutions recommended by MRS algorithms take all these objectives into account while creating the recommendation lists that consist of items offered to the users [1]. Abdollahpouri et al. [1, 2] discuss multiple real-world problems that can be solved using MRS algorithms. For example, in problems that are encountered in an e-commerce retailer setting, the business and retailer considerations are intuitive to consider as well as the user considerations. A traditional RS algorithm could recommend the highest estimated utility items to maximize the user utility/rating. However, these solutions can cause problems in the system by increasing the stockouts and perishability significantly. In our work, the system is considered a stakeholder and aims to lower the number of perished items and stockouts. Similarly, the retailer is another stakeholder that wants to increase the sales revenue as much as possible from the possible revenue obtained from the item recommendations. Consequently, we offer a model that solves an e-commerce retailer MRS problem [11] by including system (sustainability, stockouts) and the provider (revenue) considerations on top of the users' considerations.

Given the proliferation of digital shopping environments such as websites and mobile shopping apps where many user behaviors can be recorded, more data is available to tailor promotions to the needs of individual users [13]. Relevant interactions can be facilitated with an RS so that users receive personalized recommendations that match their preferences. Creating recommendations, however, is more complicated than recommending items of interest to a user because multiple objectives and stakeholders are involved [1, 2]. While we are unaware of any research that uses multi-objective RS to avoid perishables and stockouts at the same time, there has been research that studied other retail objectives and stakeholders. Sürer et al. [51] consider vendors on a retail platform as well as the preference of users when creating top-k lists. Sinan et al. [47, 48] suggest MIP models that create top-k lists utilizing interactions of the users with vendors on a retail platform.

We contribute to the RS literature by proposing and solving a mixed-integer-programming (MIP) model that combines RS and inventory management. Our proposed model creates recommendation lists that consider both user and retailer perspectives, as well as the system-enforced perishability and stockout concerns. Additionally, our model accounts for the stochastic nature of perishables, demands, and inventory levels. By including stochasticity and considerations of the users, the retailer, and the system, we discuss the complex economical impacts of RS in an online retailer setting. As far as we know, our approach to this online retailing problem and our optimization model are novel additions to the RS literature. Furthermore, using ideas from our optimization model, we propose heuristics that improve scalability efficiently. Finally, we discuss the results of our optimization model and heuristics and show that our approaches offer solutions that improve user, retailer, and system objectives simultaneously.

2 RELATED WORKS

This section surveys relevant literature from sales promotion, RS, and inventory management. There is extensive research in the sales promotion literature showing the effectiveness of email promotions, and RS articles have already investigated the effects of personalization in driving retail sales. Likewise, there is research in the inventory management literature showing the importance of alleviating the stockout and perishability issues, mostly focusing on the retailer. Our contribution is to combine these approaches to create personalized email promotions that consider both the user's preferences as well as retailer and supply chain objectives to avoid perishables and stockouts.

Sales promotion is a widely used strategy defined as an "action-oriented marketing event whose purpose is to have a direct impact on the behavior of the firm's customers" [42, p. xvii]. Promotions include price discounts, feature advertising, and other touch points such as targeted emails, special displays, and coupons. Promotions stimulate market demand, enable retailers to sell excess items, and increase their revenue. There is a long literature studying sales promotions and their effects [42], including some with a specific focus on e-commerce and email promotions. Most of the e-commerce promotions literature shows a positive link between sales promotions and increases in demand [3, 36]. Sahni et al. [46] conclude that personalized emails increase the total expenditure of customers by 37.2%.

Others have used RS to implement personalized shopping lists. Lin et al. [37] show that timely recommendations resulted in sales growth. Malthouse et al. [39] solve the problem of sponsored recommendations and content in retailing including ad revenue and user utility. Kaminskas et al. [31] implement recommendations by using text and co-occurrence based approaches. Wan et al. [53] solve recommendations using natural language processing ideas. Chen et al. [12] solve linear optimization-based algorithms, which offer items to users in a deterministic setting using RS considering user ratings with perishable items. Dadouchi and Agard [15] propose an RS method that lowers stockouts. Specifically, RS can be used to increase the revenue of the firm by engaging customers to buy more in the immediate time period [17]. Our MIP model offers an exact solution to creating personalized shopping lists by incorporating both the user and retailer perspectives and also actively modifying the demands before the customer arrivals.

Perishable items and limited inventory appear in multiple works in the dynamic assortment and inventory control literature, but without considering how promotions can be used to manage them. Bernstein et al. [8] solve which items should be offered to which users with limited inventory, not considering perishable items. Talebian et al. [52] create dynamic assortments considering perishable items with infinite inventory. Amiri et al. [6] maximize the number of perishable items sold considering one vendor and multiple buyers. Fan et al. [20] consider both replenishment strategies and the pricing of the perishables. Chua et al. [14] decide whether and how much to discount items using dynamic programming. Others solve problems with perishable inventory by incorporating partial information [19], uncertain demands [26, 34], or time delays happening in the supply chain [18]. Nguyen and Chen [43, 44] build a MIP model and consider perishability and stockouts in an inventory model setting. Chen et al. [12] propose a RS model with perishability in a deterministic setting without stockouts, Dadouchi and Agard [15] propose an RS method that lowers stockouts, Nguyen and Chen [43, 44] propose a model with both perishability and stockouts, but not including user ratings/utility and item recommendation aspects of RS. We close this gap by developing a model that considers stochasticity, perishability, stockouts, user ratings and retailer revenue/sales simultaneously. Furthermore, we offer heuristics to improve the scalability of the suggested optimization model.

3 PROBLEM DEFINITION AND FORMULATION

This section proposes a MIP model and a heuristic to solve an item recommendation problem considering both retailer and user perspectives. We maximize both the quality of the recommendations for the users and the impact on the retailer's profitability (perishability, sales). For this purpose, we first discuss the problem definition and assumptions. We then illustrate and explain our optimization model and heuristic.

3.1 Problem Definition and Optimal Solution

We consider an online retailer selling physical items to their users over the internet. The retailer periodically creates a personalized top-k promotion list such as a weekly email recommending items (I) to a known set of users (U); hereafter we use the RS term *user* instead of retailing terms *customer* or *consumer*. Therefore, we solve the problem of selecting which k items should be recommended to the users in the form of personalized email promotions. While creating top-k lists is a common RS task, we allow lists with fewer than k items to accommodate situations with very low inventory levels; hereafter we use the term "top-k list" with the understanding that some lists may have fewer than k items. We assume that recommending item i to user u might result in a demand increase γ_{iu} for the said item. Note that the RS can increase demand without reducing the price and thus profit margin. We assume that the *retailer's purchase cost* of item i is c_i, and that the retailer earns profit $\rho \times c_i$ when it sells the item, where ρ is the *markup* and $p_i = (1 + \rho) \times c_i$ is the *revenue* (selling price). We allow for items to have different prices p_i but assume a constant markup ρ (this assumption can be easily relaxed).

When deciding which items should be recommended, the following four aspects are considered. Firstly, since the available in-stock inventory ($\boldsymbol{L}$) is limited, there is a possibility of stockouts if users are unable to buy items due to unavailability. When users attempt to purchase a recommended item i that is out of stock, we assume that the retailer incurs a penalty of q_i, which measures a loss of goodwill and revenue from the sale. Secondly, some items are perishable with quantity $\boldsymbol{E}$ considered *soon-to-perish* ($0 \leq E_i \leq L_i, \ \forall i \in I$). Soon-to-perish items must be sold as soon as possible; otherwise, they will be discarded since the retailer cannot sell expired items. For any discarded item, the retailer incurs a penalty c_i (i.e., there is no cost to dispose of the perished items nor salvage value). We also assume that items are shipped following the FIFO (first-in-first-out) method, where items that expire the soonest are shipped first. Retailers enforce FIFO by keeping longer-lasting items in storage. FIFO is often used in the literature with perishable items [32]. Thirdly, different users may have different *ratings* ($\hat{r}_{iu}$) for different items; the RS should recommend items that are of interest to the user, and the user would not have purchased without the recommendation. Lastly, different items have different prices p_i, so the RS should recommend items that result in higher revenue for the retailer.

We consider these four aspects from either the user's or retailer's *perspective*. We create these perspectives by including system-enforced sustainability and stockout considerations to the traditional user consideration of utility/rating and retailer consideration of revenue. From the user's perspective, the recommended list should have items with a high average rating, while avoiding stockouts. The retailer wants to stimulate demand for items that generate high profit, while also avoiding items to perish. Balancing between these perspectives is crucial. The retailer may, for example, need to recommend items with a lower user rating to avoid items perishing. The retailer thus determines the recommendation lists considering the number of perishable items, demand, inventory, and possible demand increase after recommendations. If these values are known with certainty, the problem is deterministic. In this case, after finalizing all recommendation lists, the decision maker knows exactly how many items will be sold, what the profits will be, the average user rating, the numbers of stockout and perished items, and their costs. However, in practice, the decision maker rarely has access to complete information.

By considering stochasticity in our model we obtain top-k lists that account for unexpected situations. Therefore, we incorporate uncertainties in inventory ($\boldsymbol{L}$) and soon-to-perish items ($\boldsymbol{E}$) (both vectors of length $|I|$), and demands with $|I| \times |U|$ matrix $\boldsymbol{D}$. For example, if each value in $\boldsymbol{L}, \boldsymbol{E}, \boldsymbol{D}$ were binary then there would be a total of $2^{2|I|+|I||U|}$ scenarios to investigate. Even with only two realizations, an exhaustive enumeration would be intractable. One solution to this problem is to use Monte Carlo sampling to obtain a finite number of scenarios and the Sample Average Approximation (SAA) method to solve them [33]. We define a function $\max_{x \in X} \left[F(x) \coloneqq \mathbb{E}_{\xi} f(x, \xi)\right]$. We assume f is real-valued and cannot be computed directly, x is a point in solution space $X \subseteq R^d (d < \infty)$, and ξ is some random vector independent of x. We first generate S scenarios $\xi^1, \xi^2, \dots, \xi^S$ from random vector ξ that are independent and identically distributed unless noted otherwise. Assume that the probability of each scenario equals $1/S$. Applying the SAA method, we solve $\max_{x \in X} \left[f_S(x) \coloneqq \frac{1}{S} \sum_{j=1}^{S} f(x, \xi^j) \right]$, with the maximizer of $f_S(x)$ converging (as $S \to \infty$) to the maximizer of $f(x)$ under mild conditions [4]. In our problem, assume that we know the distributions of $\boldsymbol{L}, \boldsymbol{E}, \boldsymbol{D}$. Then, by Monte Carlo sampling, assume we obtain S scenarios by sampling, such that $\xi^1 = (\boldsymbol{L}^1, \boldsymbol{E}^1, \boldsymbol{D}^1), \dots, \xi^j = (\boldsymbol{L}^j, \boldsymbol{E}^j, \boldsymbol{D}^j), \dots, \xi^S = (\boldsymbol{L}^S, \boldsymbol{E}^S, \boldsymbol{D}^S)$. The superscript j is the scenario index, and each possible scenario ξ^j contains the information of the triple $(\boldsymbol{L}^j, \boldsymbol{E}^j, \boldsymbol{D}^j)$. The number S is decided by the decision maker. We use the shorthand notation $[S] = \{1, 2, \dots, S\}$.

3.1.1 Preliminary Optimization Model. Decision variable a_{iu} equals 1 if we recommend item i to user u and 0 otherwise. The estimated rating of item i for user u is denoted as $\hat{r}_{iu}$. For scenario j and item i, the quantity sold is x_i^j, the quantity of unmet demand (number of stockouts) is y_i^j, and the number of perished items is z_i^j. Every user u has a demand value D_{iu}^j for a given item i and scenario j, and demand increases by γ_{iu} if i is recommended to u. Increasing the demand of an item recommended to a user assumption is used in simulations in the literature [22, 27].

$$\max_{a,x,y,z} \quad \frac{1}{S} \sum_{j=1}^{S} \left[\lambda \left(\sum_{i \in I} \sum_{u \in U} \hat{r}_{iu} a_{iu}/k - \sum_{i \in I} q_i y_i^j \right) + (1 - \lambda) \left(\sum_{i \in I} \rho c_i x_i^j - \sum_{i \in I} c_i z_i^j \right) \right] \tag{1}$$

subject to:

$$\sum_{i \in I} a_{iu} \leq k \quad (\forall u \in U) \tag{2}$$

$$x_i^j \leq L_i^j \quad (\forall i \in I, j \in [S]) \tag{3}$$

$$z_i^j \geq E_i^j - x_i^j \quad (\forall i \in I, j \in [S]) \tag{4}$$

$$y_i^j + x_i^j = \sum_{u \in U} \left(D_{iu}^j + \gamma_{iu} a_{iu} \right) \quad (\forall i \in I, j \in [S]) \tag{5}$$

$$x_i^j, y_i^j, z_i^j \geq 0, a_{iu} \in \{0, 1\} \quad (\forall i \in I, u \in U, j \in [S]) \tag{6}$$

Equation (1) specifies our multi-objective function that considers both the user and retailer perspectives. Inside the left parenthesis, we maximize the average estimated ratings and minimize the cost incurred when a demanded item is out of stock (stockout). Inside the right parenthesis, we maximize the revenue of the retailer from the sales and minimize the cost incurred from the perished items. We divide the estimated ratings by k to lower the effect of choice of k on the solution quality. Trade-off parameter $\lambda \in [0, 1]$ is specified by the analyst to control the weight given to the user versus retailer perspective. If $\lambda = 1$ then we only consider the user perspective, and if $\lambda = 0$ then we only consider the retailer perspective. Constraint (2) enforces that we recommend at most k items to each user. We cannot sell more items than the available inventory, enforced by constraint (3). If the number of items sold for i is less than the soon-to-perish count E_i, then the difference is assumed to perish. Constraint (5) controls the flow of demand with the right-hand side equaling the modified demand after recommendations. Therefore, total modified demand is either sold or is considered stockout. Constraint (6) specifies the values that the decision variables can take.

3.1.2 Reformulated Optimization Model. In a deterministic setting with $S = 1$, this model can be easily solved for thousands of users and items. When the number of scenarios increases, however, the number of constraints and decision variables increases rapidly. Next, we propose an optimization model that is more efficient with memory and time as the number of scenarios grows.

Since we assume FIFO we can preprocess some of the uncertainties. For each scenario j and item i, we update $E_i^j = \max\left(0, E_i^j - \sum_{u \in U} D_{iu}^j\right)$ and $L_i^j = \max\left(0, L_i^j - \sum_{u \in U} D_{iu}^j\right)$. These are the respective counts of inventory (L_i^j) and soon-to-perish items (E_i^j) when the RS is not applied (we only consider initial demands). Next, suppose that we sort the number of perishable items ($\boldsymbol{E}$) and inventory ($\boldsymbol{L}$) as follows for each i, $E_i^1 \leq E_i^2 \leq \cdots \leq E_i^{S-1} \leq E_i^S$, and $L_i^1 \leq L_i^2 \leq \cdots \leq L_i^{S-1} \leq L_i^S$. Then, we create two incremental increase arrays for each i: $\tilde{E}_i^j = E_i^j - E_i^{j-1}$, $\forall j \in \{2, \ldots, S\}$ with $\tilde{E}_i^1 = E_i^1$, and, $\tilde{L}_i^j = L_i^{j+1} - L_i^j$, $\forall j \in \{1, \ldots, S-1\}$ with $\tilde{L}_i^S = L_i^S$. Define $X_i = \sum_{u \in U} (\gamma_{iu} \times a_{iu})$, which represents the expected demand increase for item i after implementing the RS. Next, we apply the remodeling idea suggested by Ferguson and Dantzig [21] as follows:

$$\max_{a, \tilde{y}, \tilde{z}} \quad \frac{1}{S} \sum_{j=1}^{S} \lambda \left(\sum_{i \in I} \sum_{u \in U} \hat{r}_{iu} a_{iu}/k - j \sum_{i \in I} q_i \tilde{y}_i^j \right) + (1 - \lambda) \left(\sum_{i \in I} \rho c_i X_i^j - (S + 1 - j) \sum_{i \in I} c_i (\tilde{E}_i^j - \tilde{z}_i^j) - j \sum_{i \in I} \rho c_i \tilde{y}_i^j \right) \tag{7}$$

subject to:

$$\sum_{i \in I} a_{iu} \leq k \quad (\forall u \in U) \tag{8}$$

$$\tilde{y}_i^j \leq \tilde{L}_i^j \quad (\forall i \in I, j \in [S]) \tag{9}$$

$$\tilde{z}_i^j \leq \tilde{E}_i^j \quad (\forall i \in I, j \in [S]) \tag{10}$$

$$\sum_{j=1}^{S} \tilde{z}_i^j \leq X_i \quad (\forall i \in I) \tag{11}$$

$$\sum_{j=1}^{S} \tilde{y}_i^j \geq X_i - L_i^1 \quad (\forall i \in I) \tag{12}$$

$$\tilde{y}_i^j, \tilde{z}_i^j \geq 0, a_{iu} \in \{0, 1\} \quad (\forall i \in I, u \in U, j \in [S]) \tag{13}$$

Equation (7) is a reformulation of equation (1) with new decision variables. For item i, $\tilde{z}_i^j$ is the number of soon-to-perish items sold and $\tilde{y}_i^j$ is the number of stockouts in j scenarios. Then, for calculating the perished items we use the coefficient $(S + 1 - j)$ and penalize $c_i(\tilde{E}_i^j - \tilde{z}_i^j)$. or calculating the stockouts we use the coefficient j and penalize $q_i \tilde{y}_i^j$. Note that we use incremental increase arrays $(\tilde{E}, \tilde{L})$ in this step. Because of how X_i^j is defined, every recommendation increases the equation (7) by ρc_i. The additional term $j \sum_{i \in I} \rho c_i \tilde{y}_i^j$ removes this increase if item i stockouts. Constraint (8) remains the same as constraint (2). Constraints (9) and (10) are simple bounds on decision variables $\tilde{y}$ and $\tilde{z}$, which are generally easily handled by commercial optimization software. Constraint (11) enforces that sum of $\tilde{z}_i^j$ cannot exceed X_i. Constraint (12) enforces that sum of $\tilde{y}_i^j$ should be greater than or equal to $X_i - L_i^1$. Constraint (13) remains the same as constraint (6), except in this model, decision variable x is removed. The value of $\tilde{E}_i^j$ is 0 if $E_i^j = E_i^{j-1}$ (same with L_i^j). Then, the corresponding decision variable $\tilde{z}_i^j$ ($\tilde{y}_i^j$) is removed since it is fixed to 0. These changes improve the previous optimization model's computational time and memory requirement.

3.2 Heuristic Algorithm

For very large datasets, optimization models can struggle with large memory requirements and slow solution times. We offer a heuristic that scales better for larger datasets. This heuristic is both easy to implement and fast to run. We first randomize the order of users. In each step, we recommend a single item i to user u, and then move on to the next user. This procedure continues until either all users have k items in their lists or adding an item to a user's list decreases the objective function value, which only happens if all the inventory is depleted. For a given user u, we recommend the item i^* that solves the objective function:

$$i_u^* = \operatorname*{argmax}_{i \in I} \frac{1}{S} \sum_{j=1}^{S} \lambda(\hat{r}_{iu}/k - q_i \min(0, L_i^j - \gamma_{iu})) + (1-\lambda)(\rho c_i \min(L_i^j, \gamma_{iu}) + c_i \min(E_i^j, \gamma_{iu})). \tag{14}$$

After each recommendation, the $\boldsymbol{L}, \boldsymbol{E}$ and recommendation lists of users are updated: $E_{i^*}^j = \max(0, E_{i^*}^j - \gamma_{i^*u})$, and $L_{i^*}^j = \max(0, L_{i^*}^j - \gamma_{i^*u})$. Each item i^* recommended to user u has rating $\hat{r}_{i^*u}/k$. The value of γ_{i^*u} is the demand increase of item i^* for user u. If the term $L_{i^*} - \gamma_{i^*u}$ is negative, recommending item i^* to user u causes a stockout having penalty q_{i^*}. When a soon-to-perish item i^* is recommended to user u, we incentivize that recommendation by the cost of the item (c_{i^*}) times the number of soon-to-perish items sold ($\min\{E_{i^*}, \gamma_{i^*u}\}$). Lastly, when the retailer sells an item, the objective value increases by $\rho \times c_i$. The heuristic requires solving objective function (14) at most $k|U|$ times.

3.3 Solving the Model with Massive Datasets

Our optimization model can be solved optimally for hundreds of scenarios and thousands of users and items. However, in cases with thousands of scenarios and millions of users and items, we need approximation methods to obtain a solution in a reasonable length of time. This subsection discusses two approximation approaches. First, co-clustering [23] can be applied to users, items, or both. Rather than focusing on each item and user individually, we can cluster them. This can be done by aggregating items similar to each other as only one item, or by aggregating users in clusters if they have similar preferences for similar items. Consequently, even billions of users and items can be manageable in smaller clusters. We apply this idea by reducing the item space using the taxonomy of the categories in our dataset.

Second, we apply a simple idea that we call the "best-N approach," in which we recommend item i to user u only if either item i is one of the top N rated items by that user u, or user u is one of the top N users that rated that item i the highest. None of the other user-item pairs will be considered for recommendation. The decision maker can choose N by using a grid-search algorithm. A similar approach was implemented in the RS literature [10] with good results by solely focusing on the user top N lists. Normally, the number of decision variables created for recommending an item i to user u (a_{iu}) is $|I| \times |U|$. Implementing the best-N approach decreases this number to at most $N \times (|I| + |U|)$. Thus, we reduce the number of decision variables by orders of magnitude, which alleviates both the solution time and memory requirements of the model. The heuristic model's solution time improves similarly. With the best-N approach, some users and items might appear significantly more than others and result in over recommending some items and under recommending others. We alleviate this issue by subtracting the mean rating of each item and user by itself, i.e., de-biasing them. Consequently, items or users will be included in the best-N list only if the rating gain observed is higher relative to the item-user pair.

4 COMPUTATIONAL STUDY

This section discusses the data used in our paper and the computational results of the models proposed. All results are obtained by using a laptop with Intel(R) Core(TM) i7-8750H CPU @ 2.20GHz 2.21 GHz processor information with 16.0 GB of installed RAM specifications. The optimization model is solved using Gurobi, a commercially available mathematical programming solver, [25] 9.5.0 with the optimization gap set to 10^{-4} and a one-hour time limit. Gurobi obtains the global optimal solution (within the optimization gap) using a linear-programming based branch&bound algorithm [41]. In the worst case, the algorithm implemented in Gurobi can have exponential time complexity [41], but in practice, the time complexity is much better than exponential. The heuristic approach is solved with Python 3.7.

4.1 Dataset Description

We consider an online grocery retailer located in the United States and learn user preferences from their purchase history. We examine regular users who have shopped at least 15 times within the last 6 months, giving 3731 users ($|U|$) with 78,195 orders. These orders include 27,158 unique item stock-keeping units (SKUs). Each SKU has a price, brand, and category from a retailer-provided taxonomy. We aggregate the SKUs following [39]: we match the retailer-provided taxonomy of sub-categories with brand names to come up with 4106 ($|I|$) unique item names. For example, "HAIR PRODUCTS - BRAND NAME", is considered one item. We manually tag delivery items, fresh market items, and some dairy items as soon-to-perish items using sub-categories. The number of perishable items is 651 out of 4106.

Average demand values are calculated by averaging the quantity bought for each user-item pair for a week, denoted as $\overline{D}_{iu}$. We create a $|U| \times |I|$ rating matrix by using the function $\log(x+1)$, x as the number of orders including item $i \in I$ for user $u \in U$. Next, we compare prediction algorithms SVD, k-NN, and Co-clustering using the Surprise package [29] to estimate the values of unknown item-user pairs. We use the SVD algorithm, which performed the best using 5-fold cross-validation, with RMSE and MAE values of 0.24 and 0.20, which is better than for k-NN (RMSE=0.26, MAE=0.21) and co-clustering (RMSE=0.60, MAE=0.55). Finally, the values are min-max normalized between 0 and 1. Denote the value calculated for item-user pair as η_{iu}, which we take as the probability of buying an item i in case it is recommended to the user u. We assume that the probability of buying and money spent on an item correlates with the utility of that item to the user. Therefore, the ratings are calculated as $\hat{r}_{iu} = (1+\rho) \times c_i \times \eta_{iu}$, which is the price times purchase probability. With this assumption, all values in the objective function are in dollars. More complex utility/rating choices can be considered by the decision maker. The γ_{iu} values are calculated as $\left\lceil \overline{D}_{ui} + 0.1 \right\rceil \times \eta_{iu}$. This value is continuous and can be interpreted as the average expected increase in demand when item i is recommended to user u. We assume high purchase probability and high previous demand are both important in recommender quality. We consider that recommendations can play two different roles: either the user will be reminded to order their regular needs, or they will be recommended novel items that they might want to buy. Because γ is not related to c_i, items with low purchase probability and low previous demand will rarely get recommended for that user, even if the item has a high price. Stockout costs q_i are set equal to costs of the items c_i, $\forall i \in I$. We use a constant markup value $\rho = 0.26$ [45] for simplicity, but the decision maker can choose different markups for different products.

4.2 Evaluation Procedure

This subsection discusses the settings and metrics used to evaluate our models, and benchmarks implemented to compare our approaches. These benchmarks are denoted B_p, B_r, B_s, and B_u, and they solely optimize perishability, retailer sales, stockouts, and average user rating objective functions, respectively. In other words, each benchmark obtains a solution considering only one criterion. Therefore, we compare our solutions solving multiple objectives with those focusing on a singular objective. The letter H indicates the heuristic solution and O indicates optimization. This letter is followed by a number indicating the weight (λ) value. For example, the optimization solution with weight $\lambda = 0.5$ is denoted as $O5$. We obtain solutions for $\lambda \in \{0.1, 0.5, 0.9\}$, corresponding to a higher focus on retailer perspective, equal focus, and a higher focus on user perspective, respectively.

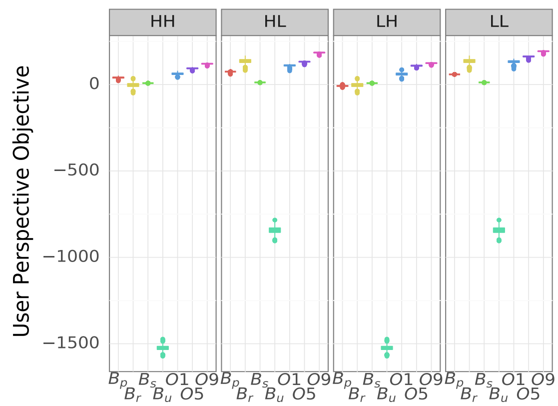
We investigate four settings: high perishables and low stockout risk (HL), low perishables and high stockout risk (LH), both high risk (HH), and finally, both low risk (LL) settings. Settings with a high risk of perishables have a larger number of soon-to-perish items $\boldsymbol{E}$, and those with a high risk of stockout have lower levels of inventories $\boldsymbol{L}$. We choose $\psi = \{0.2, 0.6\}$ and $\epsilon_L = \{30, 100\}$, and their combinations create our four settings. We generate inventory levels using $L_i \sim \text{Poisson}(\epsilon_L + \sum_{u \in U} \overline{D}_{iu})$, where $\overline{D}_{iu}$ is the expected demand for item i and user u. Average demands vary greatly from as low as 0.4 to as much as thousands. We choose lower ϵ_L values for high-risk stockouts and higher values otherwise. The demand values D_{iu} are distributed as Poisson($\overline{D}_{iu}$). In this way, we create upper bounds that are tight relative to the total demand of each item. We create a number of soon-to-perish items as a percentage of inventories such as $E_i = L_i\psi_i$, where ψ_i is distributed as a truncated normal $\mathcal{N}(\psi, 0.1)$ with bounds $[0, 1]$, $\forall i \in I$. Some items perish sooner than others, and the decision maker can incorporate this by choosing higher values of ψ_i for items that perish more rapidly and lower values otherwise. We base the number of soon-to-perish items as a percentage of the inventory, so higher inventory will result in a larger number of soon-to-perish items. For high-risk perishable cases ψ_i will be closer to 1, and 0 otherwise. As a shorthand notation, HH is $\psi = 0.6, \epsilon_L = 30$, HL is $\psi = 0.6, \epsilon_L = 100$, LH is $\psi = 0.2, \epsilon_L = 30$, and finally, LL is $\psi = 0.2, \epsilon_L = 100$.

Unless stated otherwise, we generate 500 scenarios (S) for $\boldsymbol{L}$, $\boldsymbol{E}$, and $\boldsymbol{D}$ considering each of the four settings. We obtain solutions for each approach by solving the problem with these 500 scenarios. Then, we generate 2500 new out-of-sample scenarios to test the quality of the obtained solutions. Overall, a solution is better if it has higher user ratings, retailer sales, or lower perishability and stockout objective values. These objective values (metrics) are calculated as follows, where x_{iu} is 1 if item i is recommended to user u and 0 otherwise, s_i is the number of item i sold, z_i is the number of perished item i, y_i is the number of stockouts of item i:

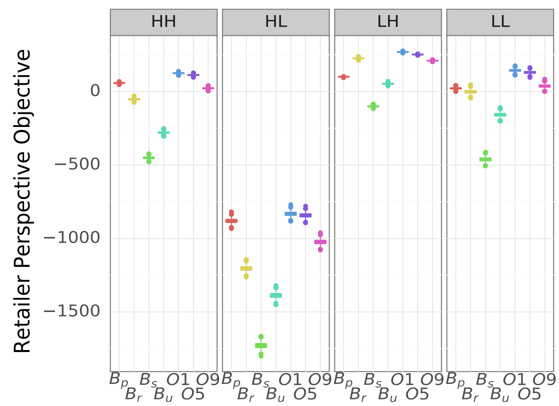
$$\text{Ratings} = \sum_{i \in I} \sum_{u \in U} \frac{\hat{r}_{iu}}{k} \mathrm{x}_{iu}, \qquad \text{Sales} = \sum_{i \in I} \rho c_i \mathrm{s}_i, \qquad \text{Perishability} = \sum_{i \in I} c_i \mathrm{z}_i, \qquad \text{Stockouts} = \sum_{i \in I} q_i \mathrm{y}_i,$$
$$\text{User Perspective} = \text{Ratings} - \text{Stockouts}, \qquad \text{Retailer Perspective} = \text{Sales} - \text{Perishability} \tag{15}$$

4.3 Results

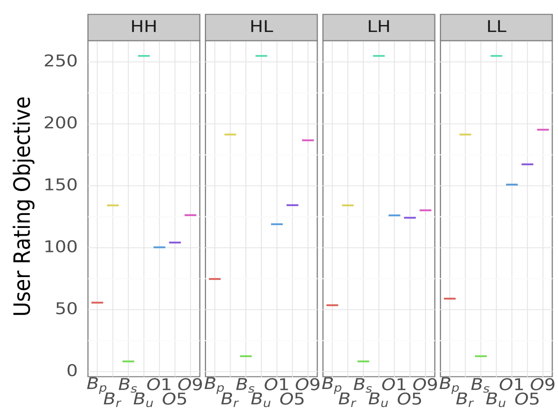
This subsection presents and discusses results obtained using our optimization model and heuristics, and compares them with solutions of benchmark models. Furthermore, we investigate the effect of including stochasticity and heuristics on the solution quality.



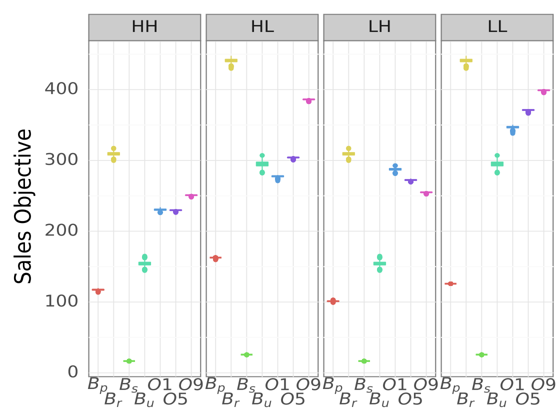
(a) User perspective objective values



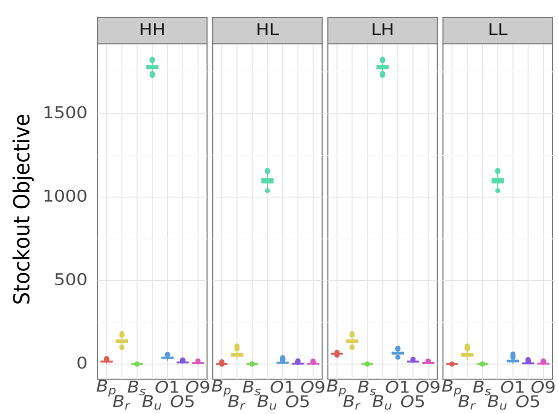
(b) Retailer perspective objective values



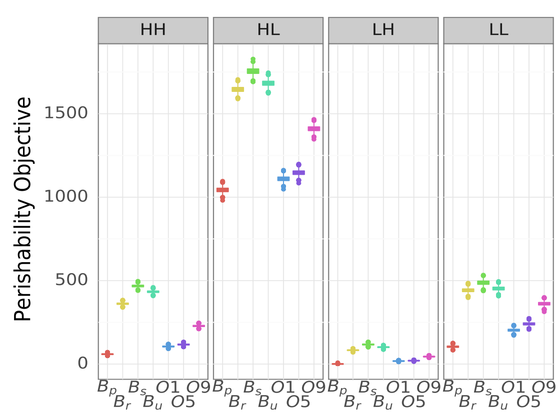
(c) User rating objective values



(d) Sales objective values



(e) Stockout objective values



(f) Perishability objective values

Fig. 1. Comparison of benchmarks, in the subfigures (e) & (f) lower values and for the rest higher values are better

4.3.1 Solution Quality of Optimization Approach. We compare our model's solution with the benchmarks in Figure 1. We use the best-N method with $N = 100$ and offer at most $k = 10$ items to every user. Figure 1 (a) shows that the overall objective values for the **user perspective**, which includes user ratings and stockouts, are similar for all approaches, except for B_u, which is the benchmark where only the average user rating is optimized. Despite B_u performing well in terms of user rating (Figure 1 (c)), users would experience many stockouts (Figure 1 (e)) that reduce customer goodwill, which results in significantly worse user perspective objective value. As expected, in the high stockout risk settings (HH and LH) the number of stockouts observed is greater, causing slightly lower user perspective objective values. The optimal approach achieves the highest user perspective objective values for all settings.

Considering the objective values from the **retailer perspective**, which include perished items and sales, we observe more varied results than those from the user perspective. The values for the high perishables and low stockouts (HL) are the lowest, indicating that many items perish, which generates high waste costs, as illustrated by Figure 1 (f). The exact opposite setting with low perish and high stockout (LH) achieves the highest retailer perspective values because of the low waste costs. The retailer perspective objective values are similar for HH and LL, which is surprising at first because they consider exact opposite situations. Looking into these results in more detail, we see that for LL, the sales are higher than for HH (Figure 1 (d)), but so are the number of perished items. In general, the solutions tend to slightly decrease the sales objective values to improve the perishability objective, resulting in a better retailer perspective objective value. We next discuss each individual objective in more detail.

When considering the **user ratings** in subfigure (c), B_u performs the best, but as discussed before, suffers from a high number of stockouts (Figure 1 (e)), which negatively affects users. Interestingly, the solution B_r results in high user ratings as well but incurs higher stockouts too. The B_r solution offers items with the greatest monetary gain considering the users' probability of buying the item recommended to them. This can also be observed from subfigure (d), where B_r achieves the highest sales objective values. This benchmark is different from only offering items with high price margins, and additionally considers the probability of users buying the recommended item, thus achieving recommendations with high user ratings. Therefore, this competitive benchmark considers both retailer and user information, and is most similar to approaches that maximize the retailer's expected profit [5, 16]. Since the monetary return is the only consideration, B_r solutions result in large numbers of stockouts and perished items. Since the recommendations are not scenario dependent, the user rating of each solution is the same for all scenarios.

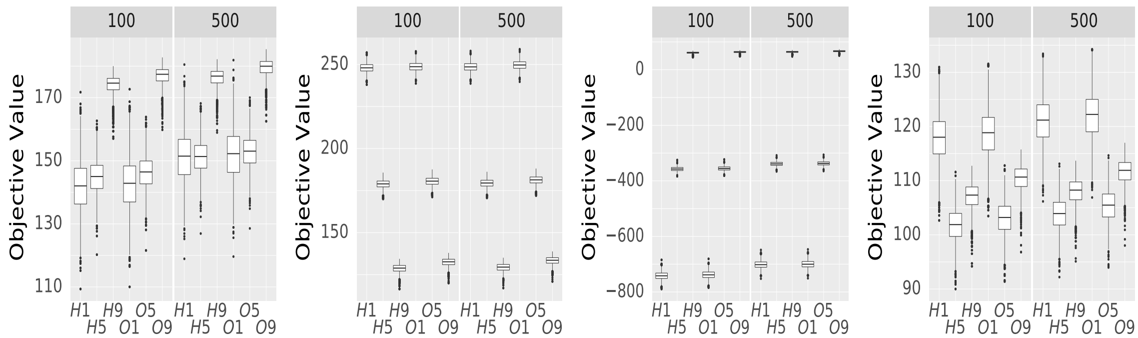
Subfigure (d) shows the **sales** objective function values. The LL setting results in the greatest sales due to the retailer being able to sell the highest profit items in a low-risk stockout and perishability environment. Counterintuitively, most of the time sales increase when the optimization solution focuses less on the retailer perspective. This happens because an increased focus on the retailer perspective usually lowers the current sales in favor of selling lower-margin and lower-rating items that will perish soon, causing long-term profits instead of short-term. Generally, less focus on retailer perspective results in less focus on recommending soon-to-perish items, thus increasing the retailer sales but decreasing the retailer perspective objective value due to perished items. This is again an important distinction that traditional RS might miss while trying to maximize the revenue of the retailer. However, the LH setting is an exception to this rule. The optimization solutions in setting LH already have a low number of perished items (subfigure (f)) and thus a weight increase results in recommending high-rating items that are not necessarily high profit for the retailer while lowering the stockouts. Benchmarks B_s and B_p perform the worst because sales are negatively affected when the only concern is selling soon-to-perish items or keeping the stockouts to a minimum.

The number of **stockouts** (see Figure 1(e)) is in a similar range for all approaches except for benchmark B_u. Interestingly, the B_u approach might seem to be user-centric, however, it results in a poor user experience because of the large number of stockouts, where users cannot purchase items that were recommended to them. This is an important and general shortcoming of the top-k recommendation rule and is a disadvantage of applying RS without considering demand changes that follow. Our optimization solution considers user experience overall and results in users obtaining the items that are recommended to them. Next, benchmark B_s solely focuses on minimizing the stockout objective function. In this benchmark, not recommending anything is one solution with the optimal value of 0. This benchmark is useful for analyzing the solution quality when little to no recommendations are made. The objective values of perishability, user ratings, and sales are significantly underperforming. Interestingly, due to the existence of multiple-optima solutions, the sales objective value is not exactly zero in some cases. For the optimal solutions, larger weights result in better stockout objective values because the focus shifts to the user perspective.

Figure 1 (f) shows the **perishability** objective function values. HL has the highest perishability values due to high inventory (low stockout risk) and high-risk perishability, which results in the greatest number of soon-to-perish items. We observe that low stockout with a low perishability ratio (LL) can result in a larger number of perished items than low inventory with a high perishability ratio (HH). Thus, we note that low risk stockout is not always desirable and the retailer should order less to reduce the number of perished items due to high inventory levels. The benchmark B_p minimizes the perishability objective function by recommending soon-to-perish items first, and thus achieves the best perishability results. The optimization solution with weight 0.1 is a close second to B_p in the perishability objective. However, benchmark B_p struggles when it comes to sales and user ratings. The proposed recommendations are neither what users prefer nor the items with high-profit margins. If the RS focuses solely on soon-to-perish items, both retailer and user perspectives suffer. The perishability objective, in general, decreases when the weight decreases due to more focus on the retailer perspective.

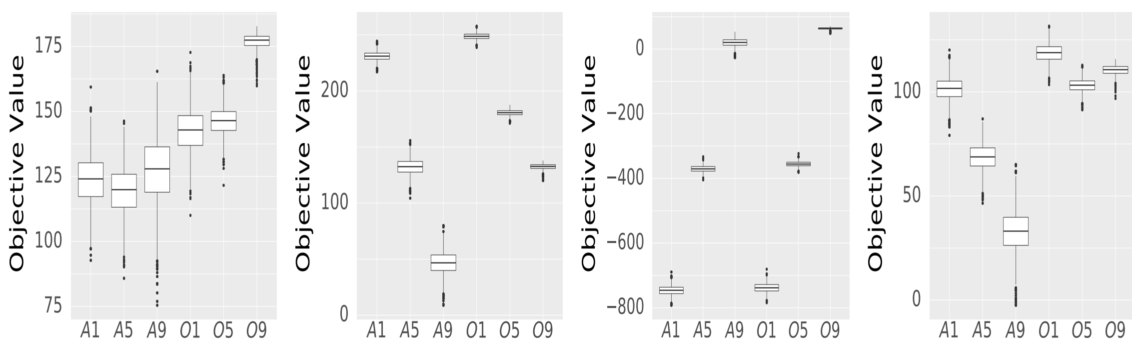
Overall, considering the user and retailer perspectives in subfigures (a) and (b), the benchmark B_u performs badly on the user metrics due to stockouts, and B_s performs badly on retailer metrics due to low sales. Benchmark B_r performs worse in user perspective when stockout is high risk, and in retailer perspective when perishability is high risk. Benchmark B_r might seem like a good option for the retailer at first, however, the stockouts and perished items that are ignored make it undesirable. Benchmark B_p performs worse than our models on user ratings and sales, due to recommendations being made towards selling the soon-to-perish items without considering user ratings or revenue. We have investigated two additional benchmarks: user-perspective (consider both user rating and stockouts) and retailer-perspective (consider both retailer sales and perishability). The user-perspective benchmark was only slightly better in user perspective objective value while significantly worse in retailer perspective objective value relative to the optimization solution due to a high number of perished items. The retailer-perspective objective value was almost identical to the optimization solution but significantly worse in user perspective solution due to a high number of stockouts. Overall, the optimization solutions perform the best when considering both the user and retailer perspectives.

4.3.2 Solution Quality of Heuristic Approach. Figure 2 compares the objective function values for heuristic approaches. Each subfigure considers one of the four settings with best-N approach applied, where $N \in \{100, 500\}$. The difference in the objective values between $N = 100$ and 500 is minimal. The objective function value difference is even smaller for $N > 500$. This result is very useful for a retailer offering a large number of products since only considering a smaller subset of the most preferred products for each user and most preferred users for each product decreases the problem size significantly. Therefore, using the best-N approach improves the memory and solution time for both models while maintaining the solution quality. We note that our heuristic approach also achieves objective function values close



(a) $\psi = 0.2$ and $\epsilon_L = 100$ (Setting LL) (b) $\psi = 0.2$ and $\epsilon_L = 30$ (Setting LH) (c) $\psi = 0.6$ and $\epsilon_L = 100$ (Setting HL) (d) $\psi = 0.6$ and $\epsilon_L = 30$ (Setting HH)

Fig. 2. Objective value comparison for $N = \{100, 500\}$



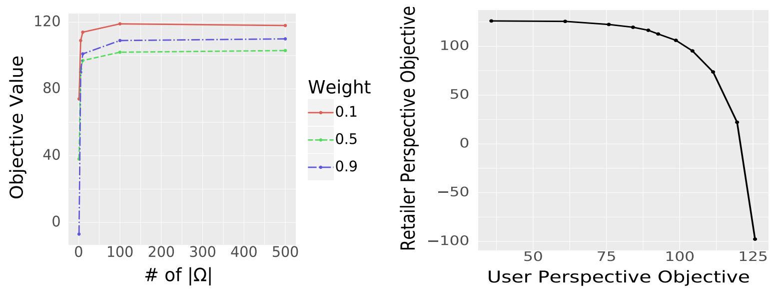
(a) Setting LL (b) Setting LH (c) Setting HL (d) Setting HH

Fig. 3. Objective value comparison of using scenarios

to the optimal solution (within 1%). The heuristic performs similarly to the optimization model that considers all the metrics in Equation 15.

4.3.3 Solution Quality Comparison of Stochastic and Deterministic Cases. We analyze the solutions using expected values of $\boldsymbol{L}, \boldsymbol{E}, \boldsymbol{D}$ instead of creating 500 scenarios for demand, inventory, and perishability. We denote these solutions with A and use the same weights as before, i.e., $\{0.1, 0.5, 0.9\}$. Figure 3 shows that it is advantageous to consider stochasticity because using only the expected values ignores the variance of the data. We conclude that it is better to account for uncertainty compared to using a solution estimated with averages.

4.3.4 Solution Quality Changes with Weights and Number of Scenarios. Figure 4 provides additional insights by focusing on HH (high in both perishability and stockouts). The discussions are similar in other settings. Subfigure (a) shows that creating 500 scenarios is more than adequate, and after 100 scenarios the improvement in the objective function value is negligible. We also observe that fewer scenarios result in a significant drop in performance for weight $\lambda \in \{0.1, 0.5, 0.9\}$. Therefore, solutions obtained by considering a variety of scenarios perform better. For example, a decision maker who



(a) (b)

Fig. 4. Objective value changes with the number of scenarios in (a), Objective values for different perspectives with different weights in (b), in setting HH

considers only the latest 3 weeks' worth of data ($S = 3$) would be at a significant disadvantage. Subfigure (b) shows the trade-off curve between the user and retailer perspectives with weights ranging from 0 to 1. We observe that increasing the weight from 0 to 0.5 almost triples the user perspective objective while reducing only 0.1 of retailer perspective objective. Therefore, considering both perspectives rather than only one improves the overall quality of the solution. The decision maker can select the best weight according to the needs of the user, retailer, or both.

5 FUTURE RESEARCH & CONCLUSION

This article proposes a MIP model and heuristics that consider RS objectives from the user and retailer perspectives. The user perspective aims to obtain highly rated item recommendations while minimizing stockouts. The retailer perspective aims to maximize profit while minimizing the losses incurred by the perished items. Our models find solutions that are high in quality for both criteria. We offer approximation methods that scale better than the optimization model. We propose a heuristic model and show that its solution quality is nearly as good as the optimal one. Therefore, the reader can decide to aim for the optimal solution and use the optimization model, or use the heuristic, which is faster to solve and more scalable. Finally, we study the improvements made to the objective function values using our models, and compared the solutions with each of the benchmarks in different settings.

Our work can be extended in multiple directions. Firstly, more emphasis on inventory could be incorporated into the retailer perspective. In this way, excess inventory could be considered and items with higher storage spaces might need to be recommended more often. In our work, we create recommendations considering inventory as constant (although not deterministic), however, inventory levels of items could be added as decision variables as well. Note that, different user segments could behave differently to stockouts (e.g., purchasing a substitute), and incurring different penalties to different user segments can be a possible direction.

The stochasticity can be applied to different parts with more knowledge of the uncertainties. Scenarios can be generated using the background knowledge of the system. For example, the inventory generation process can represent a problem with supply chain disruptions. Our work assumes the distribution of the parameters, but more work can be done solely focusing on the solution quality changes tied to the distributions. If more complex distributions are considered, the solution quality of the models may change, and it would be worthwhile to investigate the changes in solution quality with changes in distributions.

While creating our parameters such as the ratings of the items for the users and the effect of recommendation on the increased demand for a given item, we had to make certain assumptions. We use an online retailer dataset in an offline setting. If our work is extended to an online setting then it would be possible to understand and update our parameters accordingly. Next, by obtaining data from the users continuously, we could improve the quality of the parameters for each user to reflect their needs better. Even in an offline setting, future research can include different parameter creation ideas and investigate the changes in solution qualities in different settings.

REFERENCES

[1] Himan Abdollahpouri, Gediminas Adomavicius, Robin Burke, Ido Guy, Dietmar Jannach, Toshihiro Kamishima, Jan Krasnodebski, and Luiz Pizzato. 2020. Multistakeholder recommendation: Survey and research directions. *User Modeling and User-Adapted Interaction* 30, 1 (2020), 127–158.

[2] Himan Abdollahpouri, Robin Burke, and Bamshad Mobasher. 2017. Recommender systems as multistakeholder environments. In *Proceedings of the 25th Conference on User Modeling, Adaptation and Personalization*. 347–348.

[3] Narendra Agrawal, Sami Najafi-Asadolahi, and Stephen A Smith. 2020. Optimization of operational decisions in digital advertising: A literature review. *Channel Strategies and Marketing Mix in a Connected World* (2020), 99–146.

[4] Shabbir Ahmed, Alexander Shapiro, and Er Shapiro. 2002. The sample average approximation method for stochastic programs with integer recourse. *Submitted for publication* (2002), 1–24.

[5] Leman Akoglu and Christos Faloutsos. 2010. Valuepick: Towards a value-oriented dual-goal recommender system. In *2010 IEEE International Conference on Data Mining Workshops*. IEEE, 1151–1158.

[6] Seyyed Amir Hossein Salehi Amiri, Ali Zahedi, Morteza Kazemi, Javad Soroor, and Mostafa Hajiaghaei-Keshteli. 2020. Determination of the optimal sales level of perishable goods in a two-echelon supply chain network. *Computers & Industrial Engineering* 139 (2020), 106156.

[7] Eric T Anderson, Gavan J Fitzsimons, and Duncan Simester. 2006. Measuring and mitigating the costs of stockouts. *Management Science* 52, 11 (2006), 1751–1763.

[8] Fernando Bernstein, A Gürhan Kök, and Lei Xie. 2015. Dynamic assortment customization with limited inventories. *Manufacturing & Service Operations Management* 17, 4 (2015), 538–553.

[9] Michael Blakeney. 2019. *Food loss and food waste: Causes and solutions.* Edward Elgar Publishing.

[10] Keith Bradley and Barry Smyth. 2001. Improving recommendation diversity. In *Proceedings of the twelfth Irish conference on artificial intelligence and cognitive science, Maynooth, Ireland*, Vol. 85. Citeseer, 141–152.

[11] Robin Burke and Himan Abdollahpouri. 2017. Patterns of multistakeholder recommendation. *arXiv preprint arXiv:1707.09258* (2017).

[12] Du Chen, Yuming Deng, Guangrui Ma, Hao Ge, Yunwei Qi, Ying Rong, Xun Zhang, and Huan Zheng. 2020. Inventory Based Recommendation Algorithms. In *2020 IEEE International Conference on Big Data (Big Data)*. IEEE, 617–622.

[13] Kwok-Wai Cheung, James T Kwok, Martin H Law, and Kwok-Ching Tsui. 2003. Mining customer product ratings for personalized marketing. *Decision Support Systems* 35, 2 (2003), 231–243.

[14] Geoffrey A Chua, Reza Mokhlesi, and Arvind Sainathan. 2017. Optimal discounting and replenishment policies for perishable products. *International Journal of Production Economics* 186 (2017), 8–20.

[15] Camélia Dadouchi and Bruno Agard. 2018. Lowering penalties related to stock-outs by shifting demand in product recommendation systems. *Decision Support Systems* 114 (2018), 61–69.

[16] Aparna Das, Claire Mathieu, and Daniel Ricketts. 2009. Maximizing profit using recommender systems. *arXiv preprint arXiv:0908.3633* (2009).

[17] M Benjamin Dias, Dominique Locher, Ming Li, Wael El-Deredy, and Paulo JG Lisboa. 2008. The value of personalised recommender systems to e-business: a case study. In *Proceedings of the 2008 ACM conference on Recommender systems*. 291–294.

[18] Alexandre Dolgui, Manoj Kumar Tiwari, Yerasani Sinjana, Sri Krishna Kumar, and Young-Jun Son. 2018. Optimising integrated inventory policy for perishable items in a multi-stage supply chain. *International Journal of Production Research* 56, 1-2 (2018), 902–925.

[19] Qinglin Duan and T Warren Liao. 2013. A new age-based replenishment policy for supply chain inventory optimization of highly perishable products. *International Journal of Production Economics* 145, 2 (2013), 658–671.

[20] Tijun Fan, Chang Xu, and Feng Tao. 2020. Dynamic pricing and replenishment policy for fresh produce. *Computers & Industrial Engineering* 139 (2020), 106127.

[21] Allen R Ferguson and George B Dantzig. 1956. The allocation of aircraft to routes—An example of linear programming under uncertain demand. *Management science* 3, 1 (1956), 45–73.

[22] Daniel Fleder and Kartik Hosanagar. 2009. Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management science* 55, 5 (2009), 697–712.

[23] Thomas George and Srujana Merugu. 2005. A scalable collaborative filtering framework based on co-clustering. In *Fifth IEEE International Conference on Data Mining (ICDM'05)*. IEEE, 4–pp.

[24] Peter S. Goodman. March 28, 2022. New Supply Chain Risk: 22,000 Dockworkers Who May Soon Strike. *New York Times* (March 28, 2022).

[25] Gurobi Optimization, LLC. 2021. Gurobi Optimizer Reference Manual. https://www.gurobi.com

[26] Alejandro Gutierrez-Alcoba, Roberto Rossi, Belen Martin-Barragan, and Eligius MT Hendrix. 2017. A simple heuristic for perishable item inventory control under non-stationary stochastic demand. *International Journal of Production Research* 55, 7 (2017), 1885–1897.

[27] Naieme Hazrati and Francesco Ricci. 2022. Recommender systems effect on the evolution of users' choices distribution. *Information Processing & Management* 59, 1 (2022), 102766.

[28] Maarten LATM Hertog, Ismail Uysal, Ultan McCarthy, Bert M Verlinden, and Bart M Nicolaï. 2014. Shelf life modelling for first-expired-first-out warehouse management. *Philosophical Transactions of the Royal Society A: Mathematical, Physical and Engineering Sciences* 372, 2017 (2014), 20130306.

[29] Nicolas Hug. 2020. Surprise: A Python library for recommender systems. *Journal of Open Source Software* 5, 52 (2020), 2174. https://doi.org/10.21105/joss.02174

[30] Xiaoqing Jing and Michael Lewis. 2011. Stockouts in online retailing. *Journal of Marketing Research* 48, 2 (2011), 342–354.

[31] Marius Kaminskas, Derek Bridge, Franclin Foping, and Donogh Roche. 2017. Product-seeded and basket-seeded recommendations for small-scale retailers. *Journal on Data Semantics* 6, 1 (2017), 3–14.

[32] Itir Z Karaesmen, Alan Scheller-Wolf, and Borga Deniz. 2011. Managing perishable and aging inventories: review and future research directions. *Planning production and inventories in the extended enterprise* (2011), 393–436.

[33] Sujin Kim, Raghu Pasupathy, and Shane G Henderson. 2015. A guide to sample average approximation. *Handbook of Simulation Optimization* (2015), 207–243.

[34] Mervegül Kırcı, Işık Biçer, and Ralf W Seifert. 2019. Optimal replenishment cycle for perishable items facing demand uncertainty in a two-echelon inventory system. *International Journal of Production Research* 57, 4 (2019), 1250–1264.

[35] Sandra Lebersorger and Felicitas Schneider. 2014. Food loss rates at the food retail, influencing factors and reasons as a basis for waste prevention measures. *Waste Management* 34, 11 (2014), 1911–1919.

[36] Randall A Lewis and David H Reiley. 2014. Online ads and offline sales: measuring the effect of retail advertising via a controlled experiment on Yahoo! *Quantitative Marketing and Economics* 12, 3 (2014), 235–266.

[37] Koung-Lung Lin, JY-J Hsu, Han-Shen Huang, and Chun-Nan Hsu. 2005. A recommender for targeted advertisement of unsought products in e-commerce. In *Seventh IEEE International Conference on E-Commerce Technology (CEC'05)*. IEEE, 101–108.

[38] Brian Lipinski, Craig Hanson, Richard Waite, Tim Searchinger, James Lomax, and Lisa Kitinoja. 2013. Installment 2 of "Creating a Sustainable Food Future": Reducing Food Loss and Waste. *World Resources Institute Working Paper* (2013), 1–40. Retrieved May 2, 2022 from https://search.issuelab.org/resource/installment-2-of-creating-a-sustainable-food-future-reducing-food-loss-and-waste.html

[39] Edward C Malthouse, Yasaman Kamyab Hessary, Khadija Ali Vakeel, Robin Burke, and Morana Fudurić. 2019. An algorithm for allocating sponsored recommendations and content: Unifying programmatic advertising and recommender systems. *Journal of Advertising* 48, 4 (2019), 366–379.

[40] Sébastien Marchand. 2012. The relationship between technical efficiency in agriculture and deforestation in the Brazilian Amazon. *Ecological Economics* 77 (2012), 166–175.

[41] David R Morrison, Sheldon H Jacobson, Jason J Sauppe, and Edward C Sewell. 2016. Branch-and-bound algorithms: A survey of recent advances in searching, branching, and pruning. *Discrete Optimization* 19 (2016), 79–102.

[42] Scott A Neslin. 2002. Sales promotion. *Handbook of Marketing* 13 (2002), 311–338.

[43] Duc Huy Nguyen and Haoxun Chen. 2019. Optimization of a perishable inventory system with both stochastic demand and supply: comparison of two scenario approaches. *Croatian Operational Research Review* 10, 1 (2019), 175–185.

[44] Duc Huy Nguyen and Haoxun Chen. 2022. An effective approach for optimization of a perishable inventory system with uncertainty in both demand and supply. *International Transactions in Operational Research* 29, 4 (2022), 2682–2704.

[45] Timothy J Richards and Jura Liaukonyte. 2019. Switching costs and store choice. *Available at SSRN 3470937* (2019).

[46] Navdeep S Sahni, Dan Zou, and Pradeep K Chintagunta. 2017. Do targeted discount offers serve as advertising? Evidence from 70 field experiments. *Management Science* 63, 8 (2017), 2688–2705.

[47] Sinan Seymen, Himan Abdollahpouri, and Edward C Malthouse. 2021. A Constrained Optimization Approach for Calibrated Recommendations. In *Fifteenth ACM Conference on Recommender Systems*. 607–612.

[48] Sinan Seymen, Himan Abdollahpouri, and Edward Carl Malthouse. 2021. A unified optimization toolbox for solving popularity bias, fairness, and diversity in recommender systems. In *Workshop of Multi-Objective Recommender Systems (MORS'21), in conjunction with the 15th ACM Conference on Recommender Systems, RecSys*, Vol. 21. 2021.

[49] Stephen A Smith and Dale D Achabal. 1998. Clearance pricing and inventory policies for retail chains. *Management Science* 44, 3 (1998), 285–300.

[50] Ilan Stavi and Rattan Lal. 2013. Agriculture and greenhouse gases, a common tragedy. A review. *Agronomy for sustainable development* 33, 2 (2013), 275–289.

[51] Özge Sürer, Robin Burke, and Edward C Malthouse. 2018. Multistakeholder recommendation with provider constraints. In *Proceedings of the 12th ACM Conference on Recommender Systems*. 54–62.

[52] Masoud Talebian, Natashia Boland, and Martin Savelsbergh. 2014. Pricing to accelerate demand learning in dynamic assortment planning for perishable products. *European Journal of Operational Research* 237, 2 (2014), 555–565.

[53] Mengting Wan, Di Wang, Jie Liu, Paul Bennett, and Julian McAuley. 2018. Representing and recommending shopping baskets with complementarity, compatibility and loyalty. In *Proceedings of the 27th ACM International Conference on Information and Knowledge Management*. 1133–1142.

[54] Sarah Zimmerman. Dec 15, 2021. 10 disruptions that rocked supply chains in 2021. *Supply Chain Dive* (Dec 15, 2021). https://www.supplychaindive.com/news/top-supply-chain-disruptions-2021/611513/