Personalized Pricing Recommender System

Multi-Stage Epsilon-Greedy Approach

Toshihiro Kamishima and Shotaro Akaho
National Institute of Advanced Industrial Science and Technology (AIST)
AIST Tsukuba Central 2, Umezono 1-1-1, Tsukuba, Ibaraki, 305-8568 Japan
mail@kamishima.net (http://www.kamishima.net/) and s.akaho@aist.go.jp

ABSTRACT

Many e-commerce sites use recommender systems, which suggest items that customers prefer. Though recommender systems have achieved great success, their potential is not yet fulfilled. One weakness of current systems is that the actions of the system toward customers are restricted to simply showing items. We propose a system that relaxes this restriction to offer price discounting as well as recommendations. The system can determine whether or not to offer price discounting for individual customers, and such a pricing scheme is called price personalization. We discuss how the introduction of price personalization improves the commercial viability of managing a recommender system, and thereby improving the customers' sense of the system's reliability. We then propose a method for adding price personalization to standard recommendation algorithms which utilize two types of customer data: preferential data and purchasing history. Based on the analysis of the experimental results, we reveal further issues in designing a personalized pricing recommender system.

Categories and Subject Descriptors

H.3.3 [**Information Storage and Retrieval**]: Information Search and Retrieval—*Information filtering*

Keywords

personalized pricing, recommender system, collaborative filtering, multi-armed bandit

1. INTRODUCTION

A recommender system searches for items and information that would be useful to a user based on the user's behaviors or the features of candidate items [7]. Collaborative filtering (CF) is an algorithm that implements this recommender system by automating the word-of-mouth paradigm. GroupLens [13] and many other recommender systems emerged in the mid-1990s, and further experimental and practical systems have been developed during the explosion of Internet merchandizing. In the past decade, such recommender systems have been introduced and managed at many e-commerce sites to improve customer satisfaction and increase profits.

Though recommender systems have achieved great success, their full potential has not yet been reached. Existing systems are fundamentally limited to showing items that the customers would prefer; the systems cannot behave like clerks in real store. For example, boutique attendants make suggestion about coordinating clothes, and car retailers consult with customers about car accessories. Such sophisticated actions in relation to customers are not feasible for the current systems.

To open a new chapter in online merchandising, we propose a recommender system that can take an action other than recommendation: namely *price personalization*. Price personalization allows the system to adjust the prices for an item based on the customer's features, just like price negotiation with real shoppers. We discuss a system that can offer price discounts, as well as make recommendations, if a customer is expected to buy the recommended item only when it is discounted.

In this paper, we propose a *personalized pricing recommender system* (PPRS), which is a recommender system having the functionality of price personalization. This system exploits two types of customer data: preferential data and purchasing histories. Existing recommendation algorithms are used to predict customers' preference patterns from preferential data. Based on the predicted preference patterns and customers' purchasing histories, the system's offers to users are determined using multi-armed bandit algorithms [17, 6].

Our contributions are summarized as follows. First we discuss the commercial viability of recommender systems and the customers' sense of their reliability. We then enumerate the issues in designing a PPRS and propose a method for such a system. Finally, based on the analysis of experimental results using quasi-synthetic data, we reveal further issues in designing a PPRS.

In section 2, after explaining price personalization, we discuss how it will influence the system's commercial viability and the customers' sense of its reliability. A personalized pricing recommender system is proposed in section 3, and its experimental results are shown in section 4. Sections 5 and 6 cover related work, a discussion, and our conclusion.

HetRec '11, October 27, 2011, Chicago, IL, USA

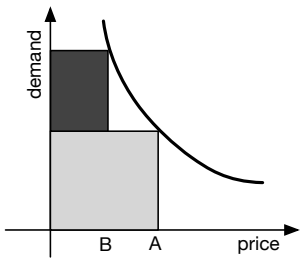


Figure 1: Added value brought by price personalization

NOTE: The horizontal and vertical axes of this chart respectively represent sales prices and the extent of demand, i.e., sales volume, at those prices.

2. PRICE PERSONALIZATION AND RECOMMENDER SYSTEMS

After describing what price personalization is, we discuss merits of introducing it into a recommender system.

2.1 Price personalization

Price personalization is a pricing scheme that allows sellers to adjust the price for an item depending on the user or transaction [18], and is also known as dynamic pricing or price customization.

Figure 1 shows the additional profit to sellers by introducing price personalization. In this chart, profit corresponds to the area of the rectangle, i.e., price × demand. If all customers are charged a flat price, the price is fixed at A, at which the area of the corresponding square (depicted by a gray rectangle) is maximized. After introducing price personalization, sellers offer price B only to customers who will not buy at price A but who will buy at price B. Accordingly, sellers obtain the additional profit depicted by the black rectangle, together with the existing profit. Further, when the seller offers a discount to those who would not buy at price A but would at price B, these customers gain by buying an item at a price (A - B) cheaper than the standard price. One may think that customers who bought at price A will complain if they discover that others are offered a lower price. We can assert that this kind of trade is indeed fair, if these customers are then guaranteed a discount in a future transaction probabilistically.

This price personalization is a kind of price discrimination, a pricing scheme where different prices are charged for the same item. In cases of existing price discrimination, prices are changed based on factors such as the physical location of the sale or customer demographics, e.g., gender or age. For example, hamburger chain stores sell the same hamburger at different prices in different sales regions. Another example is special offers for senior citizens or sales campaigns that specifically target women. One obstacle to successfully implementing such price discrimination is the resale effect: specifically, when customers buy items at low prices and then resell them at higher prices. In this case, the seller loses potential customers, and the sales volume decreases. Therefore, for price discrimination to succeed, resale activity must be blocked. In the above example of hamburger shops, if regions where prices differ are sufficiently far apart, the quality of the hamburgers declines during transport, so they cannot be resold at higher prices.

Unlike traditional price discrimination, price personalization mainly targets e-commerce. At e-commerce sites, because the sales volumes of individual customers are precisely controlled, it is generally difficult for customers to resell large numbers of items. These sites deal with personalized items, such as registered air tickets or subscription services, which cannot be resold. Further, pricing schemes are more personalized than in traditional price discrimination. For example, e-commerce sites can randomly change prices for different customers and can investigate whether or not each customer accepts the offered prices. From these sampled data, the prices can be adjusted for each customer.

At e-commerce sites, price personalization has already been introduced, but these attempts have had problems [16, 1]. We suppose that these problems can be avoided by being sincere to the customers and by changing the timing at which discounts are offered. The e-commerce sites mentioned above did not notify their customers of the fact that prices are personalized. However, if a seller notifies its customer that it has changed a price and the price is lowered rather than raised, the customer has already accepted price personalization. For example, many customers know and accept the fact that loyal customers might be offered a lower price by rewards, coupons, or some other format. Price personalization has been widely introduced in many real retail operations, such as car dealerships. We therefore think that customers would become more tolerant of price personalization if it is sincerely operated. Further, for each target item, a specific customer can be offered a discounted price only when the customer first views the item. Even if the customer views the same item later, the system never offers the discounted price. This rule is required to avoid a customer countermeasure: namely, delaying purchasing. It is well known that such inter-temporal price discrimination can be avoided if a customer delays purchasing until a discount is offered [15]. This rule also makes price comparison services, such as the price grabber or the kakaku.com [1] ineffective because discounting will not be offered on the later visits.

2.2 The Merits of Price Personalization

Here we discuss the merits of introducing the functionality of price personalization to a recommender system. As described before, the introduction of price personalization enhances commercial viability and thereby results in recommendations that are more satisfactory to customers. In other words, the recommendations become more reliable.

We first discuss the commercial viability of managing a recommender system. To improve this viability, the profit gained by managing a recommender system must be much larger than its management cost. Recommender systems might help to increase customer loyalty by providing information that would be beneficial to customers, and thereby bring additional profits to sellers. In exchange for this additional profit, sellers must pay a cost for managing recommender systems. However, because the effect of loyalty on the profit is indirect and uncertain, the additional profit might be inadequate to compensate for the management cost. Price personalization can alter this situation. It can bring fully additional profit as shown in Figure 1, and the profit should be enough to compensate for the management

[1] `http://www.pricegrabber.com` and `http://kakaku.com`

cost. Consequently, the commercial viability of a recommender system can be improved by introducing price personalization.

We then move on to the reliability of the recommendation for customers. The maximization of profit can often conflict with customers' needs. More concretely, recommender systems might offer items that do not meet customers' real needs. For example, a seller can increase its profit by selling more expensive items instead of offering lower-cost items that will satisfy customers' needs. Indeed, Shani et al. discussed a recommender system to maximize the seller's utility instead of maximizing the customer satisfaction [14]. Sellers are more motivated to make such dishonest recommendations if the management of a system is not commercially viable. Price personalization can improve the commercial viability, and thus sellers are less motivated to make such insincere recommendations at the risk of losing customers' long-term loyalty. Consequently, customers can receive more reliable recommendations from a recommender system that includes price personalization. And, of course, customers have the additional benefit of being offered price discounts.

From the discussion above,we can conclude that introducing price personalization into a recommender system is beneficial to both sellers and customers.

3. PERSONALIZED PRICING RECOMMENDER SYSTEM

In this section, we describe our *Personalized Pricing Recommender System* (PPRS), which has the functionality of price personalization. After formalizing the task and objectives of a PPRS, we discuss three problems in designing a PPRS: ambiguity in observation, class imbalance, and the exploitation–exploration trade-off. Finally, we show our implementation of our PPRS.

3.1 Formalization of a PPRS

We start from the simplest PPRS because to our knowledge, this is the first attempt to develop a recommender system with price personalization. A PPRS is passively invoked for an item that a customer is currently viewing or accessing; the system offers discounts when the customer is expected to buy the item only if a discounted price is offered. We don't consider a system that can actively select a discounted item to present to customers in this paper. We assume the prices of all items that are sold at the site are the same; this is the case for music downloading or pay-per-view videos. We further consider that there are only two levels of prices: a standard and a discounted. That is to say, the system can choose only whether to offer a standard price or a discounted one.

In response to the system's offer of a standard or a discounted price, a customer determines whether or not to buy an item. Given a target item, customers can be classified into three types based on their responses to the system.

1. *Standard*: Customers who will buy an item regardless of whether the price is standard or discounted.

2. *Discount*: Price-sensitive customers who will buy an item only if a discounted price is offered.

3. *Indifferent*: Customers who will not buy an item whether or not it is discounted.

Table 1: A summary of actions of different types of customers when standard or discounted pricing is offered

	Customer Type		
Offer	Standard	Discount	Indifferent
Standard	Buy	Not buy	Not buy
Discounted	Buy	Buy	Not buy

Table 2: A summary of a seller's rewards when different types of customers buy an item or not

	Customer Type		
Response	Standard	Discount	Indifferent
Buy	α	β	0
Not Buy	0	0	γ

We would like to emphasize these behavioral responses apply only to end customers. Otherwise, an indifferent customer who would not consume an item himself/herself might buy an item at a discounted price for the purpose of resale.

Table 1 summarizes the responses of each type of customer when a standard or a discounted price is offered. A standard customer always buys and an indifferent one never buys. A discount customer buys only if a discounted price is offered.

We then design the seller's rewards when the system receives a response from a customer. The rewards that can be gained by offering a standard price and a discounted price are denoted by α and β ($\beta < \alpha$), respectively. The system predicts which type of customer is visiting the site, and chooses a price accordingly. To a customer predicted to be a standard type, the system offers a standard price, because offering a discounted price results in the potential loss of $\alpha - \beta$. If the customer buys the item, the seller gains the reward of α, and otherwise gains no reward. To a customer predicted as a discount type, the system offers a discounted price, because the customer won't buy at a standard price. If the customer buys, the seller gains the reward of β, and otherwise gains no reward. For an indifferent customer, the system offers the standard price. This is because if the customer is not an end customer and is a reseller, he/she could buy a discounted item to resell it. Similar to the case with the first two types, with this type no reward is primarily gained if a customer doesn't buy. However, when an indifferent customer buys a target item, the item may be resold and the seller potentially loses an opportunity to directly sell it. Such a loss can be quantified by a negative value for the buying response; however, dealing with negative rewards profits is technically inconvenient. We hence design rather tricky rewards so that the seller gains a small potential reward when the indifferent customer doesn't buy and gains no reward when the customer buys. We assume that amount of such a potential reward is very small, i.e., $\gamma \ll \alpha, \beta$. These rewards are summarized in Table 2.

3.2 Three problems in designing a PPRS

We sequentially discuss three problems in designing a PPRS: ambiguity in observation, class imbalance, and the exploitation–exploration trade-off.

3.2.1 Ambiguity in Observation

The problem of ambiguity in observation is the difficulty in

distinguishing the type of customer. A recommender system doesn't know the true type of customer and the type must be guessed from customers' responses in Table 1. A PPRS must learn a rule to predict customers from the sequence of the customer's responses. We here want to emphasize that the system can offer either a standard or a discounted price. When offered a standard price, the standard customer buys whereas neither the discount nor the indifferent customer buys. Therefore, even if records of these actions are available, the learned classifier cannot discriminate between discount and indifferent customers. Similarly, when the system offers a discounted price, it cannot discriminate between standard and discount customers. To solve this problem, we take a multi-stage classification approach.

3.2.2 The Class Imbalance Problem

The class imbalance problem is the decline in accuracy when the distribution of classes is highly skewed [10]. When the number in one class is much smaller than that in another class, an instance in the minority class has a strong bias to be classified into the majority class. Because the number of indifferent customers is generally much larger than that of standard or discount customers, standard and discount customers tend to be missed with high probability, due to this class imbalance problem. To alleviate this problem, we adopt prescreening and class-weighting techniques.

3.2.3 An exploitation–exploration Trade-Off

The problem of an exploitation–exploration trade-off emerges when collecting training data. When a system predicts that a customer is a discount type but he/she is truly a standard type, the system offers a discounted price and potentially loses the reward of $\alpha - \beta$. Accordingly, to check whether the current prediction is really correct, a system has to collect training data by offering non-optimal prices for the customer type that the system is currently predicting. Inversely, if the system takes such non-optimal actions too frequently, the total reward will be reduced. Such a balance between collecting data and taking the best action is known as the problem of the exploitation–exploration trade-off.

The task of attempting to optimize this trade-off is called the *multi-armed bandit* task [6, 17]. Given a fixed set of candidate actions, which are also called arms, a system can repeatedly select one of these candidates. Each time it takes the selected action, the system gets a reward as the response to the action. If the system perfectly knows the reward for every action, the system should always select the action whose reward is the maximum among the candidates. However, these rewards are unknown to the system and must be predicted. Whenever it receives a reward, the system updates the prediction model. This updated model is used to determine the next action by taking into account the exploitation-expectation trade-off. The multi-armed bandit algorithm tries to minimize the *regret*, which measures by how much the amount of the actually gained reward is less than the amount of the best possible reward.

3.3 Implementation of a PPRS

We finally show our implementation of our PPRS. A PPRS utilizes two types of data sets: preferential data to items and purchasing histories of customers. The preferential data are used for acquiring a recommendation model, which is then used together with purchasing histories for training classifiers to predict customers' buying behaviors. We consider a scenario that a seller has been already managed conventional recommender systems before introducing a PPRS. Hence, we assume that a PPRS can initially use preferential data and purchasing histories under the condition that the seller always offers a flat standard price to all customers. After starting a PPRS, a system can collect additional preferential data and purchasing histories while offering discount or standard prices to customers accordingly.

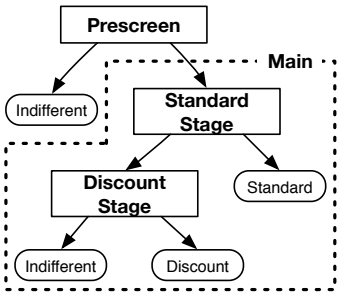


Figure 2: Customer discrimination process

A model-based recommender systems is trained by using these preferential data, and the acquired model is used for making decisions on discounting as well as for performing conventional recommendations. When a customer chooses or views an item, a PPRS determines whether or not it offers a discount price to the customer. A PPRS trains classifiers to make this decision based on the features of the target customer–item pair. As the feature vectors of data for training these classifiers, we exploit the demographics of the customer and the model parameters of the above recommender system that are related to the target customer–item pair. This is because it would be a realistic assumption that a customer's price sensitivity depends on the customer's demographic information and is closely related to the customer's preference for the target item as well. The classifiers are further updated based on the customer's response, and a PPRS repeatedly offers a standard price or a discount one whenever customers view items by considering the balance of the exploitation–exploration trade-off.

We designed a PPRS so as to solve the problems described in the previous section. We take a multi-stage classification approach to alleviate class imbalance and to face ambiguity in observation. The exploitation–exploration trade-off and class imbalance are dealt with by a multi-armed bandit algorithm and a class-weighting technique, respectively.

3.3.1 Multi-Stage Classification

Our system identifies a type of a customer by the process shown in Figure 2. This process consists of two major stages: the prescreening and main stages. The prescreening stage aims to exclude apparently indifferent customers to alleviate a class imbalance problem. The next main stage discriminates customers by combining two classifiers to deal with ambiguity in observation.

In the prescreening stage, easily detected indifferent customers are eliminated. For this purpose, we exploit a standard recommender engine. Given a target item–customer pair, the customer is regarded as indifferent if the predicted score of the customer for the target item is low. It is natural to consider that customers would not buy items that

they don't prefer. Half or more of indifferent customers are easily eliminated by this prescreening. This is an important advantage of combining a recommender system with the functionality of personalized pricing.

The main stage of this customer classifier consists of two sub-stages. In the first standard stage, standard customers and the other two types are discriminated. These non-standard customers are further divided into discount or indifferent types by the second discount classifier.

Standard and discount classifiers can be learned from the following data sets. Training data for a standard classifier are generated from purchasing histories that a customer has made in response to the offers of a standard-price items. Positive data consist of customers that buy at the standard price, and the others are treated as negative data. Further, if a discounted item is offered and a customer doesn't buy it, this fact implies that the customer would not buy the item at the higher, standard price. Therefore, these records can be also used as negative examples for a standard classifier. Training data for a discounted classifier are generated from records of the customer's actions when the system offers a discounted price and when the customer is additionally classified as the non-standard type. This is because the aim of this classifier is to identify a discount customer among customers that are already known as non-standard types. For this classifier positive data consist of customers that buy at a discounted price, while others are treated as negative data. Note that if a standard classifier misclassifies, the record of the response is inappropriate for training a discounted classifier, but such a misclassification cannot be detected. However, we consider that this problem is not so serious, because the prediction accuracy of the standard classifier is gradually improved as the amount of training data increases. Consequently, ambiguity in observation is resolved in this main stage.

3.3.2 Multi-armed Bandits and Class-Weighting

We next apply a multi-armed bandit technique to face the exploitation–exploration trade-off. Though the number of bandits is equal to that of arms in the case of a standard bandit, there are three bandits but we can chose only two arms in our case. Specifically, there are three types of customers, but a system can only take actions offering a standard or a discounted price. We hence apply standard bandit methods for each of the standard and discount classifiers in Figure 2. Note that because standard bandit techniques can optimize rewards at each classifier, the total rewards obtained by combining two classifiers may not be optimal.

Figure 3 shows the detailed flow of actions including exploitation and exploration in the main stage in Figure 2. A prescreened customer-item pair is inputted into the left standard classifier. When the pair is classified as standard type, this prediction result is exploited and the system offers a standard price. The other action is explored and the input pair is passed to a discount classifier. If the pair is classified as a non-standard type, the system takes the opposite action to that of the standard-type case. When the input pair is passed to a discount classifier, the pair is then classified into a discount type or an indifferent type. Given the prediction result, the system chooses to exploit or to explore the result, and the final action is determined in a similar way.

Furthermore, our PPRS is contextual. In a standard bandit, rewards are predicted based on the history of responses. However, in our case, the rewards additionally depend on the features of the customers and items. Such bandit problems are called bandits with covariates, contextual bandits, and so on [12]. Many sophisticated methods have been developed for these bandits with covariates, but these methods don't deal with multi-staged selection as in our case. We hence adopt the most naive approach, ϵ-greedy [17], which performs well if parameters are well tuned [12, 4]. In this approach, a system chooses the best action whose predicted reward is the maximum with probability $1 - \epsilon$ and explores the other non-optimal actions with probability ϵ. Note that this method cannot automatically tune an exploitation–exploration trade-off.

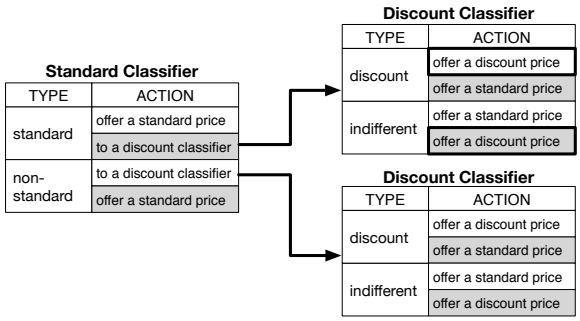


Figure 3: The detailed flow of actions in the main stage in Figure 2

NOTE: Each tabular represents a classifier of the main stage in Figure 2. The first column "TYPE" shows a customer type that is predicted by the corresponding classifier. The second column "ACTION" shows actions that a system selects. In this column, cells with white and gray backgrounds correspond to exploitation and exploration actions for the predicted customer type, respectively.

Finally, to deal with the class imbalance problem, we adopt a class-weighting technique. We used two thresholds, STh and DTh, and if a discount classifier and a standard classifier predicted class probabilities larger than these thresholds, the customer was classified as standard type and discount type, respectively. While a smaller STh indicates heavily weighing a standard class in a standard classifier, a smaller DTh indicates heavily weighing a discount class in a discount classifier. Customers come to be classified into heavily weighted classes with high probability.

4. EXPERIMENT

We implemented a simple system and applied it to quasi-synthetic data.

4.1 Experimental Conditions

The details of the experimental conditions are as follows.

4.1.1 Procedure

Our experimental process was composed of preliminary and main phases. While a system learn a recommendation model and an initial standard classifier in the preliminary phase, a PPRS in section 3.3 is performed in the main phase. In the preliminary phase, a seller manages a conventional recommendation system and always offers a flat standard

Table 3: Evaluation score table for evaluating the PPRS

(a) true reward

	S	D	I
S	α	0	0
D	0	β	0
I	0	0	γ

(b) observed reward

	S	D	I
S	α	β	0
D	0	β	γ
I	0	0	γ

(c) profit

	S	D	I
S	α	β	α
D	0	β	0
I	0	0	0

NOTE: Rows and columns of each subtable show the true and expected customer types, respectively. Letters, 'S', 'D', and 'I' indicate standard, discount, and indifferent customers, respectively.

price to all customers. A seller can collect preferential data of customers and items and purchasing histories of customers when a standard price is offered. These preferential data and purchasing histories are respectively used for building a model for a recommender system and for training an initial standard classifier, which is updated in the subsequent main phase. In the main phase, a PPRS system determines whether or not to offer discounts and updates standard and discount classifiers based on the customer's response. Note that we did not update recommendation models in the main phase for simplicity.

4.1.2 A Data Set

To test the above simple system, we generated quasi-synthetic data from MovieLens' one-million dataset[2]. We used the preference data of MovieLens, but the users' purchasing histories were synthesized. We consider that the synthesized histories should minimally satisfy these conditions: (a) Preference for the target items would become stronger in the order of a standard customer, a discount customer, and an indifferent customer. (b) The determination of purchasing activities was assumed to depend on the customers' preference for the target items and their demographics. (c) Almost all customers are indifferent, and the number of discount customers is slightly larger than that of standard customers. To satisfy these three conditions, we selected the customers who rated the target item as 5 (the most preferred) and whose age was greater than 45 years. To hide the 5 ratings, the ratings 4 and 5 were both treated as rating 4 when training the recommender systems and standard and discount classifiers. Those selected male and female customers were treated as standard and discount customers. We'd like to emphasize that though this purchasing history is simple, it is not trivial for a system to be able to obtain additional reward because of the problems shown in section 3.2. We actually tried diverse settings of parameter settings, but it was very difficult to exceed even baselines.

Half of the rating data were used in the preliminary phase to train for an initial standard classifier and a model for the recommender system. The other half of the rating data were used in the main phase. Standard and discount customers were selected as described above, and the ratios of these customers were about 3.5% and 1.4%.

4.1.3 Evaluation Scores

Table 3 shows three types of evaluation scores: a true reward, an observed reward, and a profit. For example, a system predicts that a customer is indifferent, but his/her true type is discount. In this case, a true reward, an observed reward, and a profit are respectively increased by 0, β, and 0 as shown in the second row and the third column of each subtable. The performance is quantified by the sum of scores derived by repeating this process for all customers. The first true reward is incremented if and only if the prediction of the customer types is correct. The second observed reward is calculated based on customers' responses. A true customer type cannot be always identified as pointed out in section 3.2.1. The observed reward increases if the customer's action agrees with the action that the system expected. We want to maximize the true reward, but a system can only know an observed reward in reality. Accordingly, it would be preferable for these two rewards to be highly correlated. The third profit is the sum of prices of items bought by customers. This score is used to check whether the total profit is increased by employing price personalization.

To satisfy the condition, $\alpha > \beta \gg \gamma$, reward parameters, α, β, and γ were set to 1.0, 0.5, and 0.01, respectively. If the system perfectly knew the types of customers, the best possible total reward was 25468, which is the upper bound of the reward scores. If the system offered a flat standard price, a standard customer and an indifferent customer respectively providing the rewards α and γ, the system received a total reward of 22028, which is a baseline for true and observed rewards. When a standard price is offered to all customers, the total profits brought by all standard customers was 17268, which considered as a baseline for a profit score.

4.1.4 Other Conditions

We adopted two types of conventional recommender systems, pLSA and matrix decomposition (MD), which are described in detail in [11]. For constructing standard and discount classifiers, we adopted logistic regression. A standard classifier was initially trained by the purchasing history of the first half of the data set, and it was periodically updated every time 50k responses were given. A discount classifier was trained based on the responses to the offering of discounted prices. The classifier exploited only the responses when the target user was expected not to buy at a standard price and was updated for every 50 responses.

As a multi-armed bandit technique, we took an ϵ-greedy approach. An exploration probability, ϵ was changed in the range $[10^{-1}, 10^{-3.5}]$. Class thresholds STh and DTh were changed in the ranges $[0.001, 0.5]$ and $[0.5, 0.999]$, respectively. Note that the F-measure of the initial standard classifier was maximized when STh was about 0.25. We performed ten runs for each setting of STh, DTh, and ϵ, and the average rewards over these runs were reported. We prescreened the user–item pairs whose predicted scores were less than 3.

4.2 Experimental Results

The averages of three types of scores are shown in table 4. This table shows the results under the condition of parameters where both a true reward and an observed reward were maximized. Successfully, all scores were larger than their respective baselines.

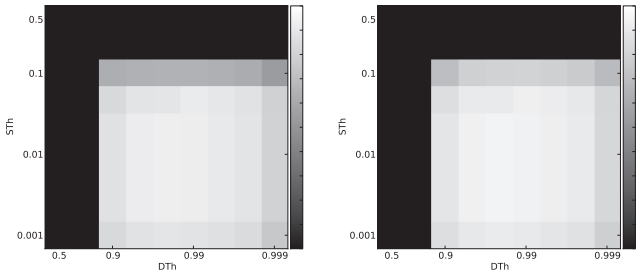
We'd like to discuss why these rewards and profits are maximized when STh is small and DTh is large. To avoid the loss of rewards, a standard customer is very important because they bring such large rewards with transactions. Consequently, a standard classifier should attach importance to

[2]GroupLens research lab: `http://www.grouplens.org/`

Table 4: Averages of three scores

	true reward	observed reward	profit
pLSA	23855.1	23897.4	20331.1
MD	23930.7	23970.2	20346.7

NOTE: These are the scores under the settings that observed rewards were maximized. For a pLSA model, the settings of parameters were $\mathsf{STh} = 0.01$, $\mathsf{DTh} = 0.97$, and $\epsilon = 10^{-2.3}$. For a MD model, the settings of parameters were $\mathsf{STh} = 0.003$, $\mathsf{DTh} = 0.97$, and $\epsilon = 10^{-2.4}$. Baselines of rewards and profits were 22028 and 17268, respectively.



(a) true reward (b) observed reward

Figure 4: Changes of average true rewards and average observed rewards as STh and DTh were varied

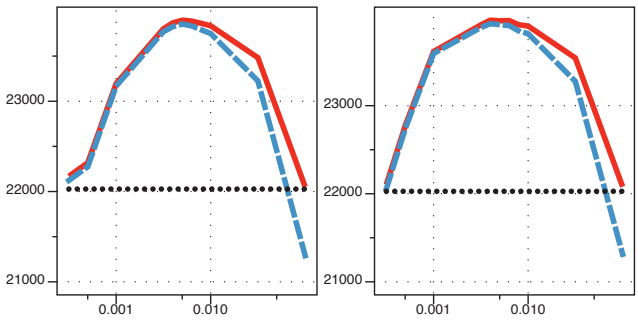
NOTE: The exploration probability ϵ was set to $10^{-2.3}$ and a pLSA model was used. Horizontal and vertical axes correspond to DTh and STh, respectively. The intensities of the cells indicate the amounts of rewards in the range, $[22000, 24000]$, and the rewards lower than 22000 were clipped.

high recall, and STh should lessen so as not to miss standard customers. For a discount classifier, high precision is preferred, and so the lager DTh is preferred. In a extreme case where all standard customers are detected by a standard classifier and only one discount customer is detected by a discount classifier, the total reward indeed increases.

Figure 4 shows the averages of true rewards and observed rewards according to the variation of two thresholds, STh and DTh. As described above, when standard and discount classifiers respectively emphasized recall and precision, both rewards became larger. In the opposite cases, rewards were drastically degraded. In addition, the peaks of these rewards agreed, and the variation of these rewards were generally correlated. This observation is highly preferable, because a system cannot know the true reward, which the system wants to maximize, and the observed reward is maximized in substitution.

The average rewards at different exploration probability ϵ are shown in Figure 5. Two rewards peaked at the same ϵ, and this result is again very preferable for training a PPRS. Finally, this parameter, ϵ, greatly affects the performance of a PPRS. To automatically tune this ϵ, we plan to adopt more sophisticated bandit algorithms, such as UCB1 [6].

These results lead to the conclusion that our PPRS could successfully gain rewards larger than all baselines under the tuned settings, and thereby could obtain the additional profits, even though a PPRS system had three problems: ambiguity in observation, class imbalance, and the exploitation–exploration trade-off.



(a) pLSA model (b) MD model

Figure 5: Changes in averages of observed rewards according to the increase of ϵ

NOTE: Horizontal and vertical axes show exploration probabilities, ϵ, and gained rewards, respectively. For a pLSA model, thresholds of classifiers were STh=0.01 and DTh=0.97. For a MD model, thresholds of classifiers were STh=0.003 and DTh=0.97. Broken, solid, and dotted lines respectively represent true rewards, observed rewards, and baseline rewards.

5. RELATED WORK

Some recommender systems employ bandit approaches. A bandit technique was applied to a Web content optimization task [2, 4, 12], which can be considered as a kind of a recommendation task. The goal of this task was to select a set of contents, such as banner advertisements or news topics, that would be most frequently accessed or clicked by target users. These selections are optimized by using a bandit technique, but it is modified because there are some constraints, such as the minimum number of exposures or the delayed feedback of users, for this optimization task. These kinds of constraints should be taken into account also in our PPRS systems in the future, because these constraints would be useful to guarantee fairness among customers. Weng et al. [19] used a bandit for finding good peer recommendation sites in a P2P-like network. Each peer site recommends an item to users at the site, and recommendations of other peers are used to improve the recommendation performance. Bandit methods are used for finding good peer recommenders.

Some recommender systems try to adjust a balance between the precision of preference prediction and the other goals. Reinforcement learning has been used to maximize the lifetime value of sellers in recommender systems [14], and this system can offer items that will be more profitable for sellers in the future instead of an item that best meets a customer's immediate need. ValuePick [5] considers both the proximity to a target user and the global value of a network, when recommending a relevant node in the network.

Kleinberg and Leighton proposed a bandit algorithm for pricing [9]. In their model, a seller sequentially sells identical goods to many customers. In each transaction, the seller offers a price, and the customer can choose whether or not to accept the price. The seller determines the price of the next transaction based on this history of transactions. Unlike this model, our pricing model also depends on the features of items and customers. Further, we considered the possibility of resale.

There are several related contributions in mechanism design literature. Stokey described an inter-temporal price conditioning [15]. In this article, the seller first offers a

higher price to a customer. If the customer does not buy it, the seller offers a lower price the next time. Stokey proved that this price conditioning never exceeds flat pricing, because a customer can postpone buying until a lower price is offered. However, Acquisti and Varian showed the possibility that price conditioning can surpass flat pricing [3]. This is derived by the additional marginal utility of customers when a seller provides a personalized service, such as one-click services or a recommendation that exploits the customers' purchase history. Bergemann and Ozmen discussed the added value brought by a recommender system [8]. This added value is obtained by reducing uncertainty in a customer's knowledge about items. They showed that several equilibrium patterns in market share can emerge from this added value.

Terui and Dahana proposed a model customer's sensitivity in price personalization [18]. A reference price is a price in a customer's mind and is used as a baseline to evaluate an item when the customer purchases it. Customers cannot sense a drop in price until it reaches a specific threshold that is some extent lower than this reference price. Conversely, the customer isn't aware of an increase in price if the price does not exceed a specific threshold that is some extent higher than the reference price. They proposed a model that is useful for estimating these thresholds, which can be exploited in order to increase profit.

6. DISCUSSION AND CONCLUSIONS

In this paper, we proposed a personalized pricing recommender system: namely, a recommender system having the functionality of personalizing prices. We discussed how it improves the commercial viability of managing a recommender system, and thereby improving the customers' sense of the system's reliability. We then implemented a simple system and pointed out issues that are specific to the implementation of a PPRS.

In a PPRS, prices of items are directly considered as rewards. If utility functions other than prices can be considered as rewards, a PPRS could be made applicable to broader purposes. For example, when sellers have a large amount of dead stock, the system can attempt more aggressive discounting. In addition to prices, cross-selling, point services, or transportation duration can be dealt with. Recommender systems have now started to provide not only simple recommendations but also more sophisticated suggestions that are more beneficial for customers, as human retailers would. Such evolved systems could be called *Attendant Systems*.

7. ACKNOWLEDGMENTS

This work is supported by the grants-in-aid 14658106, 16700157, and 21500154 of the Japan society for the promotion of science. We would like to thank for providing a data set for the Grouplens research lab.

8. REFERENCES

[1] Web sites change prices based on customers' habits. CNN.com. `http://edition.cnn.com/2005/LAW/06/24/ramasastry.website.prices/`.

[2] N. Abe and A. Nakamura. Learning to optimally schedule internet banner advertisements. In *Proc. of The 16th Int'l Conf. on Machine Learning*, pages 12–21, 1999.

[3] A. Acquisti and H. R. Varian. Conditioning prices on purchase history. *Marketing Science*, 24(3):367–381, 2005.

[4] D. Agarwal, B.-C. Chen, and P. Elango. Explore/exploit schemes for web content optimization. In *Proc. of The 9th IEEE Int'l Conf. on Data Mining*, pages 1–10, 2009.

[5] L. Akoglu and C. Faloutsos. Valuepick: Towards a value-oriented dual-goal recommender system. In *ICDM Workshop: Optimization Based Techniques for Emerging Data Mining Problems*, 2010.

[6] P. Auer, N. Cesa-Bianchi, and P. Fischer. Finite-time analysis of the multiarmed bandit problem. *Machine Learning*, 47:235–256, 2002.

[7] J. Ben Schafer, J. A. Konstan, and J. Riedl. E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5:115–153, 2001.

[8] D. Bergemann and D. Ozmen. Optimal pricing with recommender system. In *ACM Conference on Electronic Commerce*, pages 43–51, 2006.

[9] H. Daumé, III. Frustratingly easy domain adaptation. In *Proc. of the 45th Annual Meeting of the Association of Computational Linguistics*, pages 256–263, 2007.

[10] N. Japkowicz. Learning from imbalanced data sets: A comparison of various strategies. In *AAAI Workhop: Learning from Imbalanced Data Sets*, pages 10–15, 2000.

[11] T. Kamishima and S. Akaho. Nantonac collaborative filtering: A model-based approach. In *Proc. of The 4th ACM conference on Recommender Systems*, pages 273–276, 2010.

[12] L. Li, W. Chu, J. Langford, and R. E. Schapire. A contextual-bandit approach to personalized news article recommendation. In *Proc. of The 19th Int'l Conf. on World Wide Web*, pages 661–670, 2010.

[13] P. Resnick, N. Iacovou, M. Suchak, P. Bergstrom, and J. Riedl. GroupLens: An open architecture for collaborative filtering of Netnews. In *Proc. of The Conf. on Computer Supported Cooperative Work*, pages 175–186, 1994.

[14] G. Shani, D. Heckerman, and R. I. Brafman. An mdp-based recommender system. *Journal of Machine Learning Research*, 6:1265–1295, 2005.

[15] N. Stokey. Intertemporal price discrimination. *The Quarterly J. of Economics*, 93(3):355–371, 1979.

[16] D. Streitfeld. On the web, price tags blur: What you pay could depend on who you are. Washington Post, Sep. 27 2000. `http://www.washingtonpost.com/ac2/wp-dyn/A15159-2000Sep25`.

[17] R. S. Sutton and A. G. Barto. *Reinforcement Learning: An Introduction*. MIT Press, 1998.

[18] N. Terui and W. D. Dahana. Price customization using price thresholds estimated from scanner panel data. *Journal of Interactive Marketing*, 20:58–70, 2006.

[19] L.-T. Weng, Y. Xu, Y. Li, and R. Nayak. Towards information enrichment through recommendation sharing. In L. Cao, editor, *Data mining and Multi-agent Integration*, chapter 7, pages 103–125. Springer, 2009.