Mining the Customer's Up-To-Moment Preferences for E-Commerce Recommendation

Yi-Dong Shen[1], Qiang Yang[2] Zhong Zhang[3], and Hongjun Lu[2]

[1] Lab. of Comp. Sci., Institute of Software, Chinese Academy of Sciences, China
ydshen@ios.ac.cn
[2] Hong Kong University of Science and Technology, Hong Kong, China
{qyang, luhj}@cs.ust.hk
[3] School of Computing Science, Simon Fraser University, Burnaby, Canada
zzhang@cs.sfu.ca

Abstract. Most existing data mining approaches to e-commerce recommendation are past data model-based in the sense that they first build a preference model from a past dataset and then apply the model to current customer situations. Such approaches are not suitable for applications where fresh data should be collected instantly since it reflects changes to customer preferences over some products. This paper targets those e-commerce environments in which knowledge of customer preferences may change frequently. But due to the very large size of past datasets the preference models cannot be updated instantly in response to the changes. We present an approach to making real time online recommendations based on an up-to-moment dataset which includes not only a gigantic past dataset but the most recent data that may be collected just moments ago.

1 Introduction

E-commerce recommendation is aimed at suggesting products to customers and providing consumers with information to help them decide which products to purchase. Several approaches have been applied to making such recommendations, such as nearest-neighbor collaborative filtering algorithms [7], Bayesian networks [4], classifiers [3], and association rules [9]. Observe that most existing recommendation approaches are *past data model-based* in the sense that they walk through the following process:

Historical Data $\Rightarrow$ Preference Model $\Rightarrow$ Preferences $\Rightarrow$ Recommendation.

That is, they first build a preference model from a set of historical transaction data. Then they measure customer preferences over a set of products based on the model. The preferences establish a partial order on the products. Therefore, recommendation is made by choosing products with top preferences. For example, Bayesian approach builds its model – a Bayesian network, by learning the prior and posterior probabilities of products from the given dataset. Then, given a customer's current shopping information, the customer's preferences over some

products are derived by computing the posterior probabilities of these products. Similarly, the association-rule approach takes a set of association rules as its model. An association rule [1] is of the form $p_1, ..., p_m \rightarrow p_0$ $(s\%, c\%)$, which, in the context of super-marketing, may mean that there would be $c\%$ possibility that a customer who has bought products $p_1, ..., p_m$ will buy the product p_0. Let M be a preference model and $\{p_1, ..., p_m\}$ be the set of products in a customer's current shopping cart. p_0 is a *candidate* product for recommendation to the customer if M has an association rule of the form $p_1, ..., p_m \rightarrow p_0$ $(s\%, c\%)$. p_0 will be recommended to the customer if the confidence $c\%$ is among the top of all candidate products.

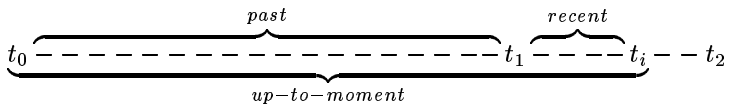
Apparently, the bottleneck of past data model-based approaches is in the development of their models. In many e-commerce applications, the historical dataset would be very large and it may take a long time to build a model from it. This suggests that the model needs to be built off-line (or in batch). As a result, such approaches are suitable only for applications where knowledge of customer preferences is relatively stable.

However, in many e-commerce applications the business patterns may change frequently. In such situations, new precious data should be collected instantly since it reflects changes (increase or decrease) to customer preferences over some products. Since past data model-based approaches are unable to catch up with such frequent changes, their models may not fully cover the up-to-moment status of the business, and therefore applying them alone would lower down the recommendation accuracy.

The above discussion suggests that methods of making use of up-to-moment (not just up-to-date) data need to be developed for e-commerce recommendation. Such methods should keep abreast of the up-to-moment changes to customer preferences. This motivates the research of this paper.

1.1 Problem Statement

The problem can be generally stated as follows. Let SM_{past} be a model built from the data which was collected between time points t_0 and t_1 as depicted below. Assume that no updated model is available until time t_2 and that there would be a certain amount of fresh data available during the period $t_2 - t_1$. Suppose a customer logs into the system for online shopping and asks for product recommendation at time t_i $(t_1 < t_i < t_2)$. For convenience, the data collected between t_0 and t_1 is called *past data*, whereas the data collected between t_1 and t_i is called *recent data*. We are then asked to design a system that can make real time online recommendations based on the *up-to-moment data* which includes both the past and recent data. By "real time" we mean a response time with which customers can tolerate during online shopping (e.g., a few seconds).



Here are the characteristics and challenges of the problem: (1) The past data would be massive, say Megs or Gigs of records, so that it cannot be directly used in real time. A model needs to be built off-line (or in batch) from the data. Model building is quite time-consuming, though applying a model to make recommendations can be done in real time. (2) The volume of the recent data is much smaller than that of the past data, possibly in the hundreds of orders. For example, in our super-store data mining project, the past dataset always contains at least three months of past transaction data whereas the recent dataset only contains the data of the current day. The recent data was most recently generated and includes some data that was collected just moments ago. (3) Thus we are faced with the following challenge in applying the up-to-moment data. The dataset is so huge that we cannot apply it to make online real time recommendations unless a model is built from it off-line or in batch in advance. However, in reality the time between the most recent data being collected and the customer requesting a recommendation may be very short (possibly less than one second), so it is impossible for us to prepare such a model in advance or build it instantly. Thus it appears that we are in a dilemma situation. Effectively resolving such a dilemma then presents a challenging task.

1.2 Our Solution

In its most general case, this problem seems too difficult to be tractable. Our study shows, however, that it could be effectively handled when some practical constraints are applied. We choose association rules as our preference model. Since the population of the recent data within the up-to-moment data is much smaller than that of the past data, statistically the recent data can only bring minor changes (increase or decrease) to the past preferences. In other words, the recent data can only play a role of small adjustments to the customer's past preferences in response to the recent minor changes. This implies that the large majority of the supporting records of any up-to-moment frequent pattern may come from the past data. Therefore, in this paper we restrict ourselves to these situations where any up-to-moment frequent pattern occurs in at least one record of the past data.

The key idea of our approach stems from the following important observation: A few patterns that do not frequently occur in the past data may become frequent in the up-to-moment data, whereas a few other patterns that frequently occur in the past data may not be frequent in the up-to-moment data. However, due to the rather small population of the recent data within the up-to-moment data, the preferences of all such inversable patterns in the past data must be close to the minimum preference that is required for recommendation.

Based on the above observation, we introduce a concept of *expanded past preference models*, as opposed to the standard preference models [1]. An expanded past preference model is built off-line from the past data and contains not only standard association rules but also a few expanded association rules that represent those patterns that are less frequent in the past data but quite likely to be frequent in the up-to-moment data. Then, given the customer's current shopping

information, we apply a criterion, called *Combination Law*, to derive the customer's up-to-moment preferences directly from the expanded past model and the recent data. Let SM_{utm} denote a standard preference model built from the up-to-moment data. We will prove that applying our approach will generate the same recommendations to the customer as SM_{utm} does, though we do not need to build SM_{utm} at all.

1.3 Related Work

For typical recommendation approaches and applications, see the 1997 special issue of the Communications of the ACM [10] and a recent special issue of Data Mining and Knowledge Discovery [8]. In particular, Shafer et al. [11] made an elegant survey on major existing systems and approaches to e-commerce recommendation. To the best of our knowledge, however, there is no report in the literature on how to use the up-to-moment data to make real time recommendations. Most existing approaches are past data model-based, which make recommendations by first building a (standard) past preference model off-line against a previously prepared training dataset and then applying the model to current customer situations. The model is updated periodically. As a typical example, the SmartPad system developed at IBM makes recommendations using a model of association rules [9]. The model is built on the past eight weeks of data from Safeway Stores and is updated weekly or quarterly to reflect seasonal differences.

Our work focuses on doing interactive and online preference mining (in real time). So it is essentially different from the incremental update of association rules, a topic that addresses how to efficiently update a (standard) past preference model that was built from an old dataset (DB) based on an incremental dataset (db) [2, 5, 6, 12]. Existing incremental updating algorithms make use of some knowledge of an old model to build a new model from $DB \cup db$. Although applying these algorithms to update a model is faster than building a new model from scratch, it is still quite time-consuming since scanning the old dataset cannot be avoided. When DB is massive, such update cannot be completed online in real time. Moreover, if we applied an incremental updating procedure to mine the up-to-moment preferences, we would have to update the model every time a new record is added to the dataset, which is quite unrealistic in real online processing.

2 System Architecture

We assume readers are familiar with association rule mining, especially with the widely used Apriori algorithm [1]. A *dataset* DS is a set of records each of which is of the form $\{p_1, ..., p_m\}$. We use $|DS|$ to denote the total number of records in DS and $count(\{p_1, ..., p_m\}, DS)$ to denote the number of records in DS that contain all the items $p_1, ..., p_m$. A (standard) *preference model* built from DS consists of a set of association rules. A set of items, $\{p_1, ..., p_m\}$, is a *frequent*

pattern in DS if its support $s\% = \frac{count(\{p_1,p_2,...,p_m\},DS)}{|DS|} \geq ms\%$, where $ms\%$ is a pre-specified parameter called *minimum support*. Let $\{p_1, ..., p_m\}$ be a frequent pattern. For each $i \leq m$, $p_1, ..., p_{i-1}, p_{i+1}, ..., p_m \rightarrow p_i$ $(s\%, c\%)$ is an *association rule* in the preference model if $c\% \geq mc\%$, where $mc\%$ is another pre-specified parameter called *minimum confidence*, and $c\%$ is the confidence of the rule given by $c\% = \frac{count(\{p_1,...,p_m\},DS)}{count(\{p_1,...,p_{i-1},p_{i+1},...,p_m\},DS)}$. We will frequently use three terms: past, recent and up-to-moment. Let x be one of them. Here is some of the notation used throughout this paper. $ms\%$: the minimum support; $mc\%$: the minimum confidence; DS_x: an x dataset; SM_x: a standard x preference model built from DS_x with $ms\%$ and $mc\%$; and EM_{past}: an expanded past preference model built from DS_{past} with $ms\%$ and $mc\%$.

The architecture of our recommendation system is shown in Figure 1. The system works on two datasets: the *past dataset* DS_{past} that stores the past data, and the *recent dataset* DS_{recent} that stores the recent data. DS_{past} is much larger than DS_{recent}, possibly in the hundreds of orders. The up-to-moment dataset $DS_{utm} = DS_{past} \cup DS_{recent}$. Since DS_{past} is gigantic, the system builds off-line or in batch an expanded past preference model from it (see Definition 1 in the following section). Then, given a customer's current shopping information,[4] the customer's up-to-moment preferences are computed online by combining the knowledge about the customer's past preferences that is derived from the expanded past model EM_{past} and the knowledge about his/her recent preferences that is derived from DS_{recent}. When a customer finishes shopping by logging out the system, a new record built from the products in the customer's shopping cart is added to DS_{recent} immediately. Since we only keep most recent data in DS_{recent}, the "old" data in DS_{recent} will be moved to DS_{past} from time to time. Moreover, when DS_{past} gets too big, some data will be removed. All such data transfers are monitored by the Database management facility.

3 Building an Expanded Past Model

Our goal is to compute the customer's up-to-moment preferences without building an up-to-moment preference model. To this end, we need to build a so-called expanded past preference model EM_{past} from DS_{past}. Two factors are considered in developing such an expanded model: *completeness* and *efficiency*. EM_{past} is complete if for any standard association rule $X \rightarrow Y(s\%, c\%)$ that is derivable from DS_{utm}, there is an expanded association rule $X \rightarrow Y(s'\%, c'\%)$ in EM_{past}. One might consider defining EM_{past} as a standard preference model which is built from DS_{past} by setting the minimums $ms\% = 0$ and $mc\% = 0$. Although doing so can guarantee the completeness, it is definitely too inefficient, if not impossible, because of the nature of the association mining problem [1]. The efficiency factor then requires that both the number of rules in EM_{past} and the time to build EM_{past} be as close as possible to those of SM_{past} – the standard

[4] It may contain the customer's profile as well as the products in his current shopping cart. To simplify our presentation, we omit the profile.

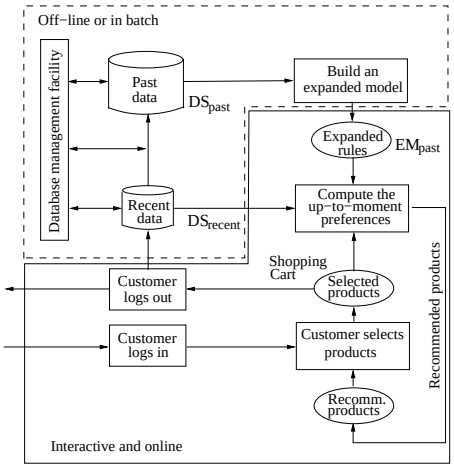


Fig. 1. A recommendation system based on the up-to-moment data.

past preference model built from DS_{past} with the pre-specified minimums. In particular, EM_{past} is said to be *scalable* if the larger the past dataset DS_{past} is, the closer EM_{past} is to SM_{past} w.r.t. the number of their rules and the running time.

In order to get high efficiency while guaranteeing the completeness, we introduce a new parameter, $\mathcal{N}_{max}$, called *one day's maximum*. Intuitively, $\mathcal{N}_{max}$ represents the one day's maximum sales of a single product, which can be obtained from past experience. More formally, let P be the set of all products, then at any time we have $\mathcal{N}_{max} \geq max\{count(\{p\}, DS_{recent}) | p \in P\}$. It is based on this new parameter that our expanded model is defined. In the sequel, both $\mathcal{N}_{max}$ and the two minimums $ms\% > 0$ and $mc\% > 0$ are supposed to be provided by the e-commerce manager.

Definition 1. *An* expanded past preference model EM_{past} *of* DS_{past} *consists of all expanded association rules defined as follows: (1)* $\{p_1, ..., p_m\}$ *is an* expanded frequent pattern *if* $count(\{p_1, ..., p_m\}, DS_{past}) > 0$ *and* $\frac{count(\{p_1,...,p_m\}, DS_{past}) + \mathcal{N}_{max}}{|DS_{past}| + \mathcal{N}_{max}} \geq ms\%$. *(2) Let* $\{p_1, ..., p_m\}$ *be an expanded frequent pattern. For each* $i \leq m$, $p_1, ..., p_{i-1}, p_{i+1}, ..., p_m \rightarrow p_i$ $(s\%, c\%)$ *is an* expanded association rule *if* $\frac{count(\{p_1,...,p_m\}, DS_{past}) + \mathcal{N}_{max}}{count(\{p_1,...,p_{i-1},p_{i+1},...,p_m\}, DS_{past}) + \mathcal{N}_{max}} \geq mc\%$ *where* $s\% = \frac{count(\{p_1,...,p_m\}, DS_{past})}{|DS_{past}|}$ *and* $c\% = \frac{count(\{p_1,...,p_m\}, DS_{past})}{count(\{p_1,...,p_{i-1},p_{i+1},...,p_m\}, DS_{past})}$.

Note that the support $s\%$ and confidence $c\%$ of an expanded association rule are the same as those defined for a standard association rule. However, due to the inclusion of $\mathcal{N}_{max}$, the thresholds for an expanded frequent pattern or an expanded association rule are a little lower than the two pre-specified minimums. This allows EM_{past} to catch those patterns/association rules that

are unable to be included in SM_{past} but may possibly appear in SM_{utm}. We have the following.

Theorem 1. *EM_{past} is complete.*

The following theorem shows that any standard association rule in SM_{past} is an expanded association rule in EM_{past}.

Theorem 2. *$SM_{past} \subseteq EM_{past}$.*

The following result assures us that the expanded association rules can be derived using standard association mining methods such as Apriori.

Theorem 3. *Any sub-pattern of an expanded frequent pattern is an expanded frequent pattern.*

Before moving to the next section, we briefly answer the question readers may pose about how to keep $\mathcal{N}_{max}$ to be the maximum count in DS_{recent}. This can be realized simply by treating DS_{recent} in batches. If at time t_i there is a product in DS_{recent} whose sales are about to reach the maximum $\mathcal{N}_{max}$ (10% below $\mathcal{N}_{max}$ in our project. The exact measure depends on applications), we add the data of DS_{recent} to the past dataset DS_{past} and build in batch a new expanded past preference model from DS_{past} in advance. As a result, when it happens that the sales of an item in DS_{recent} exceed $\mathcal{N}_{max}$, the new expanded past model must be available, which covers the past data collected up to time t_i. Thus the new expanded past model EM_{past}, together with a new recent dataset DS_{recent} that consists of the data collected from t_i up to the current moment, is used to compute the customer's up-to-moment preferences.

4 Computing the Up-To-Moment Preferences

Let $SC = \{p_1, ..., p_m\}$ be the set of products in a customer's current shopping cart. We make online product recommendations for the customer by computing his up-to-moment preferences over those products related to SC. That is, we derive from EM_{past} and DS_{recent} all rules of DS_{utm} of the form $p_1, ..., p_m \rightarrow p$ $(s\%, c\%)$ with $s\% \geq ms\%$ and $c\% \geq mc\%$. This is done via the three steps summarized in the following table.

	Given $SC = \{p_1, ..., p_m\}$.
Step 1:	Derive R_{past} (rules about the customer's past preferences) from EM_{past}.
Step 2:	Derive R_{recent} (rules about the customer's recent preferences) and $count_{p_1-pm}$ from DS_{recent} based on R_{past} (see Algorithm 1).
Step 3:	Combine R_{past} and R_{recent} to obtain PL_{utm} (the customer's up-to-moment preferences; see Algorithm 2).

The first step is to extract from EM_{past} the knowledge about the customer's past preferences over those products related to SC. We search EM_{past} to obtain

the set R_{past} of rules of the form $p_1, ..., p_m \rightarrow p$ $(s\%, c\%)$. By Theorem 1, all rules of DS_{utm} related to SC are in R_{past}. The second step is to extract from DS_{recent} the knowledge about the customer's recent preferences related to SC. By Theorem 1, it is unnecessary to derive from DS_{recent} all rules with a left-hand side $p_1, ..., p_m$; it suffices to derive only the set R_{recent} of rules each of which is of the form $p_1, ..., p_m \rightarrow p$ $(s_2\%, c_2\%)$ such that there is a rule of the form $p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$ in R_{past}. Such new rules reflect changes to the customer's past preferences over those products related to SC. Here is an algorithm.

Algorithm 1 Extracting rules about the customer's recent preferences.
Input: SC, DS_{recent} and R_{past}.
Output: R_{recent} and $count_{p_1-p_m} = count(SC, DS_{recent})$.
Procedure:
1. $R_{recent} = \emptyset$;
2. $Head = \{p | p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$ is in $R_{past}\}$;
3. Select all records containing $p_1, ..., p_m$ into $TEMP$ from DS_{recent};
4. **for** each product $p \in Head$ **do begin**
5. Compute $count_p = count(\{p\}, TEMP)$;
6. **if** $count_p > 0$ **then do begin**
7. $s_2\% = \frac{count_p}{|DS_{recent}|} * 100\%$; $c_2\% = \frac{count_p}{|TEMP|} * 100\%$;
8. Add the following rule to R_{recent}: $p_1, ..., p_m \rightarrow p(s_2\%, c_2\%)$
9. **end**
10. **end**
11. Return R_{recent} and $count_{p_1-p_m}$ with $count_{p_1-p_m} = |TEMP|$.

Although both the support and confidence of a rule are computed in Algorithm 1 (see line 7), neither the minimum support nor the minimum confidence is used because we are not ready to remove any rules at this stage. We will do it next in conjunction with those rules about the past preferences.

Theorem 4 (Combination Law). *SM_{utm} has a rule $p_1, ..., p_m \rightarrow p$ $(s_3\%, c_3\%)$ if and only if one of the following holds: (1) There are two rules $p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$ in R_{past} and $p_1, ..., p_m \rightarrow p$ $(s_2\%, c_2\%)$ in R_{recent}. Then $s_3\% = \frac{s_1\%*|DS_{past}|+s_2\%*|DS_{recent}|}{|DS_{utm}|} \geq ms\%$ and $c_3\% = \frac{s_1\%*|DS_{past}|+s_2\%*|DS_{recent}|}{\frac{s_1*|DS_{past}|}{c_1}+count_{p_1-p_m}} \geq mc\%$. (2) Only R_{past} has a rule $p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$. Then $s_3\% = \frac{s_1\%*|DS_{past}|}{|DS_{utm}|} \geq ms\%$ and $c_3\% = \frac{s_1\%*|DS_{past}|}{\frac{s_1*|DS_{past}|}{c_1}+count_{p_1-p_m}} \geq mc\%$.*

Theorem 4 is quite significant. It allows us to derive the customer's up-to-moment preferences directly from R_{past} and R_{recent}, without building any up-to-moment preference model. In the following algorithm, by CL(i) we refer to the i-th case of the Combination Law ($i = 1$ or 2).

Algorithm 2 Deriving the customer's up-to-moment preferences.
Input: R_{past}, R_{recent}, $|DS_{past}|$, $|DS_{recent}|$, $|DS_{utm}|$, $count_{p_1-p_m}$, $ms\%$ and $mc\%$.

Output: an up-to-moment preference list, PL_{utm}.
Procedure:
1. $PL_{utm} = \emptyset$;
2. **while** $(R_{past} \neq \emptyset)$ **do begin**
3. Remove from R_{past} a rule $p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$;
4. **if** R_{recent} has a rule $p_1, ..., p_m \rightarrow p$ $(s_2\%, c_2\%)$ **then**
5. Compute $s_3\%$ and $c_3\%$ by applying CL(1);
6. **else** Compute $s_3\%$ and $c_3\%$ by applying CL(2);
7. **if** $s_3 \geq ms$ and $c_3 \geq mc$ **then**
8. $PL_{utm} = PL_{utm} \cup \{p\ (s_3\%, c_3\%)\}$
9. **end**
10. Return PL_{utm} which is sorted by confidence.

Algorithm 2 computes the customer's up-to-moment preferences based on the Combination Law. In particular, it computes the up-to-moment preferences by applying the recent preferences to adjusting the past preferences. Let $PP = p\ (s_1\%, c_1\%)$ be a past preference item; i.e. R_{past} has a rule $p_1, ..., p_m \rightarrow p$ $(s_1\%, c_1\%)$. There may be three cases. First, both s_1 and c_1 are sufficiently high to survive any recent changes, so that the product p is guaranteed to be included in the up-to-moment preference list PL_{utm}. Second, PP is *positively marginal* in the sense that s_1 or c_1 is just a little higher than ms or mc. In this situation, the recent changes may invalidate the past preference by lowering it down below either of the two minimums so that the product p is excluded from PL_{utm}. The third case is that PP is *negatively marginal*; i.e. either s_1 or c_1 is a little lower than ms or mc. In this case, the past preference may be lifted above the minimums by the recent changes so that the product p is included in PL_{utm}.

Theorem 5. *Algorithm 2 is correct in the sense that for any customer's current shopping cart* $SC = \{p_1, ..., p_m\}$, SM_{utm} *has a rule* $p_1, ..., p_m \rightarrow p$ $(s_3\%, c_3\%)$ *if and only if* p $(s_3\%, c_3\%) \in PL_{utm}$.

5 Experimental Evaluation

The correctness of our approach is guaranteed by Theorem 5. We now show its efficiency by empirical experiments. Our experiments were conducted on a Pentium III 600 MHz PC with 512M memory. We applied the Apriori algorithm to mine association rules with the minimums $ms\% = mc\% = 1\%$.

Our dataset for the experiments came from a retail store. The dataset consists of eight weeks of daily transaction data. To facilitate our experiments, we fixed a past dataset DS_{past} that consists of 687571 records. We then collected five recent datasets DS^1_{recent} (4415 records), DS^2_{recent} (5830 records), DS^3_{recent} (7245 records), DS^4_{recent} (8660 records), and DS^5_{recent} (10077 records) at five consecutive sampling times $T_1, ..., T_5$. As a result, we got five up-to-moment datasets $DS^1_{utm}, ..., DS^5_{utm}$ with $DS^i_{utm} = DS_{past} \cup DS^i_{recent}$.

Since DS_{recent} is small, computing the recent preferences (the second step of our approach) takes very little time because it is accomplished simply by

counting the frequency of each single product in DS_{recent} (see Algorithm 1). It takes even less time to extract the past preferences from EM_{past} (the first step) and do the combination of the past and recent preferences (the third step; see Algorithm 2). Experiments for such claims were conducted by simulating a customer's shopping process in which we assumed that a customer asked for product recommendation at the times T_1, T_2, ... and T_5, respectively.

At time T_1 the customer's shopping cart was empty. Our system computed the up-to-moment preferences and made a recommendation. The customer selected a product from our recommendation and then asked for more at time T_2. The process then went to the next cycle. For each T_i we recorded the response time for the period of the customer presenting the request and the system sending back the recommendation. Figure 2 shows the response time of our approach in contrast to the response time of the *brutal-force* approach that directly applies Apriori to derive the up-to-moment preferences. Clearly, using our approach takes a very short response time (1.38 seconds on average), while using the brutal-force approach spends 86.7 seconds on average.

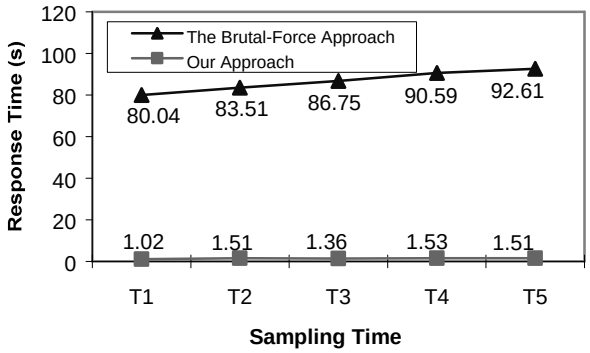


Fig. 2. The response time at five consecutive sampling times.

The above experiments demonstrate that the customer's up-to-moment preferences can be derived in real time by performing the three steps of our approach, provided that the expanded past model is prepared in advance. We now show the efficiency of building an expanded model EM_{past} as compared to building a standard past model SM_{past}. As we mentioned before, this can be measured by showing how close the two models are w.r.t. the number of their rules and the time to build them. In order to see the scalability of our approach, we built from DS_{past} five datasets as past datasets, with DS_1 (269K records) consisting of the data of the first three weeks, DS_2 (372K records) of the first four weeks, ..., and DS_5 (690K records) of the seven weeks. Moreover, by randomly picking up ten days among the seven weeks and counting the sales of each single item in each selected day, we got the number 1217 for the one day's maximum parameter $\mathcal{N}_{max}$.

The comparison of the running time and the number of rules is depicted in Figures 3 and 4, respectively. We see that it does not take much more time to build an expanded past model EM^i_{past} from DS_i than to build a standard past model SM^i_{past} from DS_i (see Figure 3), and that although EM^1_{past} is about two times bigger than SM^1_{past}, EM^5_{past} is very close to SM^5_{past} (see Figure 4). This shows our approach is efficient and has very good scalability. That is, the bigger the past dataset is, the closer the expanded model is to the standard model.

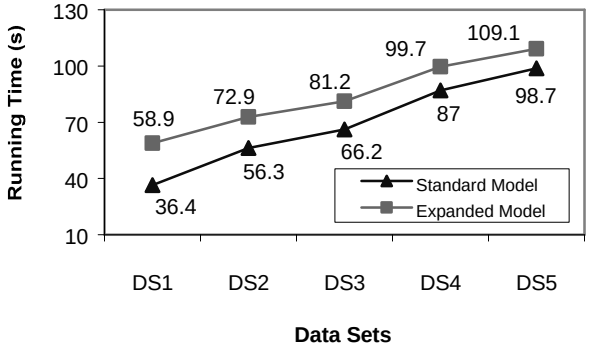


Fig. 3. Comparison of the running time of building an expanded and a standard past preference model.

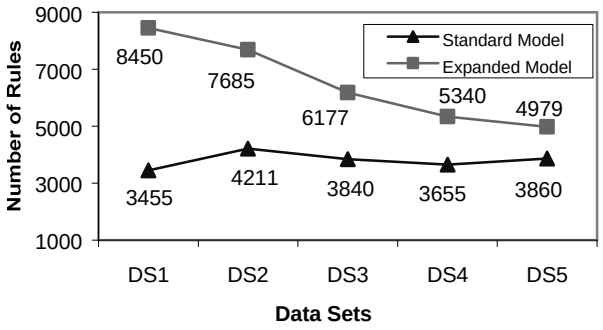


Fig. 4. Comparison of the number of rules of an expanded and a standard past preference model.

6 Conclusions

We have developed an approach to making online e-commerce recommendations based on the up-to-moment data. A key concept of this approach is an expanded

past preference model from which the customer's up-to-moment preferences can be derived in real time without the necessity to build an up-to-moment model. Our approach is suitable for applications where knowledge of customer preferences may change frequently but due to the huge size of past datasets the preference models cannot be updated instantly in response to the changes. Therefore, it is an enhancement to existing past data model-based approaches which are suitable only for applications where customer preferences are relatively stable.

Acknowledgment

The authors thank the anonymous referees for their helpful comments. Yi-Dong Shen is supported in part by Chinese National Natural Science Foundation, Trans-Century Training Program Foundation for the Talents by the Chinese Ministry of Education, and Foundations from Chinese Academy of Sciences. Qiang Yang thanks NSERC, IRIS-III program and Hong Kong RGC for their support.

References

1. R. Agrawal and R. Srikant. Fast algorithm for mining association rules. In *VLDB*, pages 487–499, 1994.
2. Y. Aumann, R. Feldman, O. Lipsttat, and B. H. Manilla. An efficient algorithm for association generation in dynamic databases. *Journal of Intelligent Information Systems*, 12:61–73, 1999.
3. C. Basu, H. Hirsh, and W. Cohen. Recommendation as classification: using social and content-based information in recommendation. In *Proceedings of the National Conference on Artificial Intelligence*, pages 714–720, 1998.
4. J. Breese, D. Heckerman, and C. Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence*, pages 43–52, 1998.
5. D. Cheung, J. Han, V. Ng, , and C. Wong. Maintenance of discovered association rules in large databases: an incremental updating technique. In *Proc. of 12th Intl. Conf. on Data Engineering*, pages 106–114, 1996.
6. C. Hidber. Online association rule mining. In *SIGMOD*, pages 145–156, 1999.
7. J. Herlocker, J. Konstan, A. Borchers, and J. Riedl. An algorithmic framework for performing collaborative filtering. In *ACM SIGIR*, pages 230–237, 1999.
8. R. Kohavi and F. Provost. Applications of data mining to electronic commerce. *Journal of Data Mining and Knowledge Discovery*, 5:5–10, 2001.
9. R. Lawrence, G. Almasi, V. Kotlyar, M. Viveros, and S. Duri. Personalization of supermarket product recommendations. *Journal of Data Mining and Knowledge Discovery*, 5:11–32, 2001.
10. P. Resnick and H. Varian. Recommender systems. *Communications of the ACM*, 40:56–58, 1997.
11. J. Schafer, J. Konstan, and J. Riedl. Electronic commerce recommender applications. *Journal of Data Mining and Knowledge Discovery*, 5:115–152, 2001.
12. S. Thomas, S. Bodagala, K. Alsabti, and S. Ranka. An efficient algorithm for the incremental updation of association rules in large databases. In *KDD*, pages 263–266, 1997.