International Journal of Electronic Commerce Studies
Vol.6, No.1, pp.1-18, 2015
doi: 10.7903/ijecs.1348

DEVELOPING A PRICE-SENSITIVE RECOMMENDER SYSTEM TO IMPROVE ACCURACY AND BUSINESS PERFORMANCE OF ECOMMERCE APPLICATIONS

Panniello Umberto
Polytechnic of Bari
Viale Japigia 182, 70126, Bari, Italy
umberto.panniello@poliba.it

ABSTRACT

Much work has been done on recommender systems (RS) and much evidence was collected from applications about their effectiveness on business. As a consequence, the use of RS has quickly shifted from information retrieval to automatic marketing tools. The main aim of marketing tools is to positively affect customers' purchasing decisions and we know through marketing literature that purchasing decisions are strongly influenced by price. However, few works have explored the issue of including price in a recommendation engine. In this paper, we want to describe the main issues of designing this type of price-sensitive recommendation engine. We want also to demonstrate what the effect is of this design on recommendations' accuracy and on business performance. We demonstrate that including price in an RS improves the accuracy of recommendations, but it has to be properly modeled in order to also improve business performance. We have experimented with a Price-Sensitive RS in a laboratory setting and compared it to a traditional one by varying several settings.

Keywords: Recommender Systems, Multidimensional, Marketing, Price Sensitivity

1. INTRODUCTION

Recommender systems (RS) deliver personalized product recommendations to users. They are widely used by online retailers such as Amazon and Netflix. RS benefit both consumers and firms. They help consumers become aware of new products and influence their selection of

desirable products from a myriad of choices[1]. They have the potential to help firms increase their profits by converting browsers into buyers, increasing customer loyalty and company sales[2, 3]. RS were born as information retrieval tools, but they have quickly become interesting for business purposes.

Despite the wide use of RS in online business and despite much evidence that using RS can positively influence a customer's purchasing behavior, product price information is not included in the design of an RS. Some research has explored related issues. For instance, some scholars have studied business-centered designs of RS to aim at increasing business goals together with customer relevance[4]. Another relevant area is the personalization of price. In this case, RS are designed in order to suggest price rather than products[5]. Very little research has been done on the issue of including price in an RS design despite early research[6]. The lack of research may be explained by observing that including price in RS design is a challenging task because it means modeling customers' price sensitivity, which varies on both customers and items' categories.

In this paper, we aim at filling this gap. First, we want to demonstrate that modeling price is critical because it has a direct impact on the business performance of a company delivering recommendations. Second, we want to demonstrate that it is possible to use the price variable in a way that the overall business performance increases. In particular, we describe the issues of designing a price-sensitive recommender system via a Multi-Dimensional approach.

It would be very important for companies to adopt price-sensitive recommendation engines. An effective RS should recommend to a customer the most relevant products at the right prices. Recommending low-priced items to users that prefer to spend more money for that product reduces the business performance. On the other hand, recommending high-priced items to users that like to spend little money reduces the effectiveness of the recommendation.

2. PRIOR WORK

RS were first designed as tools to support information retrieval. Following this hallmark, the bulk of research on RS is focused on designing algorithms able to improve recommendation accuracy (see Adomavicius and Tuzhilin[7] and Ricci et al.[8] for a review).

Much research has been done on studying other aspects which are as important as accuracy from a business point of view. There has been much

research on this perspective since Schafer et al.[6] challenged scholars to design an RS to both maximize customer utility and business value at the same time. Some scholars studied what factors affect an RS adoption by customers[9, 10]. Taking a few steps ahead along this direction, some scholars have studied how RS affect certain aspects of customers' behavior even closer to business goals, such as intentions and attitudes. For instance, Pu et al.[11] found that ease of use and perceived usefulness is important for usage intentions, while trust and choice confidence are crucial for purchasing intentions. Bharati and Chaudhury[12] studied how relevance, accuracy, completeness, and timeliness of recommendations have a significant effect on users' decision making and satisfaction.

Other scholars have directly investigated the economic effect of RS usage on the business. Schafer et al.[2] argued that recommender systems help increase sales by converting browsers into buyers, increasing cross-selling opportunities, and building customer loyalty. Fleder and Hosanagar[13] demonstrated that RS that discount item popularity in the selection of recommendable items may increase sales more than RS that do not. Pathak et al.[14] found that the strength of recommendations has a positive impact on sales. Gorgoglione et al.[3] showed the effect of recommendations on customers' purchasing behavior and business sales.

Despite much research, as reviewed above, several issues remain relatively unexplored. One of these issues is how to include price information and customer price sensitivity to improve the relevance of recommendations. It would be of obvious importance to include price in the recommendation process because price is one of the most important drivers of customer behavior and purchasing decisions. Several scholars have stated that including price would be important to RS effectiveness. Schafer et al.[6] stated that the RS could produce an indication of the price sensitivity of the customer for a given product. Burke[15] stated that utility-based RS can build a customer preference function by including features such as price, quality, and delivery date. So far, research has explored related issues, such as personalized pricing. Particularly, Bergemann and Ozmen[16] assumed that the task of defining the price of each item is external to the recommender system and gave some insights on how to manage this task when using (or not) an RS. Kamishima and Akaho[5] assumed that items' pricing can be managed with an automatic tool, such as an RS, and proposed an RS that discount the items maximizing the probability that the customer will buy it.

All these efforts show that there is an increasing interest in integrating marketing variables, such as the items' pricing, with the RS, but this connection is still not explored. In fact, items' pricing is managed by a firm's management and it is based on human decisions or on marketing

tools (such as RFM models). The price definition is correlated with several external constraints (i.e., competitors' price, items' cost, periodicity) and, even if the RS suggests to discount a specific item, the decision on discounting it or not is due to the management. Therefore, the design of models, such as the one proposed by Kamishima and Akaho[5], are difficult to be used or, at least, they require a second step (the acceptance/discard of the recommendations list by the management). In addition, works such as Bergemann and Ozmen[16] demonstrate that the effects of items' price modifications performed by management decisions on the output of a recommender system are not trivial. On the one hand, recommending low price items to users who could spend more money reduces business performance. On the other hand, recommending high price items to users that would like to spend a little money, reduces the efficacy of the recommender system.

3. INCLUDING PRICE IN A RECOMMENDER ENGINE

The goal of this section is to describe how the prices of products can be included in a recommendation engine. Even if many recommendation algorithms do exist in previous literature, we focus on the three most popular ones and we select the most promising among them.

A Content-Based (CB) recommendation engine recommends items similar to the ones the user preferred in the past. Similarity among items is computed using a set of features describing the items. Let *ItemProfile(s)* for item *s* and *UserProfile(i)* for user *i*, be two vectors representing the item characteristic and the customer preference, respectively. *ItemProfile(s)* is computed by extracting a set of keywords from a set of features describing the item, while *UserProfile(i)* is computed by analyzing the content of the items previously rated by user *i*. The content-based system matches the *UserProfile(i)* with *ItemProfile(s)* of the recommendable items and it recommends the most fitting items. Price may be included in the set of keywords. As largely demonstrated by prior research, CB recommender engines suffer from the "over specialization" problem[17] consisting of the fact that the system can only recommend items scoring high against a user's profile. Including price would boost this issue and the user would receive only recommendations of items similar to those already rated in the past with a specific price.

An alternative to CB approach is the knowledge-based approach, particularly Utility-Based (UB) recommender systems[15]. A UB engine generates recommendations based on the computation of utility of each item for each user. Utility is computed as a function of a set of features

describing an item. An utility-based recommender system applies a utility function *f(i)* over the set of all the recommendable items, *S*, thus determining the list of recommended items. If the relevance of a certain item is high, then that item is recommended to the user. Price may be included as one of the features describing products. This type of recommender systems have shown several limitations. They require users to define their utility function (i.e., users have to spend a lot of time giving a very high number of ratings) and the adaptation of this type of RS to users' preference changes is not immediate.

We propose the use of multi-dimensional (MD) collaborative filtering (CF) to include price in a recommender engine. A CF approach recommends items based on the similarity among users. Suppose a set of users share similar tastes and preferences with user *i*. If this set of users liked certain products in the past, then those products are recommended to user *i*. The relevance of an item *s* for user *i* is estimated based on the ratings assigned to item *s* by those users *C* who are "similar" to user *i*. In a pure CF approach, data is represented by a two-dimensional (2D) User x Item matrix as in Figure 1(a). In a multi-dimensional (MD) approach the data can be described by additional *k* dimensions and be represented by *k* User x Item matrices. The additional dimensions take into account additional information, such as time, location, or, more in general, the context in which customer is. Figure 1(b) represents an MD structure where an additional dimension, context, is used beyond Users and Items. Adomavicius and Tuzhilin[18] provide a broad overview of the MD RS. Price may be included in such approach if it is used as an additional dimension.

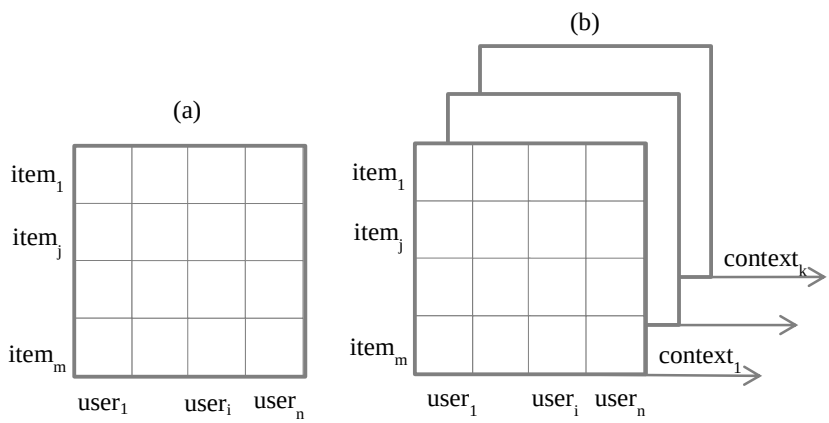

Figure 1. 2D structure (a) vs. MD structure (b)

Most of the research in this area has demonstrated that including additional contextual dimensions can improve recommendations accuracy if properly deployed[19]. Several alternative context-aware methods were compared by previous works[20] and it was demonstrated that context can increase both recommendation accuracy and diversity[21], thus also improving customers' trust[3]. Including price in an MD approach has not been studied before. The main differences in using price in an MD approach are the following:

1. Scale. Price is a numerical variable while MD approach works with a finite number of User x Item matrices. Therefore, a proper price discretization method is needed.

2. Items. Typical eCommerce companies sell different product categories and each category may have totally different price scales. Therefore, each item requires a different price discretization scale.

3. Price sensitivity. Price discretization has to represent a good estimation of customer price sensitivity. When price is discretized, the price categories have to represent the customer sensitivity to price for that item.

These issues can be explained by an example. If two items are recommended to a customer, and the two items have all the features equal except the price, the customer will naturally prefer the item with a price closer to the optimal value, or the price corresponding to the customer's maximum preference.

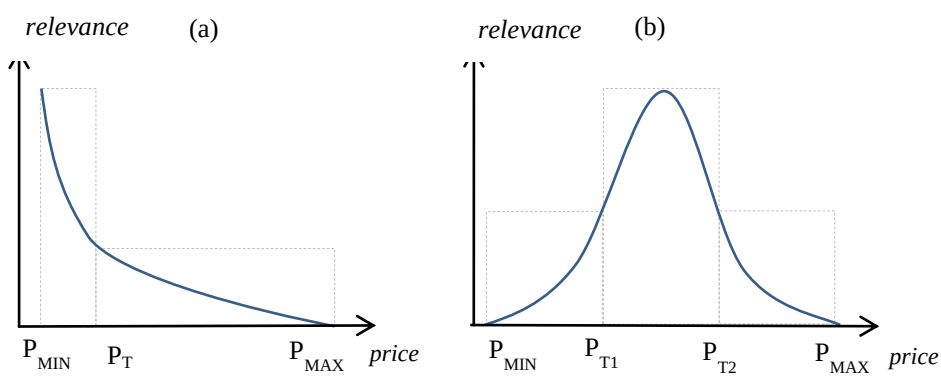

Figure 2. Customer (a) monotonic price sensitivity by two non-equal price categories and (b) Gaussian price sensitivity by three equal price categories

It is important to notice that price sensitivity varies across customers and across products. Different customers may have different sensitivity to price for the same product category and the same customer may have different price sensitivity for different products. As an example, consider an eCommerce application that sells "men's shoes" as one of its product categories (items) and suppose the price range for that category is $[P_{MIN};P_{MAX}]$ = [20 €;300 €]. Certain customers may prefer to spend very little for this kind of shoe and will look for products in the low end of the range. Others may distrust shoes at too low a price and may prefer to buy products with a price in the middle of the range. In addition, a customer who looks for shoes in the low end of the price range may show a completely different behavior when looking for a different product category. For instance, when looking for an "audio cable," he/she may want to buy a very expensive product. These situations are depicted in Figure 2(a) and (b). Figure 2(a) depicts the price sensitivity of the first kind: if price is low, then relevance is high and the curve of the relevance is monotonic and decreasing. Figure 2(b) depicts the price sensitivity of the second kind: the curve of the relevance is similar to a Gaussian, because products with a price in the middle of the range have the highest relevance.

The way price is discretized has to model price sensitivity. For instance, in Figure 2(a) price is discretized by setting a threshold P_T, which identifies two price categories: (i) low price and (ii) high price. When using an MD approach, the User x Item matrix containing customers' rating is then split into two matrices, namely (i) User x Item x Low-price and (ii) User x Item x High-price. In Figure 2(b), price is discretized by setting two thresholds P_{T1} and P_{T2}, which identify three price categories: (i) low price, (ii) mid price and (iii) high price. When using an MD approach, the User x Item matrix containing customers' ratings is then split into three matrices, each one corresponding to a price category.

In general, the main problems in discretizing price are the following.

1. Setting the right number of price categories.
2. Setting the price categories relative magnitude.

The first problem described above (setting the number of price categories) is not particularly new in the context of MD approaches to RS design, although price was never used as additional dimension. Several studies have demonstrated that increasing the number of values that context can take entails obtaining more homogenous data in each User x Item x Context matrix. This is expected to increase recommendation accuracy. However, the sparsity of data in the matrices increases and this is expected

to decrease accuracy. Research has demonstrated that in many cases accuracy prevails and MD approaches outperform 2D in terms of accuracy. Because we believe that the trade-off between homogeneity and sparsity is analogous to that between-price estimation and sparsity, we did not consider this problem and focused on the second as the research issue.

According to RS literature, including price in an RS may appear as an obvious way to improve accuracy on average. However, even if the average accuracy may increase when including price, the accuracy in certain price categories may decrease. For instance, accuracy may increase when recommending low-priced items while decreasing when recommending high-priced items, or vice versa. This directly affects the business performance of a company. In fact, high accuracy when recommending cheap products may not balance low accuracy when recommending expensive products.

Because accuracy depends on the amount of ratings available, the choice of the magnitude of price categories has an immediate effect on recommendation accuracy. Given a number of price categories, the values of the thresholds affect the relative sparsity of the User x Item x Price matrices. For instance, setting the value of P_T in Figure 2(a) means that a certain number of ratings will be included in User x Item x Low-price and another number of ratings will be included in User x Item x High-price. If the number of ratings in the first matrix is higher, then accuracy will be probably higher for low-priced items. Varying the value of P_T means obtaining different recommendation accuracy values for one price category or the other.

Since more customers have a preference for low prices, it is reasonable to expect many ratings for low-priced items and few ratings for high-priced items. Therefore, if price thresholds are not set properly, the RS will be very accurate when recommending low-priced items, while very inaccurate when recommending high-priced items. This would represent an obvious drawback for an e-commerce company because low priced items are typically less profitable than high-priced items. Therefore, the business performance may be relatively low if price is discretized in the wrong way. However, if one decreases the magnitude of price categories in the low end of price range and increases the magnitude of those in the high end, the recommender will increase its accuracy when recommending high-priced products, but the accuracy when recommending low-priced items might become too poor.

The key point is to compare the accuracy performance of a recommendation engine which includes price (we will call it Price-Sensitive

Recommender System, or PSRS) and that of a regular recommender engine (without price). We believe that it is not sufficient to compare the average accuracy. Rather, we need to compare accuracy in each price category because the choice between the two alternatives (RS and PSRS) depends on the whole business performance. If a PSRS is able to increase accuracy in all of the price categories, then it is considered better than a regular RS (without price). If using a PSRS increases accuracy only in some price categories and decreases accuracy in other price categories, then the company may prefer not to risk a decrease in profitability and then will choose a regular RS.

4. EXPERIMENTAL SETTINGS

The proposed PSRS is based on the MD recommender system's design proposed by Adomavicius et al.[19] and on the data pre-filtering approach[18, 22]. This approach uses the contextual variables to filter out those ratings that do not correspond to the specified context. This filtering is applied before the main recommendation method is launched on the data that passed the filter. In other words, if a particular dimension of interest is *K*, then the pre-filtering method selects from the initial set of all the ratings only those corresponding to the specified dimension *K*. As a result, it generates a User x Item matrix containing only the data pertaining to *K*. Then the RS method of choice (e.g., collaborative filtering) is launched on this remaining dataset that passed the filter to generate recommendations for K.

We experimented with two different CF algorithms in order to generalize, as much as possible, our results: the user-based and the item-based[23, 24]. Both algorithms use Pearson Correlation similarity and neighborhood formation. The main difference between these two algorithms is that the user-based algorithm generates recommendations by measuring similarity among users, while the item-based algorithm generates recommendations by measuring similarity among items.

We performed our experiments using a dataset coming from the study described in Palmisano et al.[25]. It was asked to a group of students to navigate and simulate purchases on Amazon.com during a period of four months. While navigation was real on Amazon.com, purchasing was simulated. Once a product was selected by a student to be purchased, the browser recorded all the useful characteristics of the transaction (i.e., whether the user clicked on the "I like it" button associated with the item) and this information was stored in the database. Further, the data was pre-processed and the resulting number of students was 556, and the total

number of purchasing transactions for the students was 31,925. We split the database in 2/3 used as training set and 1/3 used as validation set.

We used these simulated purchasing transactions as implicit ratings[26]. We preferred implicit ratings and not to use explicit ratings because this is a better simulation of the real industrial settings we are interested to. In commercial applications (such as Amazon.com), the number of implicit ratings is higher than explicit ratings. In particular, we used a 1-5 scale. If the user clicked on an item without purchasing it, a rating equal to 1 was associated with the transaction. If the user liked the item without purchasing it, a rating equal to 3 was associated with the transaction, while if the user liked the item and he/she purchased it, a rating equal to 5 was associated with the transaction.

The dataset includes 21 product categories, ranging from books to jewelry. The price scales of different items is very different. For instance, the price of books ranges in the following interval: $[P_{MIN};P_{MAX}] = [1€;100€]$. The price of jewels ranges in the following interval: $[P_{MIN};P_{MAX}] = [1€;49,505€]$.

According to Bergemann and Ozmen[16], we assume that items' price is defined as "a priori" using other mechanisms different from the recommendation engine (i.e., traditional marketing approaches) and therefore we just defined price categories and their magnitude. We decided to set two price categories: "low-price" and "high-price". We varied the relative magnitude of the price categories in order to investigate whether it has a direct impact on the performance of the proposed PSRS. Therefore, we varied the threshold value P_T used to distinguish between the two categories. We defined two criteria to set the price threshold and compared them to each other. First, we set P_T in order to balance the amount of information contained in each matrix. Second, we set P_T as the mean price in each product category. In the first criterion, for each product category, we define the value $P_{T,balance}$ that makes the sparsity of each User x Item matrix equal. As an example, suppose that 20 items were rated by customers in the "books" category at different prices in the [1€;100€] interval. If 10 of those items belong to [1€;9€] and 10 items belong to [11€;100€], then the price threshold is $P_{T,balance} = 10€$. The items with a price smaller than 10€ are then labeled as "low-price" items, while those with a price greater than 10€ are labeled as "high-price" items. Following this criterion, both the User x Item x Low-price and the User x Item x High-price matrices will include 10 data points each. Table 1 shows the example. In the second criterion, we set P_T as the mean price of each category of items. We called this threshold value $P_{T,avg}$. We set this value independently of the distribution of ratings in the price range. In the example above, we set $P_{T,avg} = 24.41€$ being the value of

the average of the rated items' price, as Table 1 shows. We labeled "low-price" the items with a price smaller than 24.41€, and "high-price" the items with a price greater than the threshold. As Table 1 shows, the choice of the price threshold clearly affects the sparsity of the two matrices. The criterion of $P_{T,avg}$ makes the User x Item x High-price matrix sparser than the other matrix.

Table 1. An example of the $P_{T,avg}$ and $P_{T,balance}$ criteria

Price of rated items	$P_{T,balance}$ = 10.00€	$P_{T,avg}$ = 24.41€
€ 1.90	Low	Low
€ 1.20	Low	Low
€ 3.70	Low	Low
€ 4.90	Low	Low
€ 5.60	Low	Low
€ 6.80	Low	Low
€ 7.40	Low	Low
€ 8.00	Low	Low
€ 9.30	Low	Low
€ 9.90	Low	Low
€ 11.40	High	Low
€ 11.50	High	Low
€ 15.20	High	Low
€ 17.00	High	Low
€ 18.70	High	Low
€ 42.30	High	High
€ 58.20	High	High
€ 80.90	High	High
€ 84.30	High	High
€ 90.00	High	High

We measured the recommendations' accuracy using a metric traditionally used in IR, such as F-measure[27]. We also measured the F-measure distinguishing between the two price categories in order to investigate whether the PSRS is effective in recommending both type of items (namely, high-priced and low-priced).

5. RESULTS

As discussed in previous sections, we compared our PSRS with a 2D collaborative filtering (CF) version. In particular, we compared a user-based PSRS with a user-based CF and an item-based PSRS with an item-based CF.

For the sake of conciseness, we depict the "accuracy gain" computed as the difference between the accuracy (F-measure) of a PSRS and the corresponding 2D CF engine. Therefore, a positive value of accuracy means that the PSRS outperforms the 2D CF, while a negative value means the reverse.

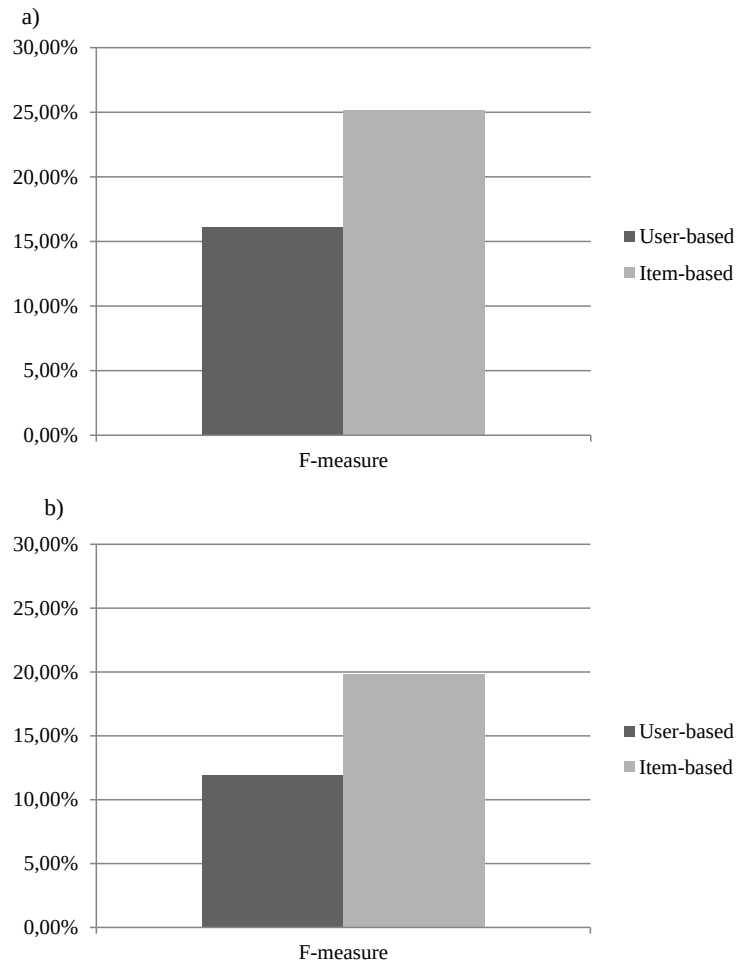

Figure 3. F-measure average gains across price categories when using (a) $P_{T,balance}$ and (b) $P_{T,avg}$

Figure 3 shows the F-measure gains across price categories when comparing the user-based PSRS and the item-based PSRS with the corresponding 2D CF approaches. In particular, Figure 3 shows the results when using $P_{T,balance}$ (Figure 3(a)) and $P_{T,avg}$ (Figure 3(b)). The PSRS outperforms the CF for both the user-based and the item-based engines in both cases. The accuracy gain is substantial and the result is in line with

previous literature that demonstrated that an MD recommender system can outperform a 2D approach. In general, the result shown by Figure 3 means that including price in a recommender engine can increase the average accuracy of recommendations. This does not mean, however, that accuracy increases in all of the price categories. Figure 4 clearly shows this point.

Figure 4 reports the F-measure gain for the two price categories, low-price and high-price, when using $P_{T,balance}$ (Figure 4(a)) and $P_{T,avg}$ (Figure 4(b)).

The first result coming from Figure 4 is that the ability of a PSRS to outperform a 2D CF approach depends on the way price thresholds are set. The second important finding is that if the price threshold is properly set, then the PSRS outperforms the 2D CF for both price categories. If the $P_{T,balance}$ criterion is used to set price categories (Figure 4(a)), the PSRS proves to be more accurate than a 2D CF approach both when low-priced items are recommended and when high-priced items are recommended. If the $P_{T,avg}$ criterion is used to set price categories (Figure 4(b)), the PSRS outperforms the 2D CF engine only when low-priced items are recommended.

This makes an important difference for a company because products at different prices are characterized by different profitability. Typically, high-priced products are more profitable than low-priced products. Being able to increase the accuracy of recommendations for both price categories may be crucial to improving the business performance of an online company.

In conclusion, we demonstrated that including item price in a recommender system is possible through a multi-dimensional design. However, including price in a multidimensional recommender system is tricky because the definition of the price categories' magnitude has a direct impact on business performance. Our findings show that it is possible to find a proper way to discretize price and including it in a MD approach. We propose to balance the amount of information in each User x Item x Price matrix as criterion. If this criterion is followed, then the PSRS outperforms a 2D CF recommender in all the price categories. This would allow a company to increase its overall business performance. If the wrong criterion should be used, the business performance might decrease as a whole.

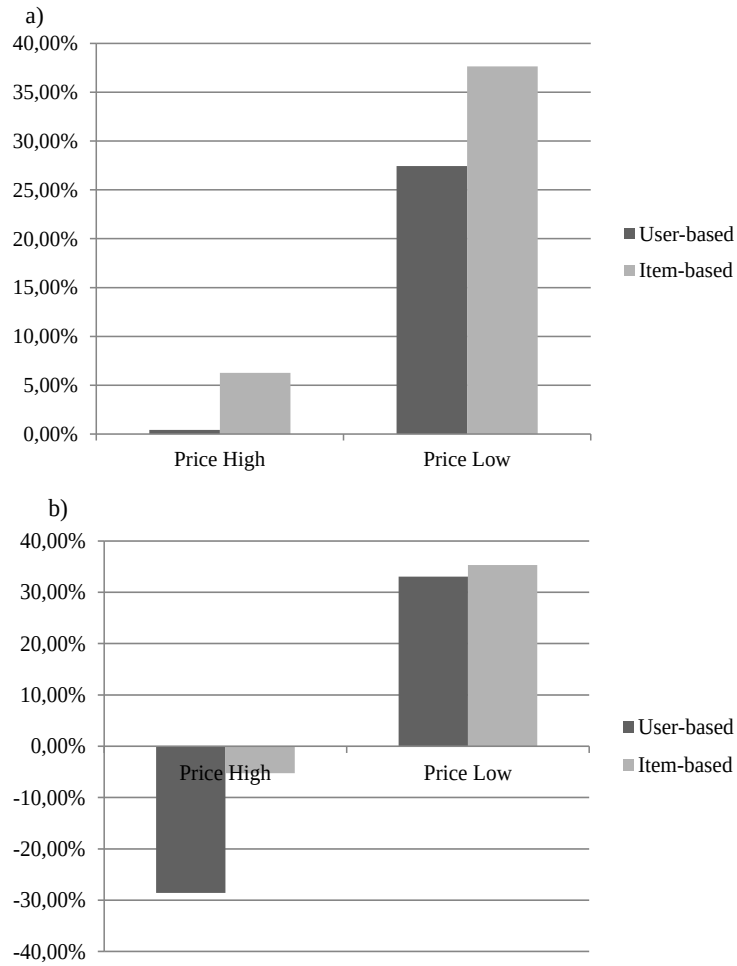

Figure 4. F-measure gains in each price category when using (a) $P_{T,balance}$ and (b) $P_{T,avg}$.

6. CONCLUSIONS

This research represents a first step to designing price-sensitive recommender systems. At this initial research stage, we want to describe the main issues of designing this type of recommendation engine and to demonstrate what the effect of this design is on recommendations accuracy and business performance.

We experimented with a Price Sensitive Recommender System (PSRS) designed as an MD approach based on a CF. We measured the accuracy gain with respect to a plain 2D CF recommender engine. We used both a

User-based and an Item-based engine, and compared the PSRS to the 2D CF engine when using two different criteria to set two price categories, namely low-price and high-price. In the first criterion, $P_{T,balance}$, we balance the sparsity of the User x Item x Price matrices. In the second criterion, $P_{T,avg}$, we simply split the whole price range of the item category into two equal intervals.

Our findings demonstrate that the way price is set affects business performance. Including price in the recommendation engine increases accuracy on average. However, using the $P_{T,avg}$ criterion to set price categories makes the accuracy of low-priced recommendations increase while the accuracy of high-priced recommendations decreases. This is critical from a business viewpoint. If a PSRS increases accuracy when recommending cheap products and decreases accuracy when recommending expensive products, then the firm may risk reducing its profitability, thus making the use of price in the recommendation engine counter-productive. Our research also demonstrates that it is possible to set price in such a way that the business performance increases as a whole. In fact, when using the $P_{T,balance}$ criterion, the accuracy of high-priced recommendations increases as well as that of low-priced recommendations.

Several steps have to be done in the direction we envisioned in this paper. The first of these future steps is to compare the PSRS to other approaches to include price in an RS, namely utility-based and content-based approaches and discussing pros and cons of these approaches. Another important step is to compare a PSRS with well-established marketing models to build customer price sensitivity and exploit this knowledge to include price in a recommendation engine in even more effective ways. Finally, a crucial further research step is to realize a live experiment in order to measure the actual reaction of customers when delivering price-sensitive recommendations. Additional performance metrics should be related to customer behavior, such as actual purchases, intentions of purchases, and trust.

7. REFERENCES

[1] A. Pham, and J. Healey, Tell you what you like. *Los Angeles Times*, September 20, 2005.

[2] B. Schafer, J. Konstan, and J. Riedl, E-commerce recommendation applications. *Data Mining and Knowledge Discovery*, 5(1), p115-153, 2001. http://dx.doi.org/10.1007/978-1-4615-1627-9_6.

[3] M. Gorgoglione, U. Panniello, and A. Tuzhilin, The effect of context-aware recommendations on customer purchasing behavior and

trust. *Paper presented at the fifth ACM Conference on Recommender Systems*, Chicago, October 23-27, 2011. http://dx.doi.org/10.1145/2043932.2043951.

[4] K. Hosanagar, K. Ramayya, and L. Ma, Recommended for you: The impact of profit incentives on the relevance of online recommendations. *Paper presented at the International Conference on Information Systems*, Paris, France, December 14-17, 2008. http://aisel.aisnet.org/icis2008/31.

[5] T. Kamishima, and S. Akaho, Personalized pricing recommender system: multi-stage epsilon-greedy approach. *Paper presented at the 2nd International Workshop on Information Heterogeneity and Fusion in Recommender Systems*, Chicago, IL, USA, October 23-27, 2011. http://dx.doi.org/10.1145/2039320.2039329.

[6] J.B. Schafer, J. Konstan, and J. Riedi, Recommender systems in e-commerce. *Paper presented at the 1st ACM conference on Electronic Commerce*, Denver, CO, USA, November 3-5, 1999. http://dx.doi.org/10.1145/336992.337035.

[7] G. Adomavicius, and A. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, 17(6), p734-749, 2005. http://dx.doi.org/10.1109/TKDE.2005.99.

[8] F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor, *Recommender systems handobook*. USA: Springer, 2011. http://dx.doi.org/10.1007/978-0-387-85820-3.

[9] A. Cooke, H. Sujan, M. Sujan, and B. Weitz, Marketing the unfamiliar: The role of context and item-specific information in electronic agent recommendations. *Journal of Marketing Research*, 39(4), p488-497, 2002. http://dx.doi.org/10.1509/jmkr.39.4.488.19121.

[10] T.-P. Liang, H.-J. Lai, and Y.-C. Ku, Personalized content recommendation and user satisfaction: Theoretical synthesis and empirical findings. *Journal of Management Information Systems*, 23(3), p45-70, 2007. http://dx.doi.org/10.2753/MIS0742-1222230303.

[11] P. Pu, L. Chen, and R. Hu, A user-centric evaluation framework for recommender systems. *Paper presented at the fifth ACM Conference on Recommender Systems*, Chicago, October 23-27, 2011. http://dx.doi.org/10.1145/2043932.2043962.

[12] P. Bharati, and A. Chaudhury, An empirical investigation of decision-making satisfaction in web-based decision support systems. *Decision Support System*, 37(2), p187-197, 2004. http://dx.doi.org/10.1016/S0167-9236(03)00006-X.

[13] D. Fleder, and K. Hosanagar, Blockbuster culture's next rise or fall: The impact of recommender systems on sales diversity. *Management Science*, 55(5), p697-712, 2009.

http://dx.doi.org/10.1287/mnsc.1080.0974.

[14] B. Pathak, R. Garfinkel, R. Gopal, R. Venkatesan, and F. Yin, Empirical analysis of the impact of recommender systems on sales. *Journal of Management Information System*, 27(2), p159-188, 2010. http://dx.doi.org/10.2753/MIS0742-1222270205.

[15] R. Burke, Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4), p331-370, 2002. http://dx.doi.org/10.1023/A:1021240730564.

[16] D. Bergemann, and D. Ozmen, Optimal pricing with recommender systems. In J. Feigenbaum, J. Chuang, and D. Pennock (Eds.), Pr*oceedings of the 7th ACM Conference on Electronic Commerce* (p43-51). Ann Arbor, Michigan, USA: ACM Press, 2006. http://dx.doi.org/10.1145/1134707.1134713.

[17] P. Lops, M. Gemmis, and G. Semeraro, Content-based recommender systems: State of the art and trends. In F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor (Eds.), *Recommender systems handbook* (p73-105). USA: Springer, 2011. http://dx.doi.org/10.1007/978-0-387-85820-3_3.

[18] G. Adomavicius, and A. Tuzhilin, Context-aware recommender systems. In F. Ricci, L. Rokach, B. Shapira, and P.B. Kantor (Eds.), *Recommender systems handbook* (p217-253). USA: Springer, 2011. http://dx.doi.org/10.1007/978-0-387-85820-3_7.

[19] G. Adomavicius, R. Sankaranarayanan, S. Sen, and A. Tuzhilin, Incorporating contextual information in recommender systems using a multidimensional approach. *ACM Transactions on Information Systems*, 23(1), p103-145, 2005. http://dx.doi.org/10.1145/1055709.1055714.

[20] U. Panniello, A. Tuzhilin, M. Gorgoglione, C. Palmisano, and A. Pedone, Experimental comparison of pre- vs. post-filtering approaches in context-aware recommender systems. *Paper presented at the third ACM Conference on Recommender Systems*, New York City, NY, USA, October 22-25, 2009. http://dx.doi.org/10.1145/1639714.1639764.

[21] U. Panniello, A. Tuzhilin, and M. Gorgoglione, Comparing context-aware recommender systems in terms of accuracy and diversity. *User Modeling and User-Adapted Interaction*, 24(1-2), p35-65, 2014. http://dx.doi.org/10.1007/s11257-012-9135-y.

[22] U. Panniello, M. Gorgoglione, and C. Palmisano, Comparing Pre-filtering and Post-filtering Approach in a Collaborative Contextual Recommender System: An Application to E-Commerce. *E-Commerce and Web Technologies, Lecture Notes in Computer Science*, 5692, p 348-359, 2009. http://dx.doi.org/10.1007/978-3-642-03964-5_32.

[23] J.L. Herlocker, J.A. Konstan, A. Borchers, and J. Riedl, An algorithmic framework for performing collaborative filtering. *Paper presented at*

the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, Berkeley, August 15-19, 1999. http://dx.doi.org/10.1145/312624.312682.

[24] B. Sarwar, G. Karypis, J. Konstan, and J. Riedl, Item-based collaborative filtering recommendation algorithms. *Paper presented at the 10th International Conference on World Wide Web*, Hong Kong, China, May 1-5, 2001. http://dx.doi.org/10.1145/371920.372071.

[25] C. Palmisano, A. Tuzhilin, and M. Gorgoglione, Using context to improve predictive modeling of customers in personalization applications. *IEEE Transactions on Knowledge and Data Engineering*, 20(11), p1535-1549, 2008. http://dx.doi.org/10.1109/TKDE.2008.110.

[26] D.M. Nichols, Implicit rating and filtering. *Paper presented at the Fifth DELOS Workshop on Filtering and Collaborative Filtering*, Budapest, November 10-12, 1997.

[27] J.L. Herlocker, J.A. Konstan, L.G. Terveen, and J.T. Riedl, Evaluating collaborative filtering recommender systems. *ACM Transactions on Information Systems*, 22(1), p5-53, 2004. http://dx.doi.org/10.1145/963770.963772.