Attribute Aware Anonymous Recommender Systems

Manuel Stritt, Karen H.L. Tso, Lars Schmidt-Thieme

Computer-based New Media Group, Department of Computer Science
Albert-Ludwigs-Universität Freiburg, Germany
{stritt,tso,lst}@informatik.uni-freiburg.de

Summary. Anonymous recommender systems are the electronic pendant to vendors, who ask the customers a few questions and subsequently recommend products based on the answers. In this article we will propose attribute aware classifier-based approaches for such a system and compare it to classifier-based approaches that only make use of the product IDs and to an existing knowledge-based system. We will show that the attribute-based model is very robust against noise and provides good results in a learning over time experiment.

1 Introduction

Recommender systems (RS) are used by online commercial sites, e.g. amazon.com and ebay.com (giftfinder), to help users to find products that fit their preferences. They use user profiles that contain preference indicators for specific products or types of products to assess the interest a customer may have in product offerings and recommend those he is assumed to like most. Preferences can either be specified directly by customers, i.e., by rating products, or indirectly indicated by their behavior, e.g., search keywords, products viewed in detail, products put in the market basket or products purchased as well as frequencies and durations of such events.

In general, there are two types of RS [10]:
(i) **RS with user identification** that require a user to login (or rely on other unsafe user identification mechanisms such as cookies) and therefore can accumulate preference indicators, e.g., purchase histories over time. Whenever a customer logs in the system, his past profile is looked up and used for recommending. New users have to provide some front-up information, e.g., some initial product ratings, before they can use the system. This is sometimes called the **new user problem** [9].
(ii) **Anonymous recommender systems** that do not require a user identification and therefore do not have any initial information about a customer. As in the context of e-commerce direct ratings and usage-based preference indicators are too time consuming to collect for single use, other mechanisms have to

be used to elicit preference information. Usually, a short questionnaire that is customized to the product category and asks for customer needs is presented up-front. This questionnaire sometimes is called **task specification**.

While RS with identification often can collect referenced indicators that are closely related with specific products, product types or attributes, the main problem of anonymous recommender systems is that the task specification has to be sufficient broad and generic to be filled-in easily by customers, but then has to be related to specific products to be useful for recommendations.

In most commercial systems such as the Mentasys Sales Assistant[1], this relation between customer needs and products is modeled initially explicitly by means of a complex conceptual model that is adapted later-on by usage information. To create such a conceptual model requires a domain expert and methods for eliciting his knowledge etc. and bears many problems in its own, not the least, that it is time consuming, expensive, and has to be done for each product category.

Alternatively, one could try to learn such preference models automatically with machine learning methods as reported in [10]. These models are based only on the product IDs. In this article we will introduce classifier based models that take the attributes of the products in account, too. For this, we will make the following contributions: (i) we will propose a classification model setup for learning anonymous recommender systems and provide evaluation methods (section 3), (ii) we will introduce a classifier-based model that makes use of the product attributes (section 4) and (iii) we will provide additional empirical evidence for system behavior over time (section 5).

2 Related Work

There are in general four recommendation approaches: collaborative filtering, content-based, hybrid and knowledge-based [7].

Collaborative Filtering [6], the most commonly-used technique is the attend to find users with the same preferences and to recommend objects to a user that other users with the same preference liked. It uses the simple nearest neighbor methods and does not make use of object attributes. This method has been quite successful in terms of recommendation quality. Hence, due to their simplicity and good quality, collaborative filtering is the prevalent method in practice.

Content-based Filtering (CBF) stems from Information Retrieval (IR), computes comparison of rated items of a single user and the item in the repository. Item attributes information is used for this technique. The performance of using solely CBF have shown to be rather poor. Yet, attributes usually contain valuable information that could improve the performance of recommender systems.

[1] see http://www.mentasys.de.

Hybrid collaborative/content-based Filtering combines both CF and CBF techniques. Some hybrid recommender systems are described in [1], [8], [5] and [12].

Knowledge-based uses knowledge from both the users and the products/items to generate recommendations.

The first three techniques are mostly suited for persistence recommendation, whereas the knowledge-based technique is commonly used for task-based ephemeral recommendation.

The prediction task for various recommendation approaches can be handled using different methods. Commonly used methods are neighborhood formation, association rule mining, machine learning techniques, ...etc. In general, it can be done in two different ways in the research literature: (i) using heuristic correlation measures and (ii) using learning methods to train a classification model that predicts further ratings or rated items. In most cases, classification models have shown to be suitable for prediction tasks when used with products or users attributes information ([2] and [3]). Thus, using classification models would be an appropriate approach for handling knowledge-based RS. In [10] we introduced classifier-based models for anonymous RS that only take the product IDs into account. In contrast to this paper we will now describe models that make use of the product attributes, too. Attribute-based models have already been used to be useful on data with varying characteristics as described in [11].

3 Framework

The anonymous RS framework makes use of the following entities:

- Set of answers A = $\{a_1, a_2, ..., a_l\}$
- Set of products I = $\{i_1, i_2, ..., i_n\}$
- Product ranklists $R \in I^n$
- Set of profiles $U \in (P(A))^m$
- Sessions $S \in P(A)\text{x}I$

With this framework an anonymous RS can be modeled as a classification task [10]. The learning table can be gathered from successful[2] (U,I) instances from an existing system or might be generated from a domain expert. In our scenario we have made use of the data gathered from an existing system provided by MENTASYS GmbH that we will call the status-quo system. The models proposed in this article are trained and evaluated on about 20000 instances based on data provided from the status-quo system.

[2] The success-criterion should represent that a user likes a recommended product. In our scenario we define a success as product view (e.g. the user clicked on the recommended product).

Evaluation

For an in-vitro evaluation, the list of ranked products proposed by the classifier-based model are evaluated on the data of the status-quo system [10]. We define that a hit (i.e. a user likes the proposed product) succeeds if an entry (u,i) appears in the test data. A hit can be seen as an entry in only one session or in all sessions containing the same profile.

An ideal case would be to propose a viewed product on the first position of the ranklist than listing it at the bottom of the ranklist (which is again better than not proposing it at all). Therefore, the rank positions are assigned with different weights. Breese et al. proposed the so-called breese-score [4] that weights the rank positions in exponential decay. For evaluating different anonymous RS, we use a score that is based on this breese-score but also takes the session s and profile u in account:

$$RankScore(s, u, r) = \frac{hitcount(s, u, r)}{2^{(r-1)/(\alpha-1)}}$$

We set the parameter α to 5 as in [4]. The function *hitcount* is dependent of the the session s, the profile u and the rank r and can be calculated in different ways, that lead to different measures:

- **multicount (MC)**: number of product impressions in $TSess$ of product listed at Ranklist(r)
- **singlecount (SC)**: 1 if $MC > 0$, else 0

Where $TSess$:

- **local**: $TSess = s$
- **global**: $TSess \subseteq S; \forall s_i \in TSess : \exists(u_j, i_j)|u_j = u$

The rankscore for the complete ranklist R is given by the sum over all ranks:

$$RankScore(s, u, R) = \sum_{i=1}^{|R|} RankScore(s, u, r_i)$$

To get an expressive comparable score, the scores can be calculated as percentage to the maximal rankscore obtained from the optimal ranklist.

$$RankScore = \frac{RankScore_{list}}{RankScore_{max}} * 100$$

The optimal ranklist consists of the products in *TSess* ordered descendingly by the frequencies. In this article we confine to the global, multicount scores as this one is the most expressive score for customer preferences [10].

4 Attribute-Based Models

In [10] we introduced ID-based models for anonymous RS. These classifier-based models have been seen as a U → I training that leads to a P → R assignment. Another possibility is to make use of the product attributes (e.g. price, weight, height) in the hope that they provide more information than just the product-IDs. With this strategy, the model should be able to learn preferences in sense of what attributes customers like, what is closer to the recommendation task of a human salesman.

The idea is to use one classifier for each attribute that propagates one attribute given a profile. This step is followed by a second step that assigns the set of attributes a product-ID distribution. This second step can be seen as a classification problem, too. Figure 1 shows the work flow of this strategy.

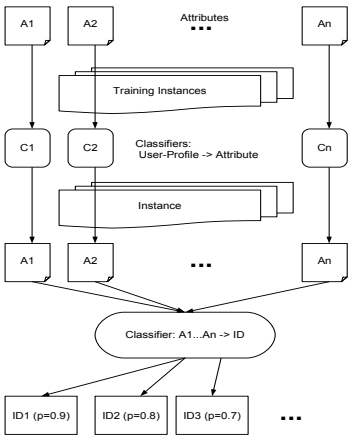

Fig. 1. Attribute-based model.

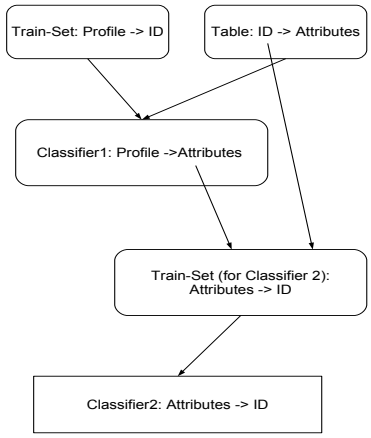

Fig. 2. Classifier: attributes → product-ID.

The training for the classifier of the second step is based on the products of the training table for the first step. The attributes proposed from the first step then are combined with the (original) product-IDs what leads to the training table for the second classifier that is able to predict a product-ID given a set of attributes. Figure 2 shows this strategy.

One experiment was made with only one attribute, the name of the picture of the product, which is a unique identifier for the product (Attr(bilds)). Further experiments with 5 expressive attributes (attr(5)) and 15 attributes (attr(15)) of the products are made as well as experiments with nearly all attributes (attr(80)) and with nearly all attributes but unique identifier attributes (attr(80 -bilds)). Figure 3 shows the results for global multicount

scores in comparison to the status-quo system and the ID-based model using a NB-Tree classifier.

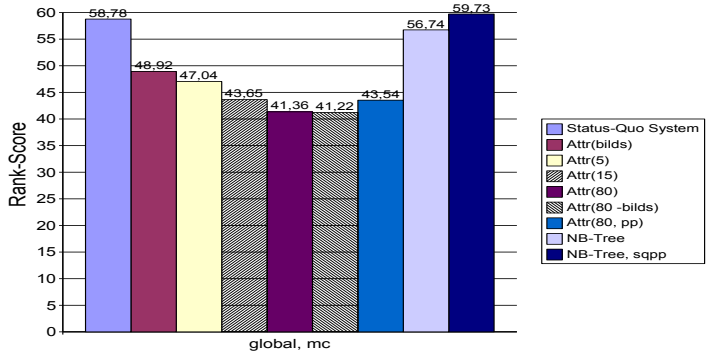

Fig. 3. Results of attribute-based models.

Unfortunately, the attribute-based results are not competitive compared to the ID-based system using a NB-Tree classifier. We think that the bad performance of such attribute-based models is based on the kind of evaluation method that takes only the product-IDs in account and does not care about the attributes. Thus, the model would only get a good score if the model proposes exactly the products the status-quo system proposed, even if the attribute-based model proposes products that, regarding the attributes, fits more the desires of the customers. Anyway, in learning over time (see 5) the results are competitive to the ID-based system and seem to be more robust in sense of a lower variance.

5 Learning over time

In all experiments so far, all sessions have been randomized before splitting them into training and testing datasets. In real life, only data from the past can be taken into account for training because no data from the future is available. To simulate such a scenario, the whole data is divided into 10 segments ordered ascending by time. Each time segment contains data of about two weeks and is evaluated using only the time segments in the past for training. This means segment 1 is evaluated using segment 0 as training data and segment 2 is evaluated using segment 0 and 1 as training data and so on.

Figure 4 shows the results from learning over time by comparing the attribute-based model to the ID-based model using a NB-Tree classifier. This shows that the attribute-based model is competitive to the ID-based model in learning over time. The advantage of the attribute-based model is the lower

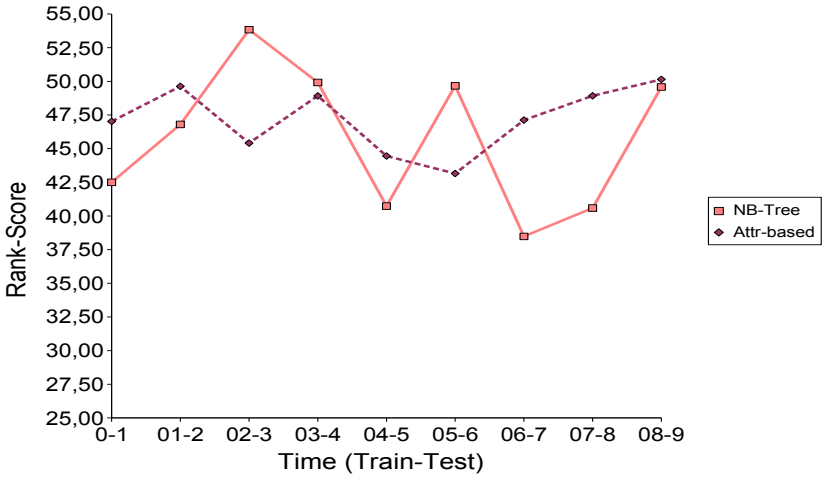

Fig. 4. Learning over time for an attribute-based model in comparison to an ID-based model using a NB-Tree classifier. (Scores from global, multicount.) Mean (attr/id): 47,19 / 45,79. Variance (attr/id): 5,95 / 28,62. (XY-Z on the x-axis means that segment Z was evaluated using segments X to Y as training data.)

variance. This is an evidence that the attribute-based system is the more robust one that is able to compensate noisy data.

6 Conclusion

Anonymous recommender systems are able to help users to find the products of their needs. The assignment from latent variables (the user preferences) to products is normally designed by experts knowledge. These knowledge-based systems are very successful but it takes a lot of effort to build these systems and to foster them. In this article, we proposed classifier-based models that make use of the product attributes.

In our first experiment the attribute-based models were not competitive to the classifier-based system (section 4). One reason for this could be the evaluation method that is based on "hard" IDs that don't have to stand for the real user preferences. A better way would be to define a metric based on the attributes and then compare the user preferences to the recommendations based on this metric (and not just hit or no hit).

Anyway, in learning over time the attribute-based model gives a good performance and this with a lower variance. The learning over time experiment is very important because it only takes data from the past into account what is closer to a real live scenario. In this scenario the product set of the training

and test data differs more that in a scenario where training and test datasets are taken out of a randomized data.

Overall, attribute-based models seem to by a very robust method for recommender systems and provide advantages when the data is noisy. Further work has to be done to evaluate these models in a real live experiment to counterpoise the disadvantages of the in-vitro evaluation.

References

1. Marko Balabanović and Yoav Shoham. Fab: content-based, collaborative recommendation. *Communications of the ACM*, 40(3):66–72, 1997.
2. Justin Basilico and Thomas Hofmann. Unifying collaborative and content-based filtering. In *ICML '04: Proceedings of the twenty-first international conference on Machine learning*, page 9, New York, NY, USA, 2004. ACM Press.
3. Chumki Basu, Haym Hirsh, and William Cohen. Recommendation as classification: using social and content-based information in recommendation. In *AAAI '98/IAAI '98: Proceedings of the fifteenth national/tenth conference on Artificial intelligence/Innovative applications of artificial intelligence*, pages 714–720, Menlo Park, CA, USA, 1998. American Association for Artificial Intelligence.
4. John S. Breese, David Heckerman, and Carl Kadie. Empirical analysis of predictive algorithms for collaborative filtering. In *14th Conference on Uncertainty in Artificial Intelligence(UAI-98)*, pages 43–52, 1998.
5. Robin Burke. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.
6. David Goldberg, David Nichols, Brian M. Oki, and Douglas Terry. Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, 35(12):61–70, 1992.
7. Zan Huang, Wingyan Chung, and Hsinchun Chen. A graph model for e-commerce recommender systems. *J. Am. Soc. Inf. Sci. Technol.*, 55(3):259–274, 2004.
8. P. Melville, R. Mooney, and R. Nagarajan. Content-boosted collaborative filtering. In *Eighteenth National Conference on Artificial Intelligence (AAAI-2002)*, pages 187–192, Edmonton, Canada, 2001.
9. A. Rashid, I. Albert, D. Coseley, S. Lam, S. McNee, J. Konstan, and J. Riedel. Getting to know you: Learning new user preferences in recommender systems, 2002.
10. Manuel Stritt, Karen Tso, Lars Schmidt-Thieme, and Dirk Schwarz. Anonymous recommender systems. *ÖGAI Journal*, pages 4–11, 2005.
11. Karen Tso and Lars Schmidt-Thieme. Evaluation of attribute-aware recommender system algorithms on data with varying characteristics. In *Proceedings of the tenth Pacific-Asia Conference on Knowledge Discovery and Data Mining (PAKDD 2006)*. Springer, 2006.
12. Cai-Nicolas Ziegler, Lars Schmidt-Thieme, and Georg Lausen. Exploiting semantic product descriptions for recommender systems. In *Proceedings of the 2nd ACM SIGIR Semantic Web and Information Retrieval Workshop*, Sheffield, UK, July 2004.