Anytime AHP Method for Preferences Elicitation in Stereotype-Based Recommender System

December 13, 2007

Abstract

In stereotype-based recommendation systems, user profiles are represented as an affinity vector of stereotypes. Upon the registration of new users, the system needs to assign the new users to existing stereotypes. The AHP (Analytic Hierarchy Process) methodology can be used for initial elicitation of user preferences. However, using the AHP procedure as-is will require the user to respond to a very long set of pairwise comparison questions. We suggest a novel method for converting AHP into an anytime approach. At each stage, the user may choose not to continue. However, the system is still able to provide some classification into a stereotype. The more answers the user provides, the more specific the classification becomes.

Keywords: recommendation systems, preferences elicitation, decision tree, Analytic Hierarchy Process

1 Introduction

Recommender Systems — systems that recommend items to users — can be found in many modern web sites for various applications such as helping users find web pages that interest them, recommending products to customers in e-commerce sites, recommending TV programs to interactive-TV users and showing personalized advertisements.

Different approaches for producing recommendations can be found in the literature (see [10] for a survey on recommendation approaches). Common to all methods is the utilization of some kind of a user profile or user model for recommendation. We propose to create a set of stereotype profiles and use an affinity vector of stereotypes as the user profile. These stereotypes are automatically created by an update process that the system undergoes.

In this paper, we briefly review the details of our recommendation system. Specifically, we describe how stereotypes are created and updated, and how recommendations are generated. We focus on the task of associating new users to stereotypes.

We propose to associate new users to stereotypes by using a questionnaire, that is generated automatically from the stereotypes after each update. Existing users do not need to undergo the questionnaire again and they are automatically classified to new stereotypes through the update process.

Our questionnaire is created as an interactive, easy to use process. At each stage two items are presented to the user who is asked to select the preferred item. The items are presented as pictures to ease the answering process. The questionnaire is an anytime algorithm, i.e. at each stage the user may choose not to continue. Nevertheless, the system is still able to offer some classification into a stereotype. The user's classification becomes more specific as the number of answers she provides increases.

2 Stereotype-Based Recommender System

With the explosion of online available data, recommender systems [16] become very popular especially in web sites. While there are many types of recommender systems ranging from manually predefined un-personalized recommendations to fully automatic general purpose recommendation engines, two dominating approaches have emerged - Collaborative Filtering (CF) and Content Based (CB) recommendations.

In many situations, people who are looking for recommendations ask for the advise of their friends. Collaborative filtering stems from this idea. While the population on the internet that can supply advise is very large, the recommendation problem shifts into identifying what part of this population is relevant for the current user.

The greatest advantage of CF is that it is independent of the specification of the item and can therefore provide recommendations for complex items which are very different yet are often used together. The major drawback of this approach is the inability to create good recommendations for new users that have not yet rated (many) items, and for new items that were not rated by (many) users. The last drawback is sometimes referred to as the *cold-start* problem

The idea of content-based recommendations originated from the field of information filtering, where documents are searched given some analysis of their text. Items are hence defined by a set of features or attributes. Such systems define a user by using preferences over this set of features. They obtain recommendations by looking for the best matches of user profiles and item profiles. Although some researchers [10] separate between methods that learn preferred attributes of rated items (called content-based) and methods that ask the user to specify her preferences over item attributes (called demographic filtering), we refer to all methods that recommend according to item attribute preferences as content-based recommendation.

CB systems can easily provide valid recommendations to new users, assuming that their profile is specified, even if they never used the system before. However, new user will get incorrect recommendations in case their profile is not specified. CB engines can provide recommendations for new items that were never rated before based on the item description and are therefore very useful in environments where new items are constantly added.

As we see above, the disadvantages of the CF and CB approaches can be amended by combining the two into a hybrid method [2]. Many hybrid approaches use two recommendation algorithms and combine their results in some manner, e.g. combining the results by their relevance, mixing the output of the two algorithms, switching from CB into CF once the cold-start phase is over, or using the output of one algorithm as the input to the second algorithm.

It seems that a more appropriate combination would be to create an algorithm that is by nature a hybrid of CF and CB, and not an ad-hoc combination of two independent algorithms. This kind of recommendation approach is adopted by the stereotype-based recommender systems proposed in this paper. Modeling users by stereotypes (or communities) is a well studied concept [17]. Stereotypes are a generalization of users — an abstract user entity that provides a general description for a set of similar users (a community).

In CF systems, stereotypes are described by a set of ratings over items. Similarity between users can be identified by their affinity to various stereotypes. In CB systems stereotypes are a set of preferences over item attributes, and users can belong to a single stereotype [17] or to several stereotypes [12]. Recommendations are computed based on the stereotype and then normalized given the user affinity to a stereotype.

We take the content-based approach here, and define an ontology over media items, using an expert in the field. It is reasonable to assume that an expert will be able to identify the key features relevant for people when they choose which movie to see. A media item profile is an instantiation of this ontology, and a stereotype profile assigns relevance values for various attribute values of the ontology. For example, a movie profile may have Bruce Willis and Samuel L. Jackson as actors, and a stereotype profile may assign to Bruce Willis as an actor the value 0.97 and to Mel Gibson as an actor the value 0.85 while assigning to Mel Gibson as a director the value 0.67.

Receiving recommendations for stereotypes can be done by matching item profiles with the stereotype profile, resulting in relevance values over media items.

A user in our domain is modeled by an affinity list to stereotypes, which is organized as a vector. A user may belong for example to a stereotype titled "Action" with relevance 0.8 and to a stereotype titled "Comedy" with relevance 0.7.

In the proposed method, we generate recommendations based on the relevant stereotypes.

First, we need to compute recommendations for the stereotypes. As a stereotype describes a content-based profile, explicitly stating preferences over the possible values of item attributes, we activate a matching engine that computes the relevance of a media item to a stereotype. As the number of stereotypes is not expected to be too high, these lists can be persistent in the database. Moreover, it is expected that many items will have low relevance to stereotypes so the lists can be truncated after some threshold.

Once a request for a recommendation for user u with affinity vector v is received, we compute the relevance of media item i to user u as follows:

$$relevance(i,u) = \sum_{s \in stereotypes} v(s) relevance(i,s) \quad (1)$$

where $relevance(i,s)$ is the *persisted* relevance of item i to stereotype s. Note that this process is much faster than matching each user with all items in the database using the matching engine, and therefore can scale up much better.

Our system supports both positive and negative feedbacks. We use binary like/dislike feedbacks since we assume that users find it easier to supply feedbacks of this type rather than numeric values. Nevertheless, our system can be easily adapted to handle numeric feedbacks as well. The affinity vector is updated according to the user's feedback.

3 The AHP Method for Preferences Elicitation

When a new user registers into the system, her initial profile need to be created. Research in the area has mainly focused on using a set of examples (e.g. a number of movies the user likes) or through a form specifying the user interests [17]. Such approaches have drawbacks — while rating observed movies is a painless process, using only a set of rated movies can later cause the system to only recommend movies that are similar to the ones the user rated. Furthermore, filling forms is usually considered a boring chore by users which consequently causes them to either avoid filling the forms or answer questions arbitrarily (e.g. always picking the first answer).

Methods that ask the user to specify her preferences over item attributes are also known as preference elicitation or preference based search. Viappiani *et al.* [24] describe a preference elicitation method and an example-based critiquing, that avoids the problems of preference fluency, domain knowledge and user effort.

In this paper, we adopt the Analytic Hierarchy Process (AHP) [19, 20] in order to elicit preferences to users. AHP is a decision support tool that was originally developed to solve multi-criteria decision problems. It uses a multi-level hierarchical structure of objectives, criteria, subcriteria, and alternatives. By using pairwise comparisons, it can obtain the weights of importance of the decision criteria, and the relative performance measures of the alternatives in terms of each individual decision criterion. In the domain of information systems, AHP has been used in the past for helping decision makers to select the best IT solution (see for instance [15]). While AHP has been extensively studied

in the Operation Research community, there are very few attempts to use it in recommendation systems, most recently by [8].

When using the AHP as a method for eliciting user preferences, the user has to express her preferences by responding to the upper triangle of a comparison square matrix (known as the judgment matrix [19]). Every element in the matrix refers to a single pairwise comparison, in which the user is asked to select the preferred alternative from two possible alternatives. Usually, the answer is chosen from several discrete choices. It was previously shown in psychological studies [9] that due to the human capacity, the best number of discrete choices is 7 ± 2, each of which is a linguistic phrase such as "I much prefer alternative A to alternative B" or "I equally like alternative A and alternative B".

The linguistic phrase selected by the user is then quantified by using a scale. Such a scale is a one-to-one mapping between the set of discrete linguistic choices and a discrete set of numbers representing the importance or weight. Saaty [20] suggests to match the linguistic phrases to the set of integer values $i = 1, ..., 9$ in order to represent the degree alternative A is preferred over alternative B. In this case, 1 represent that both alternatives are equally preferred. Similarly, the inverse numbers $(1/i)$ are used to represent the degree to which alternative B is preferred over alternative A.

After answering all questions, the next step is to determine the affinity vector implied by the comparisons. One way to achieve this is to calculate the right principal eigenvector of the judgment matrix [19]. There are other methods for converting judgment matrix into an affinity vector [5]. In this paper, we are using the eigenvector method due to its popularity.

Given a judgment matrix with pairwise comparisons, the corresponding maximum left eigenvector is approximated by using the geometric mean of each row i.e. the k-th root is taken from the product of the elements in each row (where k is the number of stereotypes). Next, the numbers are normalized by dividing them by their sum. Figure 1 illustrates the AHP method on a given judgment matrix.

Using AHP as-is requires the user to answer to a total of $\frac{k \cdot (k-1)}{2}$ comparison questions. If there are $k = 20$ alternatives then the user is required to answer a set of 190 questions. This drawback is a well known issue in AHP [4] and sometimes referred to as the *information overload* problem [3].

Harker [6] noticed that some questions are redundant and are necessary only for crosschecking judgments, measuring consistency and increasing the affinity precision. This lead to the development of a procedure for calculating AHP priorities with missing judgements [6]. This procedure is known as the Incomplete Pairwise Comparison (IPC) algorithm. It reduces the required number of comparisons to be between $(k-1)$ to $\frac{k \cdot (k-1)}{2}$. Moreover, it uses a gradient procedure for choosing the next comparison, and stopping rules to terminate the procedure after sufficient redundancy has been achieved. Nonetheless, making fewer judgements incurs a loss of accuracy [3].

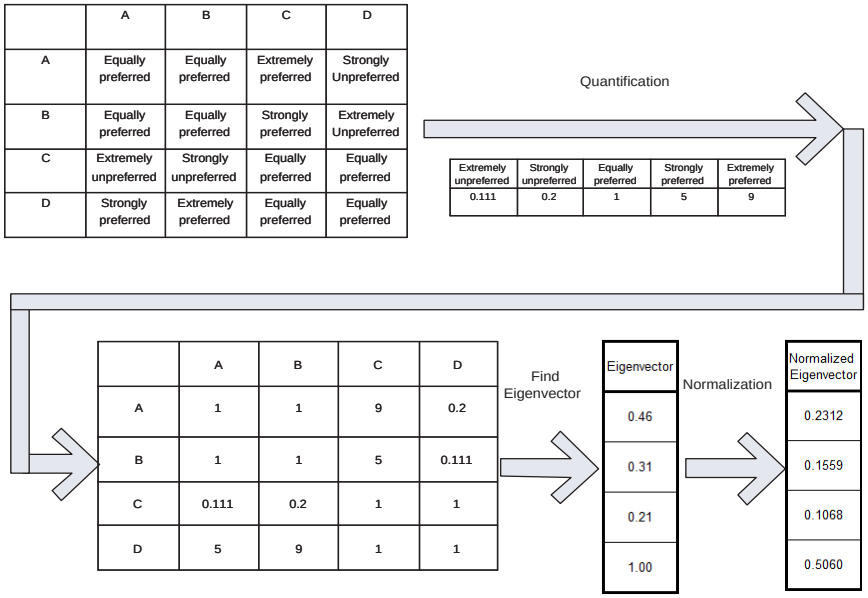

Figure 1: Illustration of the AHP Method for Preferences Elicitation

4 Anytime AHP for Elicitation of Initial Stereotype Affinity Vector

In this paper, we are mainly interested in the use of AHP as a method for extracting user preferences regarding various stereotypes. Specifically, in the judgment matrix, the user is asked to select one out of two possible stereotypes. After the user fills in the judgment matrix, the stereotype affinity vector is determined by using the eigenvector method described in Section 3.

We propose to develop an anytime approach i.e. the user may answer any desired number of questions. Whenever the user stops answering questions we have some classification of the user into stereotypes. The advantages of the anytime approach for preferences elicitation have been already studied in the past [13].

Our anytime approach is based on a decision tree, with pairwise comparisons at the decision nodes. Each node, both leaves and inner nodes, is also assigned an affinity vector of stereotypes, so if the user does not wish to answer more questions, the *current* affinity vector will be assigned to her. The more questions the user answers the more specific her affinity vector becomes. Furthermore, even without answering any question we are still able to assign some affinity vector under the assumption, for example, that most of the users of such systems tend to be teenagers. Figure 2 illustrates the root of the decision tree. Each inner node represents a different pairwise comparison question. Based on the response to the first (root) question, the decision tree is used to select the next pairwise comparison question to be asked. Every path in the decision tree represents a certain set of questions that the user is asked. Assuming that there are k stereotypes then the longest possible path contains $\frac{k\cdot(k-1)}{2}$ inner nodes (questions).

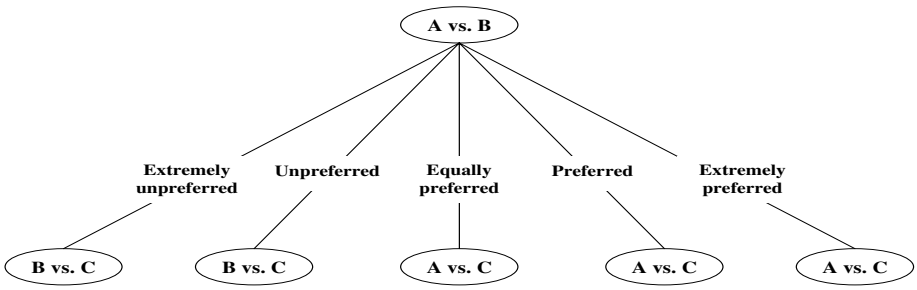

Figure 2: An illustration of the pairwise decision tree.

When a new stereotype model is reconstructed, one should also create a new hierarchical questionnaire in order to be able to assign new subscribers, which join the service, to the new stereotype model.

We suggest the use of a supervised inducer for decision trees. In order to use an inducer, a training set must be created and provided to the algorithm. The target attribute in this training set is the stereotype association. Each input attribute in the training set refers to a pair of media items, for which the user will be requested to provide her preferences. Note that all these input attributes are nominal and can be assigned to one of the seven values described in Section 3.

In order to create instances for the training set, we enumerate all possible combinations of the judgment matrix. For each combination, the affinity vector is calculated using the eigenvector method described in Section 3. Each combination is represented by k instances (rows) in the training set, one instance for each stereotype. The instance is labeled with the corresponding stereotype label and is weighted according to the appropriate value in the normalized eigenvector. If there are n choices in each question and there are k stereotypes, then the training set contains $k * n^{k*(k-1)/2}$ instances. The same judgment matrix can be obtained by switching stereotypes (i.e. the last value should be divided by $k!$). Assuming $n = 5$, the full enumeration (also known as the full factorial design) becomes unpractical for $k > 5$.

Because full factorial design is not practical in our case, we use Design of Experiments (DoE) methods [11] to create the training set. Design of experiments seeks an efficient way to collect useful information. The collected data is then analyzed by statistical methods in order to deduce valid and objective conclusions. We use fractional factorial designs, such as the Taguchi methodology [23], which employ a special set of orthogonal arrays. Table 1 illustrates a standard orthogonal array for $L8(2^7)$ design for 7 binary attributes. Each row represents the levels of the selected attributes. The design consists of 8 experiments instead of 128 experiments required by a full factorial design ($2^7 = 128$). It is important to note that every attribute setting has the same number of occurrences in every test setting of all other factors. Any 2 columns form a two-factor complete factorial design. The $L8(2^7)$ design reduces the number of experiments by 120. The reduction in larger arrays (like in our case) is even more substantial.

After creating the training set, we need to induce the decision tree. For this purpose we need a decision tree inducer which can take into account the

Table 1: The $L8(2)^7$ Taguchi's design.

Instances	Attributes						
	a_1	a_2	a_3	a_4	a_5	a_6	a_7
1	1	1	1	1	1	1	1
2	1	1	1	2	2	2	2
3	1	2	2	1	1	2	2
4	1	2	2	2	2	1	1
5	2	1	2	1	2	1	2
6	2	1	2	2	1	2	1
7	2	2	1	1	2	2	1
8	2	2	1	2	1	1	2

instances' weights and is also capable of providing probabilistic classification (namely, attach an affinity vector to every node in the tree). The well-known C4.5 algorithm [14] is suitable for this purpose [1]. The affinity vector that is assigned to a specific node is the mean of all training instances vectors that belong to the node.

The learning task discussed here can be viewed as an extension of the Multilabel Classification problem. In multilabel learning, each instance in the training set is associated with a set of labels and the task is to produce a label set whose size is unknown a priori for the unseen instance [26].

A task that is more closely related to the problem at hand is the label preferences learning problem [1] in which the goal is to predict, for any instance, a preference relation among a finite set of labels. Brinker *et al.* [1] differentiate between two principal approaches to address preference learning tasks: the utility function and the preference relations. The problem discussed here uses the utility function approach. Each instance is associated with a set of labels (stereotypes); each label has a different weight (affinity/utility). In order to obtain a ranking, the labels are ordered according to their utility scores. Nevertheless, the problem solved here focuses on the mapping of the judgment matrix (pairwise relations) into a utility function, and not on learning the relation that exists between the instance input attributes (such as user demographic characteristics) and their corresponding utility values. Although the mapping function from the judgment matrix to the utility vector is known in advance, we train a decision tree in order to find the optimal order of the questions. This is needed in case the user quits the process before completing the entire questionnaire.

After constructing the decision tree, we can transform each node in the tree (which refers to a pair of stereotypes) into a simple visual question that the user can easily answer. Note that the creation of the decision tree was performed independently of the stereotype definitions. The only parameter that matters is the size of the set of stereotypes. Thus, the construction of decision trees of frequent sizes can be performed in advance.

Because the user has no notion of the system's stereotypes, we propose to represent the pairwise comparison of two stereotypes (s, t), by the posters of

[1] C4.5 is using the gain ratio splitting criterion. In this case other splitting criteria might be useful, particulary the MSE gain criterion.

two popular media items that achieve maximal discrimination between these two stereotypes. An example of such a question can be seen in Figure 3. In order to select media items of this kind, we look for a pair of items (i, j) for which the term: $relevance(i, s) - relevance(i, t) + relevance(j, t) - relevance(j, s)$ is maximized. Additionally, to maximizing the above measure, we also add the following constraints:

- The two items should have a similar high popularity rate and have been published in about the same time. This constraint is required to ensure that the user identifies the items and that there is no bias other than her own personal preferences.
- The same item cannot be used twice i.e. in two different pairwise comparisons. This avoids boring the user. More importantly, we get a better picture of the user preferences. Counting on a single item to represent a stereotype may mislead. Let us assume that we have an "action" stereotype and a single movie of "Mel Gibson" was selected to represent it. If the user dislikes this specific actor, but likes action movies, we do not capture correctly the user's preferences. To satisfy this constraint, while traversing the tree top-down, we select a pair of items without replacement.

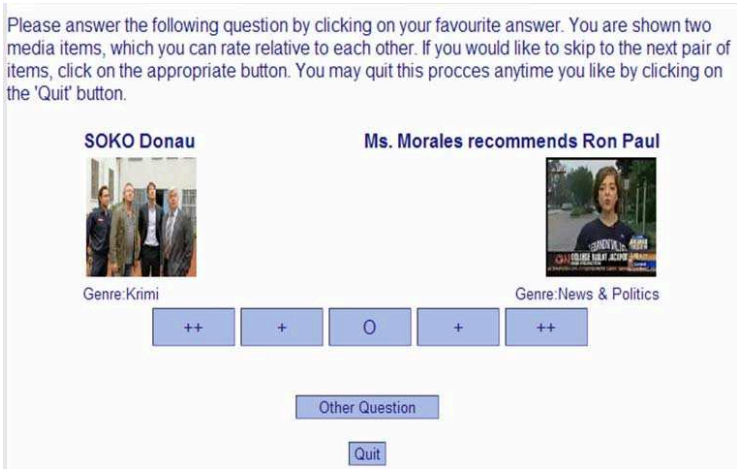

Figure 3: MediaScout questionnaire example.

5 Experimental Evaluation

To evaluate the predictive power of the proposed method, we compared our approach to a number of other algorithms using a prediction test.

5.1 Evaluation Measures

Several evaluation metrics have been proposed in the literature for multilabel classification problems [18]. Brinker *et al.* [1] propose to use two measures that

are more suitable to ranking problems: Spearman rank correlation [22] and Kendall's tau [7]. The Spearman rank correlation calculates the sum of squared rank-distances and is normalized such that it evaluates to -1 for inverse rankings and to $+1$ for identical ones. Kendall's tau is defined by the minimal number of pairwise inversions of (adjacent) labels needed to transform the first ranking into the second one.

In the present study, the training data contains the actual utility scores. Accordingly, we follow [3] and use the mean squared error (MSE)[2] as a goodness-of-fit measure. Formally, the MSE of the matrix i is defined as:

$$SSE_i = \sum_{s=1}^{k} \left(v_i(s) - v_i^F(s)\right)^2 \tag{2}$$

$$MSE_i = \frac{SSE_i}{k} \tag{3}$$

where:

- k is the size of the input matrix
- $v_i(s)$ are the normalized weights obtained for sth factor from the decision tree for the instance (matrix) i
- $v_i^F(s)$ are the normalized weights obtained for the s-th factor from the full matrix i
- SSE_i is the sum of the squared errors from the full matrix i
- MSE_i is the mean squared error

Since the MSE value strongly depends on the number of questions that are presented to the user [3], we measure it as well. The mean number of questions the user was asked to respond can be estimated by weighting the size of all paths in the generated decision tree. Assuming that the judgment matrix is uniformly distributed, then the expected mean is given by:

$$MQL = \sum_{i=1}^{q} \frac{n^{k(k-1)/2-l_i}}{n^{k(k-1)/2}} l_i = \sum_{i=1}^{q} \frac{l_i}{n^{-l_i}} \tag{4}$$

where q is the number of leaves (paths) in the tree and l_i is the length (number of questions) of the ith path.

5.2 Comparative Results

In all cases, the tree was constructed using the C4.5 induction algorithm[3]. This experiment was executed using the "unpruned" option, namely the pruning process was disabled. The size of the tree was controlled by changing the min-NumObj parameter, which indicates the minimum number of instances needed for every node in the tree (splits are not performed when this constraint is not fulfilled). We compared the results of C4.5 with randomly selecting questions in the judgment matrix.

[2]The MSE is one of the most commonly used measures of success for numeric predictions.
[3]Using the Weka environment

Figure 4 shows the MSE obtained by the proposed method and the random method for a 4×4 judgment matrix. Because this is a relatively small matrix the decision tree is trained on the full enumeration training set. Similarly the MSE is evaluated on the entire enumeration set. The x-axis represents the mean number of questions the user has answered. As expected the MSE measure decreases as more responses are provided. Both the random method and the decision tree based method begin with the same MSE level (i.e. before the user started the questionnaire) and reach the same MSE level of zero when the user fully completed the questionnaire. However, the proposed method has reduced the MSE value by 20% between these limits.

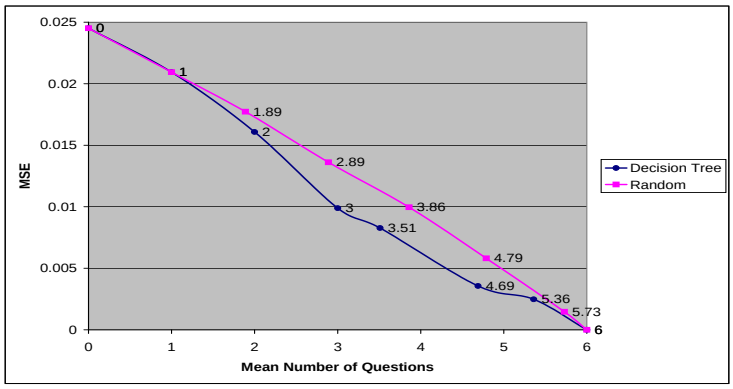

Figure 4: MSE performance for 4x4 judgment matrix

Figure 5 shows the MSE obtained by the proposed method and the random method for a 10×10 judgment matrix. Since full enumeration is not feasible for this size of matrix, the decision tree was trained on a sample of $100,000$ instances using Taguchi's [23] method. The MSE was evaluated on a random test set of $100,000$ instances. Similar to the first experimental result, the MSE measure decreases as more responses are provided. For 4 questions, the proposed method has reduced the MSE by 8%.

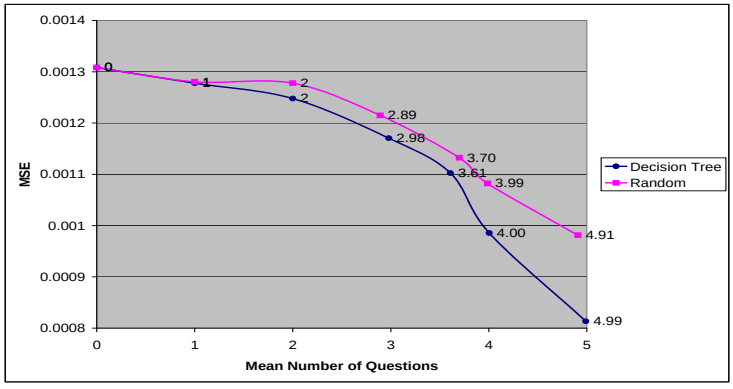

Figure 5: MSE performance for 10x10 judgment matrix

6 Discussion

Our proposed system uses a stereotype approach to elicit recommendations, incorporating principles from both the CB and the CF methodologies. It uses a hybrid algorithm that is designed to overcome the disadvantages of both approaches.

The task of classifying new users to stereotypes is a major concern for us. We ask users to answer a questionnaire in order to provide information about their preferences. The process of filling usual questionnaires is tiresome and users tend to either skip them or provide arbitrary answers when forced. We would like our system to make the answering process as easy as possible.

The approach proposed in the paper is an anytime AHP. It allows the user to stop answering questions whenever she chooses. We hope that users will complete the questionnaire, thus providing the system with as much information as possible. Nevertheless, even if the users stop before the completion of the questionnaire, we still have a classification of the user to stereotypes.

Our questionnaire is also easy to use — questions are short and require the user to select one of two items (e.g. movies). moreover, items are provided as images and the user is required to select her preferences using a slider.

The questionnaire is automatically constructed through a process of decision tree learning, aiming at achieving the information that best allows us to classify the user to various stereotypes. This automatic construction is executed after each update of the stereotype model.

The affinity vector assigned to a specific node is the mean of all training instances vectors that belong to the node. The use of the mean, ensures that the minimum MSE approximation for a given node is obtained. Formally, for a given node j, the mean is the optimal solution to the following optimization problem:

$$\begin{aligned} &\min \sum_{i=1}^{m_j} \sum_{s=1}^{k} \left(v_i^F(s) - x(s)\right)^2 \\ &s.t. \\ &\sum_{s=1}^{k} x(s) = 1 \end{aligned}$$

where m_j is the number of training instances that belong to the given node j and x is the decision vector to be found. Thus, if the training set contains the full enumeration of all judgment matrices, the mean provides the optimal value. Nevertheless, for large matrices, in which the training set is not exhaustive, there is no guarantee that the mean provides the minimum MSE. In this case, we can use the approximation used by the IPC method [6]. However, it requires that at least $k-1$ questions are answered.

The proposed method is executed in a batch mode, i.e. the entire training set is sampled before the training phase begins. This modus operandi is very restrictive because the number of training instances in each node decreases exponentially with the path size (number of questions). Thus, the proposed approach can be helpful only for selecting the first very few questions (for instance, in the second experimental result, which used a training set of $100,000$, we could suggest to select only the first 5 questions of a total of 45 questions). If more questions need to be selected (in case the user is willing to continue answering additional questions), we propose to extend only the specific path that

the user has selected. In order to do so, we sample more instances (judgment matrices) for this specific leaf and use the above mentioned splitting criterion to select the next question. This extension is performed on-line, based on the actual responses of the user.

Additional issues to be further studied include:

- Combining the proposed method with other approaches for decreasing the number of required pairwise comparisons. In particular, clustering methods for AHP [21] and the Balanced Incomplete Block Designs (BIBD) method that divides a large-scale AHP matrix into smaller subsets [25].
- In this paper we assume that the AHP with full judgment matrix is capable to capture the user's preferences. However, a field study is needed to validate this assumption.
- The proposed methodology needs to be examined by using other mappings for converting a judgment matrix into an affinity vector (other than the eigenvector method used in this paper).
- Examine evaluation measures other than the MSE, and adjust the splitting criterion to the evaluation measure (currently, we are using the information gain splitting criterion that is commonly used in the C4.5 algorithm).

7 Conclusions

A commercial recommender system for recommending media content, such as movie trailers and clips, to users of mobile phones is presented. It combines different approaches to recommendations, such as, expert systems, collaborative filtering and content based recommendations, into a single hybrid algorithm. The algorithm withholds the advantages of the various approaches while minimizing their disadvantages.

The present paper focuses on the way new users are introduced to the system through a questionnaire. It explains the mechanism of the AHP-based questionnaire. A method for using AHP in an anytime manner is proposed. For this purpose, we used a decision tree induction algorithm. The comparative study shows that the proposed hierarchical questionnaire is better than random traversal of the AHP questions.

The system is currently under development for commercial deployment within a large communication company, and is expected to be used in mobile phones.

References

[1] K. Brinker, J. Furnkranz and E. Hullermeier, Label Ranking by Learning Pairwise Preferences, *submitted to Journal of Machine Learning*, 2006.

[2] R. Burke, Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, 2002.

[3] F. J. Carmone, A. Kara and S.H. Zanakis, A Monte Carlo investigation of incomplete pairwise comparison matrices in AHP. *European Journal of Operational Research*, Volume 102(3):538–553, 1997.

[4] J. S. Dryer, Remarks on the analytic hierarchy process, *Management science* 36(3), 249–258, 1990

[5] B. Golany and M. Kress, A multicriteria evaluation of methods for obtaining weights from ratio-scale matrices, *European Journal of Operations Research*, 69 pp. 210–220, 1993.

[6] P. T. Harker, The incomplete pairwise comparisons in the analytic hierarchy process, *Mathematical and Computer Modelling*, 9(11):837–848, 1987

[7] M. G. Kendall, *Rank correlation methods*, Charles Griffin, London, 1955.

[8] D. Liu and Y. Shih, Integrating AHP and data mining for product recommendation based on customer lifetime value, *Information and Management*, V. 42 No. 3, p.387–400, March 2005.

[9] G. A. Miller, The Magical Number Seven, Plus or Minus Two: Some Limits on our Capacity for Processing Information, *Psychological Review*, 63, 81–97, 1956.

[10] M. Montaner, B. Lpez, and J. L. De La Rosa, A taxonomy of recommender agents on the internet. *Artificial Intelligence Review*, 19:285–330, 2003.

[11] D. C. Montgomery, *Design and Analysis of Experiments*, 4th edition, Wiley, New York, 1997.

[12] J. Orwant. Heterogeneous learning in the doppelgänger user modeling system. *User Model. User-Adapt. Interact.*, 4(2):107–130, 1995.

[13] P. Pu, B. Faltings and M. Torrens. User-Involved Preference Elicitation. *In workshop notes of the Workshop on Configuration, the Eighteenth International Joint Conference on Artificial Intelligence (IJCAI'03)*, pages 56–63, August 2003.

[14] J. R. Quinlan. *C4.5: Programs For Machine Learning*. Morgan Kaufmann, Los Altos, 1993.

[15] S. Ray, An Ahp Based Decision Model To Evaluate Various E-Learning Packages, *6th International Conference on Enterprise Information Systems*, 2004.

[16] P. Resnick and H. R. Varian. Recommender systems. *Communications of the ACM*, 40(3):56–58, 1997.

[17] E. Rich. User modeling via stereotypes. *Cognitive Science*, 3(4):329–354, 1979.

[18] J. Rousu, C. Saunders, S. Szedmak and J. Shawe-Taylor, Kernel-Based Learning of Hierarchical Multilabel Classification Models, *Journal of Machine Learning*, 7:1601–1626, 2006.

[19] T.L. Satty, A Scaling Method for Priorities in Hierarchical Structures. *Journal of Mathematical Psychology*, 1977.

[20] T.L. Satty, *The Analytic Hierarchy Process*. McGraw-Hill International, New York, U.S.A., 1980.

[21] T.L. Satty, The Seven Pillars of the Analytic Hierarchy Process, In Kksalan M. (ed.), Multiple Criteria Decision Making in the New Millennium. *Proceedings of the 15th International Conference, MCDM*, Springer. *Lecture Notes Econ. Math. Syst.* 507, Berlin, 15-37, 2001

[22] C. Spearman, The proof and measurement of association between two things. *American Journal of Psychology*, 15:72-101, 1904.

[23] G. Taguchi and S. Konishi, *Orthogonal Arrays and Linear Graphs*, American Supplier Institute Inc., Dearborn, MI (1987).

[24] P. Viappiani, B. Faltings and P. Pu, Evaluating Preference-based Search Tools: a Tale of Two Approaches. *In Proceedings of the Twenty-first National Conference on Artificial Intelligence (AAAI-06)*, Boston USA, July 16-20, 2006.

[25] E. N. Weiss and V. R. Rao, AHP Design Issues for Large Scale System, *Decision Science*, Vol. 18(1), pp. 43–61, (1987).

[26] M. Zhang and Z. Zhou, Multilabel Neural Networks with Applications to Functional Genomics and Text Categorization, *IEEE Transactions on Knowledge and Data Engineering*, 18(10):1338-1351, October, 2006.