Preference Elicitation for Narrowing the Recommended List for Groups

Lihi Naamani-Dery[1,3], Meir Kalech[2], Lior Rokach[2,3], Bracha Shapira[2,3]
[1]Department of Industrial Engineering and Management at Ariel University, [2]Department of Information Systems Engineering at Ben-Gurion University, [3]Telekom Innovation labs at Ben-Gurion University
{ln,kalech,liorrk,bshapira}@bgu.ac.il

ABSTRACT

A group may appreciate recommendations on items that fit their joint preferences. When the members' actual preferences are unknown, a recommendation can be made with the aid of collaborative filtering methods. We offer to narrow down the recommended list of items by eliciting the users' actual preferences. Our final goal is to output top-k preferred items to the group out of the top-N recommendations provided by the recommender system ($k < N$), where one of the items is a *necessary* winner. We propose an iterative preference elicitation method, where users are required to provide item ratings per request. We suggest a heuristic that attempts to minimize the preference elicitation effort under two aggregation strategies. We evaluate our methods on real-world Netflix data as well as on simulated data which allows us to study different cases. We show that preference elicitation effort can be cut in up to 90% while preserving the most preferred items in the narrowed list.

Categories and Subject Descriptors

H.3.3 1 [**Information Storage and Retrieval**]: Information Search and Retrieval

General Terms

Algorithms, Human Factors, Experimentation

Keywords

Preference Elicitation, Group Recommender Systems

1. INTRODUCTION

Group recommendation may be useful in various common daily scenarios. For example, a group who wishes to dine together may appreciate recommendations on relevant restaurants that fit their joint preferences [11]. Given the list of recommended restaurants, the group's members often interact among themselves to narrow down the possible options and choose a restaurant [10]. When the members' actual preferences regarding the recommended items are known, some aggregation strategy can be used to compute the most preferred restaurant out of the list of recommended items. However, often full preferences are not readily available, mainly because it can be tiresome or difficult for users to determine all of their preferences in advance. Consider for example a setting of 30 optional dinner locations for a group. The group members might be reluctant to provide input on each of the restaurants.

The problem we investigate in this paper is how to narrow down the list of items recommended by the system by eliciting the users' actual preferences. Since preference elicitation requires time and effort, the goal is to stop elicitation as soon as possible. We borrow from the closely related domain of social choice [2], where preference elicitation has been examined under different voting protocols. In the worst case for most voting protocols all the preferences are needed in order to determine a winner [3]. However, in practice it has been shown that the required information for determining a winner can be cut in more than 50% [6, 8, 10]. Given partial preferences, it is possible to define the set of *necessary* winners, i.e., items which must necessarily win given a certain aggregation strategy, as well as the set of possible winners, i.e., items which can still possibly win [7]. These definitions enable the elicitor to determine whether enough information about the users' preferences is available, and iteratively request for more user ratings. Once a necessary item is identified, it is recommended to the group. An additional benefit of this process is that the output is not a predicted item, but rather an item that *certainly* fits the group's joint preferences.

Although outputting a definite winner is the most accurate result, there are advantages in providing top-k items (where one of them is a necessary winner) out of the top-N recommendations list ($k < N$). Less elicitation effort is required for outputting top-k items than for outputting one necessary winner (i.e., $k = 1$). Thus issues such as elicitation costs are reduced. Also, users might prefer to receive a few options rather than just one. Thus, if a certain alternative is unavailable, the group can quickly switch to another alternative without applying an additional preference elicitation process. For example, if the system recommends only a fish restaurant as the winner item, but one of the group members dislikes fish, the group might prefer to switch to another alternative recommended by the system rather than to perform another elicitation round.

Of course, the top-k winners list depends on the applied aggregation strategy. The aggregation strategy should be a fair one. In his famous work, Arrow shows that there is no perfect aggregation system [1]. One of the major differences between aggregation strategies is the social environment in which they are used. The emphasis can be either on mindfulness toward the individual user or on mindfulness toward the majority of the group [5]. Two state-of-the-art aggregation strategies that hold this tension in between are the *Majority Based Strategy* and the *Least Misery Strategy*. In the *Majority Based Strategy* the users' ratings of the different items are aggregated and the item with the highest total value is the winner. In the *Least Misery Strategy* the chosen item cannot be the least preferred by any of the users [9]. Preferences elicitation methods usually assume that the Majority based strategy is employed [6, 8, 10]. We show that the aggregation strategy affects the effort required in the preference elicitation and evaluate two state-of-the-art aggregation strategies.

RecSys'14, October 6–10, 2014, Foster City, Silicon Valley, CA, USA.
Copyright © 2014 ACM 978-1-4503-2668-1/14/10…$15.00.
http://dx.doi.org/10.1145/2645710.2645760

Our final goal is to output top-k preferred items to the group out of the top-N recommendations provided by the recommender system (k<N), where one of the items is a *necessary* winner. We propose an iterative preference elicitation method, where users are required to provide item ratings per request. We suggest a heuristic that attempts to minimize the preference elicitation effort under two aggregation strategies, while still outputting a necessary winner. We evaluate our methods on real-world Netflix data as well as on simulated data which allows us to study different cases. We show that the preference elicitation effort can be cut in up to 90% in some cases.

We focus on the Range voting protocol which requires users to submit ratings to items. Applications that ask for ratings are quite common, for example Amazon and Netflix[1]. We assume that users' preferences are unknown in advance, but can be acquired during the process, i.e., if a user is asked for her preference on an item she is able to submit it. We also assume a user submits her true preferences and thus in this paper we do not consider manipulation. Lastly, we assume an approximate distribution of each user's preferences for each item exists, as in [10].

2. RELATED WORK

Both [9] and [13] study how different strategies affect group members. However, the aggregation strategies have not been studied in the context of *preference elicitation*. Practical preference elicitation is gaining interest recently. In [6] it is assumed that each user holds a predefined decreasing order of the preferences. In an iterative process, the users are requested to submit their highest preferences; the request is for the rating of one item from all the users. However, requiring the users to predefine their preferences can be inconvenient to the users. In [10] two practical heuristics that attempt to minimize preference elicitation effort are presented for the Range voting protocol. However both output one definite item and not top-k items. In [8] a practical elicitation process is proposed for the Borda voting protocol. The output is multiple winners. However the authors did not consider the Range protocol. In all of the above, the authors considered only the majority based aggregation strategy. The algorithm proposed in [8] lacks details and the authors did not provide us with explanations so we cannot expand it to the Least Misery strategy or to the Range voting protocol. In this paper we expand the preference elicitation algorithm for the Range voting protocol suggested in [10]. To the best of our knowledge the issue of preference elicitation and returning one or more items under the Least Misery strategy has not been investigated before.

3. MODEL DESCRIPTION

Let us define a set of users (voters) as $V = \{v_1, v_2, \dots, v_m\}$ and a set of candidate items as $C = \{c_1, c_2, \dots, c_n\}$. When queried, the users assign ratings to the items from a discrete domain of values $D = \{d_{min}, \dots, d_{max}\}$ where d_{min} and d_{max} are the lowest and highest values, respectively. User v_i's preferences are represented by the rating function $value: VxC \to D$. For abbreviation we denote q_j^i as a single preference of v_i for a single item c_j. The user-item rating q_j^i is not necessarily known to the voting center. Let $O^i = \{q_p^i, \dots, q_t^i\}$ represent the set of user v_i responses to queries. Note that this set does not necessarily contain all the items. $O^A = \{O^1, \dots, O^m\}$ is a set of O^i sets. In an iterative process, in each round one user is queried for her ratings on one item. The preferences are aggregated according to a predefined aggregation strategy. The process terminates once a predefined termination condition is reached.

[1] www.amazon.com, www.netflix.com

We denote the applied aggregation strategy str. The item score depends on the strategy used. As mentioned, in the Majority based aggregation strategy the mindfulness is towards the majority of the group. The user rating for an item are added: $s_{majority}(c_j) = \sum_{i\in=\{1,\dots,m\}} q_j^i$. Although the Majority Based Strategy is fairly common, a disadvantage of this strategy is that it can be unfair towards users with the minority view. In fact, the authors in [14] state that their system works well for a homogenous group. However, when the group is heterogeneous, dissatisfaction of the minority group occurs. In the Least Misery strategy, the chosen item cannot be the least preferred by any of the users. The items score is the score of the least preferred item: $s_{least}(c_i) = \min_{i\in=\{1,\dots,m\}} q_j^i$. An implementation can be found in [12] for example. The disadvantage is that the minority opinion can dictate the group – if all users but one really want some item to win, it will not be chosen [9].

Let k be the number of alternatives that will be presented to the group members. When given a set of responses to queries O^A, the goal is to determine whether the iterative process can be terminated. The termination condition is that the winner item is definitely one of the k items. The function $pmax^A(c_j, O^A)$ computes the possible maximum rating for item c_j, given the known user preferences. Similarly, the function of the possible minimum rating is $pmin^A(c_j, O^A)$.

Definition 1. *(Possible Maximum): given the set of responses O^A and an aggregation strategy str, the possible maximum score of candidate c_j, denoted $pmax^A(c_j, O^A)$, is computed as follows:*

$$pmax^A(c_j, O^A, str) = \begin{cases} \sum_{i\in 1..m} pmax^i(c_j, O^i) & str = majority \\ min_i\left(pmax^i(c_j, O^i)\right) & str = least\ misery \end{cases}$$

$$where\ pmax^i(c_j, O^i) = \begin{cases} d_g & if\ \exists q_p^i = d_g \in O^i \\ d_{max} & otherwise \end{cases}$$

Definition 2. *(Possible Minimum): given the set of responses O^A and an aggregation strategy str, the possible minimum score of candidate c_j, denoted $pmin^A(c_j, O^A)$ is computed as follows:*

$$pmin^A(c_j, O^A, str) = \begin{cases} \sum_{i\in 1..m} pmin^i(c_j, O^i) & str = majority \\ min_i\left(pmin^i(c_j, O^i)\right) & str = least\ misery \end{cases}$$

$$where\ pmin^i(c_j, O^i) = \begin{cases} d_g & if\ \exists q_p^i = d_g \in O^i \\ d_{min} & otherwise \end{cases}$$

A necessary winner NW is a set of items whose possible minimum aggregated rating is equal or greater than the possible maximum aggregated rating of all the others. Formally:

Definition 3. *(Necessary Winners set): $NW = \{c_i | pmin^A(c_i, O^A) \geq pmax^A(c_j, O^A)\ \forall c_j \in C \backslash c_i\}$*

Although the necessary winner set may contain more than one item, we assume that there is only one necessary item. In the case of a few winning items, the first item is selected lexicographically. In order to decide on the next query, the elicitor considers the rating distribution of the user-item preferences. The distribution can be inferred from rankings of similar users using collaborative filtering methods [10]. Formally the rating distribution is:

Definition 4. *(Rating Distribution): the voting center considers q_j^i as a discrete random variable distributed according to some rating distribution vd_j^i, such that $vd_j^i[d_g] \equiv P_r(q_j^i = d_g)$.*
Where $vd = \{vd_1^1, vd_2^1, \dots vd_m^1, vd_1^2, vd_2^2, \dots, vd_m^2, \dots, vd_m^n\}$ is the set of all the rating probability distributions.

The example presented in Table 1 shows the rating distribution of three users for two items in the domain $D = \{1,2,3\}$. E.g., the probability that v_1 will assign a rate of 1 to item c_1 is 0.2.

We assume independence between the probability distributions. While the independence assumption is naive, it can be used for approximating the actual probability. An attempt to address dependency will yield probabilities that are too complex for a system. When facing the tradeoff between the models accuracy and practicality, we chose to model a practical system. However, note that the precise probability value is not required if the queries are still sorted correctly according to the value of the information they hold (their informativeness). In the closely related domain of machine learning, a similar naive assumption is known to provide accurate classification, though the independence assumption is not always true [4]. We therefore argue that the decrease of accuracy, if at all exists, is not significant.

The initial rating distribution is calculated before the heuristics are applied. Both heuristics iteratively reveal one new rating upon request. This allows updating the distribution every time a new rating is added. The accuracy is expected to grow with the number of ratings acquired. Note that the proposed algorithm can be used when no history of ratings is available (this is known as cold start). In such a case, the initial distribution will be uniform, and will be updated as we proceed.

Table 1. Rating distribution of 3 users, $V = \{v_1, v_2, v_3\}$

	v_1		v_2		v_3	
	c_1	c_2	c_1	c_2	c_1	c_2
$d_1 = 1$	0.2	0.2	0.4	0.5	0.3	0.7
$d_2 = 2$	0.2	0.2	0.3	0.2	0.3	0.1
$d_3 = 3$	0.6	0.6	0.3	0.3	0.4	0.2

4. The Preference Elicitation Heuristic

The Extended Dynamic Information Gain (*EDIG*) Heuristic is an iterative algorithm based on the DIG algorithm proposed in [10]. It uses a greedy calculation in order to select a query out of the possible $m \times n$ queries. The chosen query is the one that maximizes the information gain. The information gain of a specific query is the difference between the prior and the posterior probability of the candidates to win given the possible responses to the query. The algorithm terminates once a winner is within the requested top-k items. In order to select a query, the heuristic calculates the information gained from each one of the optional queries and then selects the one that maximizes it. To compute the information gain, the winning probability of each item is calculated. The score s that the candidate can achieve depends on the strategy:

$$S = \begin{cases} \{n \cdot d_{min}, \dots, s_{majority}, \dots, n \cdot d_{max}\} & if\ str = majority \\ \{d_{min}, \dots, s_{least}, \dots, d_{max}\} & if\ str = least \end{cases}$$

Let us denote the probability an item has a certain score $Pr(c_j = s)$. The probability of an item to win is:

Definition 5. *(Item Winning Probability): Under the independence of probabilities assumption, the probability that item c_j is a winner is the aggregation of c_j's probabilities to win over the possible ratings s:* $Pr(NW = c_j) = \sum_{s \in S, i \neq j} \Pr(c_j = s \land \forall\ c_i < s) = \sum_{s \in S} \Pr(c_j = s) \cdot \prod_{\forall i \neq j} \Pr(c_i < s)$

To compute the probability that an item will receive the score *s*, and to compute the probability that an item will receive a score of at most *s*, for the majority based strategy we use:

$$Pr(c_j = s) = \sum_{x=d_{min}}^{d_{max}} \left(\prod_{i=1}^{n-1} Pr(q_i^j = s - x) \cdot Pr(q_n^j = x)\right)$$

$$Pr(c_j < s) = \sum_{x=\min(S)}^{s-1} Pr(c_j = x)$$

For the Least Misery strategy we use:

$$Pr(c_j = s) = min_{x \in S} \left(\prod_{i=1}^{n-1} Pr(q_i^j = s - x) \cdot Pr(q_n^j = x)\right)$$

$$Pr(c_j < s) = min_{x \in S} Pr(c_j = x)$$

Next, the heuristic calculates the information gain of the $m \times n$ possible queries. The information gain of a query is the difference between the prior entropy and the posterior entropy, given the possible responses to the query:

$$H(NW) = -\sum_{j=1}^{m} Pr(NW = c_j) \cdot \log\left(Pr(NW = c_j)\right)$$

Definition 6. *(Information Gain): The Information Gain (IG) of a query is:* $IG(NW|q_j^i) = H(NW) - \sum_{g=min}^{max} H(NW|q_j^i = d_g) \cdot Pr(q_j^i = d_g)$ *where* $H(NW|q_j^i = d_g)$ *represents the entropy of NW given the possible values by querying user v_i on item c_j.*

The query that maximizes the information gain is selected: $argmaxIG_{i,j}(NW|q_j^i)$. The query selection process continues until the termination condition is reached.

5. EVALUATION

We examine the performance of the EDIG algorithm, under two aggregation strategies: Majority and Least Misery. The tradeoff between the number of items outputted (top-k) to the group and the number of preference elicitation required is investigated. Performance is measured in terms of elicitation effort, i.e., the required number of queries (requests for the user to provide ratings), in order to reach the termination condition. As mentioned in the related work, to the best of our knowledge there are no other algorithms that operate (or can be expanded to operate) under the same settings. Therefore the baseline for measuring the effectiveness of our method is a random procedure (RANDOM), which randomly selects the next query. To account for the randomness in RANDOM, each experiment was repeated 20 times.

We present an evaluation on simulated data as well as on a real-world dataset, the Netflix prize dataset. The Netflix dataset (http://www.netflixprize.com) consists of ~100,000 users and ~16,000 items. We consider a setting of a group of 10 members and 10 items. The users in the group are drawn randomly from a subset of Netflix where all users gave rating to 100 items. 90 items are used to create the initial rating distribution. The 10 remaining items are the items in question. To account for randomness, each experiment was repeated 10 times.

We simulate a cold start scenario where no history of ratings is available. We consider a setting of a group of 10 members and 10 items where the users rate their preferences on a scale of 1-4. We set the user-item distribution to a Uniform distribution. For example, the initial distribution for any user-item pair is: {0.25,0.25,0.25,0.25}, indicating that item c_j has a 25% probability to be rated 1-4 by user v_i. Similarly, we simulated a problem where one item is suspected to be more favorable and has a skewed distribution skewed to the right (Right-Skew), and a problem where one item is suspected to be less favorable and has a distribution skewed to the left (Left-Skew). The distribution values for the right and left skews were set to: {0.011,0.011,0.147,0.832} and {0.832, 0.147,0.011,0.011}. In all cases, we cast lots to set the user-item ratings. To account for randomness, each experiment was repeated 10 times.

We compared the two strategies: Majority (MAJ) and Least Misery (LM) combined with EDIG and RANDOM heuristics. We examined different termination conditions, from $k = 1$, i.e., requiring one definite winner, to $k = 9$, i.e., requiring the winner to be one of the top 9 items. The results on the Netflix data are displayed in Figure 1. Axis x presents the termination condition k=1,…,10, Axis y presents the number of queries required in order to find a winner within the top-k. As expected, all combinations (of heuristics with strategies) drop in the amount of elicitation effort required as the termination condition includes a larger number of k items. The best results are received for EDIG combined with MAJ strategy. This combination reduces the elicitation effort in up to 35%. All the results were found significant with a 95% confidence level using a one-tailed t-test.

Table 2. Preference Elicitation effort on simulated data

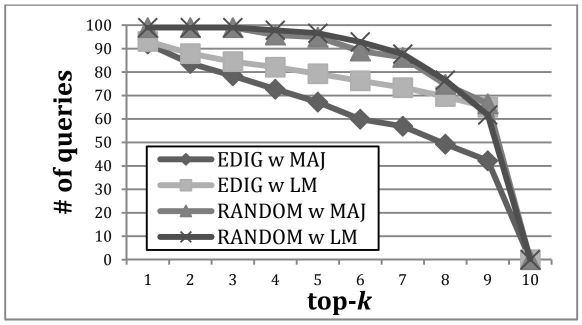

Figure 1. Preference Elicitation effort on the Netflix dataset

	Right-Skew		Uniform		Left-Skew	
topk	**MAJ**	**LM**	**MAJ**	**LM**	**MAJ**	**LM**
1	27.7	83	79.7	61.9	83	42.2
2	26.1	67.8	75.7	44.5	76.1	35.3
3	24.7	54.5	72.7	38.7	69.4	27.7
4	23.2	44.1	70.1	32.6	65.5	21.2
5	21.2	35.9	65.2	24.2	61.6	15.5
6	19.3	28.9	59.7	18.4	59.6	9.8
7	17.4	23.9	55.5	13.7	55.5	6.6
8	15.2	17.3	50.3	7.5	49.6	3.4
9	12	14.7	35.1	2.3	39.3	0.2

For the simulated data with different skewness levels: Right-Skew, Uniform (cold-start problem) and Left-Skew, EDIG significantly outperformed RANDOM with a confidence level of 95% using a one-tailed t-test. For a cold-start problem (Uniform skewness, columns 4-5 in Table 2) LM outperforms MAJ reducing the preference elicitation effort in 22%-93%. For a Left-Skew (columns 6-7 in Table 2), again LM outperforms MAJ for every k. The trend changes for the Right-Skew (columns 2-3). Here MAJ outperforms LM. All of these results were found significant with a confidence level of 95% using a one-tailed t-test. The results indicate that it is advised to consider the users assumed behavior. If the user preferences are assumed to be skewed towards some favorable item, as is the case with the Netflix dataset and with the simulated data with a right skew, it is best to use a heuristic that employs the Majority based strategy. If the results are uniformly skewed or it is assumed that some item or items are less preferred (left skew), it is best to choose a heuristics that employs the Least Misery strategy. Reducing the preference elicitation effort can also be achieved by changing the preference elicitation termination condition and requiring a winning item within top k instead of a definite item.

6. CONCLUSIONS

We propose an iterative preference elicitation method, where users are required to provide item ratings per request. We suggest a heuristic that attempts to minimize the preference elicitation effort under two aggregation strategies. The goal is to output top-k preferred items, where one of the items is a *necessary* winner, out of the top-N recommendations provided to the group by the recommender system ($k < N$). We show that the aggregation strategy affects the effort required in the preference elicitation and evaluate two state-of-the-art aggregation strategies. We demonstrate that the preference elicitation effort can be cut in up to 90% in some cases, while a necessary winner is still outputted. Since different aggregation strategies are used in different situations, in the future we plan to investigate other aggregation strategies and heuristics that combine a few strategies.

7. REFERENCES

[1] Arrow, K. J. 1951. *Social Choice and Individual Values.* Cowles Foundation, New Haven.

[2] Brandt, F., Conitzer, V. and Endriss, U. Computational social choice. 2013. In Weiss, G. ed. *Multiagent Systems.* MIT Press.

[3] Conitzer, V. and Sandholm, T. 2005. Communication complexity of common voting rules. In *Proceedings of the 6th ACM conference on Electronic commerce.* ACM, 78-87.

[4] Domingos, P. and Pazzani, M. 1997. On the optimality of the simple Bayesian classifier under zero-one loss. *J. Mach. Learning*, 29, 2, 103-130.

[5] Jameson, A. and Smyth, B. 2007. Recommendation to groups. *The adaptive web*, 596-627.

[6] Kalech, M., Kraus, S., Kaminka, G. A. and Goldman, C. V. 2011. Practical voting rules with partial information. *J. Auton. Agents Multi-Agent Syst*., 22, 1, 151-182.

[7] Konczak, K. and Lang, J. Voting procedures with incomplete preferences. 2005. In *Proceedings of the 19th IJCAI.* Edinburgh. .

[8] Lu, T. and Boutilier, C. Multi-winner Social Choice with Incomplete Preferences. 2013. *Proceedings of IJCAI-13.* 263-270.

[9] Masthoff, J. 2011. Group recommender systems: Combining individual models. *Recommender Systems Handbook*, 677-702.

[10] Naamani Dery, L., Kalceh, M., Rokach, L., Shapira, B. 2014. Reaching a Joint Decision with Minimal Elicitation of Voter Preferences. Information Sciences, Vol.278, 466-487.

[11] Naamani Dery, L., Kalech, M., Rokach, L. and Shapira, B. 2010. Iterative voting under uncertainty for group recommender systems. In *Proceedings of the 4th ACM RECSYS.* 265-268.

[12] O'connor, M., Cosley, D., Konstan, J. A. and Riedl, J. 2001. PolyLens: a recommender system for groups of users. In *proceedings of ECSCW 2001.*199-218.

[13] Senot, C., Kostadinov, D., Bouzid, M., Picault, J. and Aghasaryan, 2011. Evaluation of group profiling strategies. In *Proceedings of the 22nd AAAI.* AAAI Press, 2728-2733.

[14] Yu, Z., Zhou, X., Hao, Y. and Gu, J. 2006. TV program recommendation for multiple viewers based on user profile merging. *User Modeling and User-Adapted Interaction,* 16,63-82.