2019 IEEE International Conference on Web Services (ICWS)

Recommending Packages of Multi-criteria Items to Groups

Edgar Ceh-Varela, Huiping Cao
Department of Computer Science, New Mexico State University
Las Cruces, NM., USA.
Email: eceh@nmsu.edu, hcao@cs.nmsu.edu

***Abstract*—Most recommender services help individual users by recommending items within a single category based on the items' overall ratings. However, this may not be sufficient for group activities. For example, a group of friends using an online travel website look for a weekend getaway package (with hotel and restaurant), where the group members have different preferences over the characteristics (e.g., price, service, ambient) of these items. We call items with multiple characteristics as multi-criteria items. The items may come from different categories (e.g., hotel, restaurant). This paper proposes a novel problem of recommending packages of multi-criteria items to a group of users by leveraging users' preferences over categories. As far as we know, our work is the first paper studying this problem. We propose two models to measure the preference of a group to a package. The first model utilizes users' preferences for all the items and all categories, while the second model further leverages the influence of different group members to user preferences. We further introduce a new metric, to measure the fairness of the recommendations to different group members. We present an approach that utilizes co-clustering to incorporate items' characteristics in the calculation of user preferences and creating recommendations. Finally, we conduct extensive experiments with three real datasets. The experiments show that the second model can find packages that balance better the preferences of all the group members.**

***Keywords*-group recommender systems, fairness, multi-criteria items, item categories**

I. Introduction

Recommender systems suggest new items to users based on the item selections of these users or similar users in the past. Traditional recommender systems aim to help *individual* users by recommending a list of items from a *single* category, e.g., a movie, a song, or a restaurant. These recommendations are based on the items' *overall* ratings given by the users.

However, people participating in group activities may require recommendations for the whole group. These group activities could include a *weekend getaway package* to a group of tourists, a *set of courses* to a group of students, or *entertainment packages* to group of friends. Making such recommendations is not an easy task because these group activities may need to combine items from *different categories* (e.g., hotels, restaurants) to form a *package*, and that each group member cares about different *characteristics* or *features* (e.g., price, service, ambient, etc.) of these items. For example, in a group of two people on a *hotel-restaurant* package, one cares more about hotel price and restaurant service, while the other person cares more about the sleep quality of the room and the ambient of the restaurant. We call the items with ratings on multiple characteristics as *multi-criteria items*. In this paper, we propose a novel problem of recommending a *package* of *multi-criteria* items from *different* categories to a *group of users*. Systems for recommending multi-criteria items have been proposed [1], [2]. However, they only consider items from a single category and for single users (not a group of users). Recent works also have explored the direction of recommending packages of items to a group of users [3], [4]. However they do not deal with multi-criteria items. To the best of our knowledge, Mengash and Brodsky [5]–[7] is the only group who has worked on recommending packages of multi-criteria items to a group of users. Nonetheless, their methods are semi-automatic (details see Section II) and do not consider that items come from multiple categories.

Users have different levels of preferences for different categories, and this is reflected in the way they select items [8]–[10]. As far as we know, this work is the first to study recommending packages of multi-criteria items to user groups considering users' preferences for the different categories. There are several major challenges to solve our proposed problem. (i) When a group is temporary (i.e., formed for a single activity and then dissolve), its modeling is complex because there is no previous knowledge about group interaction. (ii) User preferences need to be properly modeled while considering the features (characteristics) and categories of items. (iii) A user may belong to multiple similar user-groups, where different groups of users share similar preference on different item features. Similarly, an item may belong to multiple similar item-groups because of their multiple features. (iv) The goodness of the recommended items needs to be properly evaluated such that every user is equally happy with the recommendation based on her/his preferences.

To address these challenges, this paper proposes the following models, metric, and method, which form the major contribution of this paper.

- We propose two models that generate multi-criteria item packages from different categories for a group of users.
- We design a method to predict missing ratings by applying co-clustering techniques.

978-1-7281-2717-0/19/$31.00 ©2019 IEEE
DOI 10.1109/ICWS.2019.00054

- We present a *balance* metric to evaluate the goodness of recommended packages.
- We conduct extensive experiments to show the feasibility of the proposed model, approach, and the metric.

This paper is organized as follows. Related works are reviewed in Section II. Section III formally defines the problem of recommending multi-criteria item packages to a temporary group of users. In Section IV we present our proposed algorithms to find equally liked item packages for all the group members. Section V presents our experimental evaluation. The final conclusions are presented in section VI.

II. Related Work

A. Group recommender systems.

Several works have proposed approaches to recommend packages of items to a group of users. Pujahari and Padmanabhan [11] combine the user-user and item-item Collaborative Filtering (CF) techniques to predict items that are common for the majority of the users in the group. Feng et al. [12] present a group recommender based on latent relationships. It calculates the group interests and item features by combining the users' preference profiles and items' content information. Zhu et al. [13] presents the PromoRec algorithm which recommends the most popular items for an automatically identified group of users by finding the items that are Group-Attractive and Group-Favorite. Qi et al. [3] propose two probabilistic models for recommending packages to a group of users. In [3], the concept of *expert user*, package *viability*, and group *fairness* are leveraged in the models to select the best packages for the group. Recently, a group consensus score function has been proposed [14] to facilitate recommending items relevant to the majority of a group and deprecating recommendations towards a few users. This function is based on user and item saturation functions.

All the above mentioned recommender systems use only the overall ratings of items, and do not utilize any ratings on specific item features.

B. Multi-criteria recommender systems.

Several systems have been developed to recommend multi-criteria items. Adomavicius and Kwon [15] propose a similarity-based method and an aggregation function-based method to incorporate multi-criteria information in CF recommender systems. Likewise, Liu et al. [16] find the item's dominant set of features. Then user clusters are formed according to their criteria preferences, and recommendations are based on ratings by other users from the same cluster. In the same way, Jannach et al. [17] make hotel recommendations based on multi-dimensional customer ratings. Multi-criteria ratings are used with regression-based methods. These recommender systems are for individual users rather than user groups.

C. Multi-criteria item-package recommender systems to groups of users.

Recommending a set of multi-criteria items to a group of users has been proposed in recent years. Mengash and Brodsky [7] propose a Group Composite Alternatives Recommender (GCAR), which uses the Instant Runoff Voting (IRV) to make composite recommendations to a group of users. GCAR is extended in [6] by proposing a Decision-Guided Group Package Recommender (DG-GPR), which uses a Hybrid Condorcet-Instant Runoff Voting. In [5] they extend their previous works to support large heterogeneous groups. However, all these methods are *semi-automatic*, because they require *actual* users to *manually* rank the set of candidate packages and make recommendations based on users' rankings.

All the above mentioned methods do not consider users' preference for the item's category. Furthermore, the user similarly is defined using one aggregated score from the multiple criteria. Our work is different in that we do not aggregate the effect of the multiple criteria in an early stage.

III. Problem statement

Our problem involves users, the items that users rated, and the categories that the items belong to. Let $\mathcal{U}$, $\mathcal{I}$, and $\mathcal{C}$ be the set of users, items, and item categories respectively. The set $\mathcal{U} = \{u_1, u_2, \cdots, u_U\}$ contains U users who have expressed their preferences for items. The set $\mathcal{I} = \{i_1, i_2, \cdots, i_I\}$ consists of I items. The set $\mathcal{C}$ comprises C different item categories $\mathcal{C} = \{c_1, c_2, \cdots, c_C\}$ (e.g., $\mathcal{C}$ = {hotel, restaurant, museum}). Each item $i \in \mathcal{I}$ must belong to one and only one category $c \in \mathcal{C}$, and must be rated at least by one user $u \in \mathcal{U}$. Let $c(i)$ denote the category of the item i and I_c be the total number of items in the category c. We suppose that each item $i \in \mathcal{I}$ is described by a set of features (characteristics) $\mathcal{F} = \{f_1, f_2, \cdots, f_F\}$. The number of features is different for different item categories. A user $u \in \mathcal{U}$ can rate a feature f of an item i with a rating r_{uif}, which is a non-negative real number. We consider the overall rating as one special item feature.

The ratings from a user u to an item i can be represented as an F-dimensional criteria vector $\vec{r}_{ui} = (r_{ui1}, r_{ui2}, \cdots, r_{uiF})$. We call these items with ratings on multiple features as *multi-criteria items*. The notations are summarized in Table I.

Table I: Important notations

Symbol	Meaning
$\vec{r}_{ui}$	Ratings from a user u to an item i on this item's features
$POC(u,c)$	User u's preference for a category c
$aPOC(u,c)$	User u's adjusted preference for a category c
$Prob(u,i_j)$	The degree that user u likes item i_j (used in Eq. 2)
$Pref(u,P)$	Preference of u over a package P (defined in Eqs. 2 and 3)
$Pref(G,P)$	Preference of a group G over a package P (defined in Eq. 4)

A. User preference over item categories.

Past works [8]–[10] show that, if a user rates more items in one category (say c_1) than in another category (say c_2), then this user prefers c_1 more than c_2. For example, a user who likes action movies is expected to rate more action movies than movies in other categories (e.g. drama, romance). Given this intuition, we define a user's preference over a category as follows.

Definition 1 *User preference over a category. Given a user u, her/his preference for an item category c, denoted as $POC(u, c)$, is defined as $POC(u, c) = \frac{I_{uc}}{I_u}$.*

Here, I_{uc} is the number of items rated by the user u in the category c, and I_u is the total number of items rated by the user u. The definition implies that $\sum_{c=1}^{C} POC(u, c) = 1$. For example, if a user u has ranked 10 movies in total: five on action movies (A category), three on dramas (D category), and two on romance movies (R category), then $POC(u, A) = \frac{5}{10}$, $POC(u, D) = \frac{3}{10}$, and $POC(u, R) = \frac{2}{10}$. Therefore, $POC(u, A) + POC(u, D) + POC(u, R) = 1$, and it shows that u prefers A over D and R, and prefers D over R.

The definition of $POC(u, c)$ does not take into consideration the fact that some categories have more items than others (e.g. there are more restaurants than museums). This fact indicates that the larger categories have more rated items than smaller categories. Accordingly, the percentage of items rated in each category by a user u is different. For example, let us suppose that category A contains 100 movies and category B contains 200 movies. According to the definition $POC(u, c)$, if user u has rated 90 movies from category A and 90 movies from category B, then user u has the same preference on both categories. However, this calculation cannot capture all the information as it can be seen that user u has rated 90% of movies from category A, and only 45% of the movies in category B.

Considering the percentage of items that are rated in each category by a user u, we define an *adjusted User Preference Over a Category* (aPOC) as follow:

$$aPOC(u, c) = \frac{POC(u, c) \cdot Pct_{I_u,c}}{\sum_{c'=1}^{C} POC(u, c') \cdot Pct_{I_u,c'}} \quad (1)$$

Where $Pct_{I_u,c}$ is the percentage of the items rated by user u in the category c. The definition implies that $\sum_{c=1}^{C} aPOC(u, c) = 1$.

Recall that our research problem is to recommend one or more *item sets* to *a group of users*. We define the following term, *package*, to constrain an item set.

Definition 2 *Package of items from different categories. An item package P consists of C items $\{i_1, i_2, \cdots, i_C\}$ where (i) $c(i_j) \neq c(i_k)$ if $j \neq k$, and (ii) $\cup_{j=1}^{C} c(i_j) = \mathcal{C}$.*

The first condition in the definition means that each item of the package belongs to a different category. The second condition means that the items in the package cover all the categories. For example, if we have two categories (hotel and restaurant), then a package should contain one hotel and one restaurant (instead of two hotels or two restaurants).

Let a temporary group of users which needs a recommendation be $G \subseteq \mathcal{U}$. For example, a group of friends looking for a set of movies or activities for vacation. Let $u \in G$, and a given package P. We propose two different models to measure users' preference over a package. These are based on different social theories, and were adapted because, as being temporary groups, there is not prior information about the interaction between their members.

1) The first approach to model user's preference over a package: Existing study [18] suggests that, in social activities, once a person makes a decision, her/his primary behavior becomes defending or justifying that decision. The first model ($M1$) is based on this theory.

In this model, the preference of $u \in G$ over P, denoted as $Pref(u, P)$, considers several intuitive factors. The first factor measures the degree that a user u likes the item i among all the items that this user rates. Let us denote this factor as $Prob(u, i)$. Let $r_{u,i}$ be the overall rating given by user u to the item i. $Prob(u, i)$ can be quantified using $\frac{r_{u,i}}{\sum_{k=1}^{I_u} r_{u,i_k}}$. It captures the ratio of the rating that the user u gives to the item i and the summation of the ratings from the user u. The second factor takes into consideration user preference over the categories of the items in the package. This factor is captured using $aPOC(u, c(i))$ for all the items in the package. The third factor captures the importance of item i to the user u compared with the importance of this item to other group members. Let us denote this factor as $Prob(u|G, i)$. Considering that $Prob(u, i)$ measures the degree that a user u likes an item i, this importance factor then can be defined as $\frac{Prob(u,i)}{\sum_{v \in G} Prob(v,i)}$.

Leveraging all these factors, we can define the preference of u over a package P.

$$Pref(u, P) = \prod_{j=1}^{C} Prob(u, i_j) \cdot aPOC(u, c(i_j)) \cdot Prob(u|G, i_j) \quad (2)$$

This definition implies that a user u prefers an item package P, if (i) each *item* in the package has high probability to be liked by the user, (ii) the package contains elements belonging to *categories* on which the user has a high preference, and (iii) the importance of each item to this user among all the group members.

2) The second approach to model user's preference over a package: Social influence and conformity theory [19], [20] suggests that it is common for people to change their own opinion due to input from other people.

The second model ($M2$) calculates $Pref(u, P)$ based on this theory and takes one more factor than the first model. This factor models the influence from other members of G to the user u. This influence factor is denoted as $Inf(u|G, i_j)$ and defined as $\sum_{v \in G} Prob(v, i_j)$, where $u \neq v$. Leveraging the influence factor, the second model M_2 defines

$Pref(u, P)$ as follows:

$$Pref(u, P) = \sum_{j=1}^{C} (Prob(u, i_j) + aPOC(u, c(i_j))) \cdot Inf(u|G, i_j) \quad (3)$$

B. Group preference over a package.

The group preference over a package is the aggregation of the preference of users in this group. In the following definitions (Defs. 3 and 4), we are given a group of users G, an item package P, and the preference $Pref(u, P)$ of each user $u \in G$.

Definition 3 *Group preference over a package. The group preference for a package P, denoted as $Pref(G, P)$ is defined as:*

$$Pref(G, P) =_{agg} \mathcal{G}_{m=1}^{|G|} Pref(u, P) \quad (4)$$

Where agg is a user defined aggregation function (e.g. sum, average, min, max).

We are interested in item packages which give a better balance between the preference of the users in the group. In order to quantify this, we define a new metric, *balance*.

Definition 4 *Balance of a package. The balance of a package P for all the users in the group G, denoted as $Balance(G, P)$, is defined as:*

$$Balance(G, P) = \frac{1}{|G|} \cdot \sum_{u \in G} \left| \frac{Pref(u, P)}{\sum_{v \in G} Pref(v, P)} - \frac{1}{|G|} \right| \quad (5)$$

A lower balance value is preferred. Here $\frac{1}{|G|}$ is the ideal balance for all the group members, meaning that all the users are equally satisfied with the package. For example, if a group has three users we would like to recommend a package that is equally liked by all users, meaning that each user's preference for that package is close to 33%.

Then, given a set of packages $\mathbb{P}_1, \mathbb{P}_2, \cdots, \mathbb{P}_p$ and a user group G, each package gets a group preference value $Pref(G, \mathbb{P}_p)$ indicating group G's preference over them.

C. Problem definition

Given the terminologies defined above, we define our problem as follows.

Definition 5 *Multi-criteria package recommendation for a group of users. Given a 3-way tensor with the ratings from all the users on all item features*

$$R_{UIF} = \begin{pmatrix} \vec{r}_{1,1} & \vec{r}_{1,2} & \cdots & \vec{r}_{1,I} \\ \vec{r}_{2,1} & \vec{r}_{2,2} & \cdots & \vec{r}_{2,I} \\ \vdots & \vdots & \ddots & \vdots \\ \vec{r}_{U,1} & \vec{r}_{U,2} & \cdots & \vec{r}_{U,I} \end{pmatrix}$$

, a group of users $G \subseteq \mathcal{U}$, and the definition of group preference over packages $Pref(G, \cdot)$, the **problem** *is to find a set of item packages $\mathcal{P} = \{\mathbb{P}_1, \mathbb{P}_2, \cdots, \mathbb{P}_K\}$, such that $\forall \mathbb{P}_k \in \mathcal{P}, \nexists \mathbb{P}_l | \mathbb{P}_l \notin \mathcal{P}$ s.t. $Balance(G, \mathbb{P}_l) \leq Balance(G, \mathbb{P}_k)$ (i.e., the returned set contains the top-K packages with the best balance for G).*

Algorithm 1: FindCC (R_{UIF}) //find co-clusters

Input : 3-way tensor R_{UIF}
Output: SIM_{UI}, CC_{UI}

1 $M_{UI} \leftarrow$ the overall rating of users on items (matrix corresponding to the "overall rating" feature in R_{UIF}
2 $M_{UF} \leftarrow$ averaging the I (item) dimension of R_{UIF}
3 $M_{IF} \leftarrow$ averaging the U (user) dimension of R_{UIF}
4 Generate co-clustering CC_{UF} from M_{UF} to extract groups of users sharing similar feature preference ;
5 Derive SIM_{UI} for each group of users in CC_{UF} and using items from M_{IF} which satisfy the users' feature preference;
6 Generate co-clustering CC_{UI} from M_{UI} to extract groups of users sharing similar overall ratings to items;
7 Return SIM_{UI}, CC_{UI}

IV. Proposed Approach

This section presents our approach of recommending packages with multi-criteria items to a group of users. Figure 1 shows the framework of our proposed approach, consisting of two major steps, *predicting missing ratings* and *recommending packages*. The details are presented in Sections IV-A and IV-B respectively.

A. Prediction of missing ratings

Users' preferences are reflected in their ratings. For the items that are not rated yet, we need to learn users' preferences through rating prediction. To predict the missing rating of a user u to an item i, we take a CF approach by utilizing the ratings from u's similar users and the ratings for the similar items of item i. The major challenge of this task is explained as the third challenge in Section I. Therefore, for a given user (item) we are interested in finding the niche of users (items) where all the members share something in common (e.g. similar preference for a set of features) [21].

We propose a co-clustering based approach to solve this problem. This approach finds similar users and similar items by using the different features. These similarities are used to predict the missing ratings.

1) Grouping of users and items: The first step is to find groups of similar users and groups of similar items. For this step we use Algorithm ***FindCC*** (Algorithm 1). It generates two intermediate results SIM_{UI} and CC_{UI}, which are used to calculate similarities among users in the next step.

The algorithm first constructs three matrices, M_{UI}, M_{UF}, and M_{IF}. The matrix M_{UI}, user-item matrix, consists of the overall ratings of each user to every item. The matrix M_{UF}, user-feature matrix, contains the aggregated importance of each item feature to every user by averaging all the item ratings from each user on every feature. The matrix M_{IF}, item-feature matrix, has the ratings of each feature for every

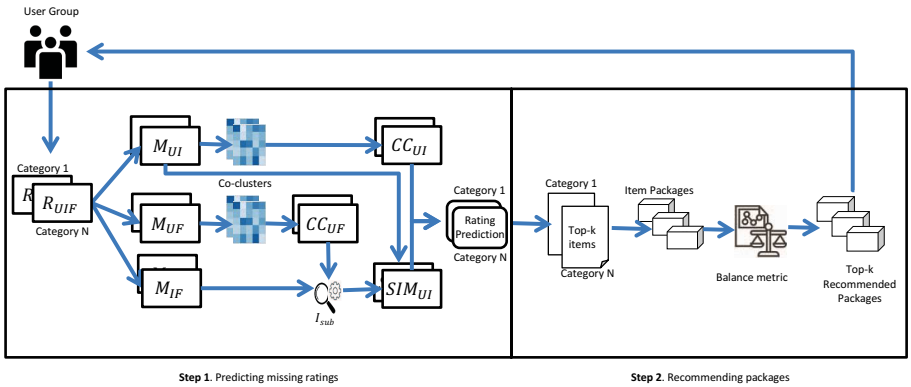


Figure 1: Proposed framework

item by averaging all user ratings on given items and item features. The next step *FindCC* is to derive SIM_{UI} from M_{UF}, M_{IF}, and M_{UI}. In particular, the algorithm calculates user-feature co-clusters CC_{UF} (Line 4). For each co-cluster, which consists of a group of users U_{clu} and a group of features F_{clu}, we calculate a feature vector by averaging all user's ratings for each feature. Using the feature vector for the co-cluster, we search M_{IF} to find the I_{sub} items that are closest to the co-cluster's feature vector on features F_{clu}.

We conduct the same procedure for all the co-clusters in CC_{UF} and get a $U \times I_{sub}$ matrix, SIM_{UI}. The last step of *FindCC* is to calculate CC_{UI}, which contains the user-item co-clusters, where each co-cluster groups users who have similar overall ratings over the corresponding items in the cluster. As an example, let the user cluster be U_{clu}= $\{u_1, u_2\}$, and the features cluster be $F_{clu} = \{f_1, f_2\}$. Having the co-cluster $\begin{pmatrix} r_{u_1,f_1} & r_{u_1,f_2} \\ r_{u_2,f_1} & r_{u_2,f_2} \end{pmatrix}$, this co-cluster's feature vector is $(\frac{r_{u_1,f_1}+r_{u_2,f_1}}{2}, \frac{r_{u_1,f_2}+r_{u_2,f_2}}{2})$.

Given $I_{sub} = 2$, we need to find two items that have features closest to this feature vector. Let the total number of features be 3, and the total number of items be 4, and M_{IF} be $\begin{pmatrix} r_{i_1,f_1} & r_{i_1,f_2} & r_{i_1,f_3} \\ r_{i_2,f_1} & r_{i_2,f_2} & r_{i_2,f_3} \\ r_{i_3,f_1} & r_{i_3,f_2} & r_{i_3,f_3} \\ r_{i_4,f_1} & r_{i_4,f_2} & r_{i_4,f_3} \end{pmatrix}$.

By comparing the co-cluster's feature vector with the feature vectors on f_1 and f_2 for all the items, $(r_{i_1,f_1}, r_{i_1,f_2}), \cdots, (r_{i_4,f_1}, r_{i_4,f_2})$, we can get the 2 most similar items. Suppose the two items are i_2 and i_3, then, the rows for u_1 and u_2 in SIM_{UI} contain i_2 and i_3. Running the algorithms on all the other user-feature co-clusters, we can fill in all the rows of SIM_{UI}.

The co-clusters are calculated using the spectral bicluster algorithm [22].

2) Similarity measurement: The rating prediction requires defining similarities among users and similarities between items. We use Adjusted Cosine Similarity (aCS), because it has shown to work better when users have different rating scales (i.e., some users usually give high ratings and others generally give lower ratings) [23]. aCS removes the effect of the different rating scales by subtracting the average ratings for all users from each user's ratings.

User similarity. To find the initial similarity between user u and user v we use the following equation:

$$aCS_{u,v} = \frac{\sum_{i \in I} (r_{u,i} - \overline{r}_u) \cdot (r_{v,i} - \overline{r}_v)}{\sqrt{\sum_{i \in I} (r_{u,i} - \overline{r}_u)^2} \cdot \sqrt{\sum_{i \in I} (r_{v,i} - \overline{r}_v)^2}} \quad (6)$$

where $\overline{r}_u$ and $\overline{r}_v$ are the averaged item ratings from user u and v respectively. Eq. (6) does not consider whether users u and v are from the same cluster or not. However, this simplifies the real situation. For example, given that users u_1 and v_1 co-occur in five co-clusters (out of ten), and users u_2 and v_2 co-occur in two of ten clusters, the similarity between u_1 and v_1 should be higher than that between u_2 and v_2.

We use the ratio of the number of co-clusters in which both users co-occur and the total number of co-clusters to adjust the user similarity. In the above example, aCS_{u_1,v_1} is adjusted by 0.5, and aCS_{u_2,v_2} is adjusted by 0.2. Formally, we define the similarity $sim(u, v)$ between user u and user v as follows.

$$sim_{u,v} = \frac{|CC_{uv}|}{|CC|} \cdot aCS_{u,v} \quad (7)$$

Here, $|CC|$ is the total number of co-clusters in CC_{UF} and CC_{UI}, and $|CC_{uv}|$ represents the number of co-clusters in which u and v co-occur (either in CC_{UF} or CC_{UI}).

Item similarity. In the same way, to find the initial similarity between item i and item j we use:

$$aCS_{i,j} = \frac{\sum_{u \in U} (r_{u,i} - \overline{r}_u) \cdot (r_{u,j} - \overline{r}_u)}{\sqrt{\sum_{u \in U} (r_{u,i} - \overline{r}_u)^2} \cdot \sqrt{\sum_{u \in U} (r_{u,j} - \overline{r}_u)^2}} \quad (8)$$

where $\overline{r}_u$ is the user's average ratings. Similar to the definition of user similarity, we argue that two items appearing in more co-clusters should be more similar than items that co-occur in less co-clusters. Utilizing this intuition, the

similarity between two items i and j are defined as follows.

$$sim_{i,j} = \frac{|CC_{ij}|}{|CC|} \cdot aCS_{i,j} \tag{9}$$

Here, $|CC|$ is the total number of co-clusters in SIM_{UI} and CC_{UI}. $|CC_{ij}|$ represents the number of co-clusters that two items i and j co-occur in SIM_{UI} and CC_{UI}.

3) Ratings prediction: To predict the missing rating of a user u for item i ($\hat{r}_{ui}$) in each co-cluster, we combine user similarity and item similarity (Eqs. (7) and (9)) as follows.

$$\begin{aligned}\hat{r}_{ui} = \alpha \times \Big[\overline{u} + \frac{\sum_{u,v \in CC} sim(u,v) \cdot (r_{vi} - \overline{v})}{\sum_{u,v \in CC} |sim(u,v)|}\Big] + \\ (1-\alpha) \times \Big[\overline{i} + \frac{\sum_{i,j \in CC} sim(i,j) \cdot (r_{uj} - \overline{j})}{\sum_{i,j \in CC} |sim(i,j)|}\Big]\end{aligned} \tag{10}$$

Here, α is a balancing parameter that can be adjusted to emphasize either the role of user similarity or the role of item similarity. r_{ui} represents the rating of a user u to an item i. For a user u, $\overline{u}$ is the averaged rating of the user u to all the items he has rated. For an item i, let U_i be the set of users who rate item i, then $\overline{i}$ is the averaged rating of an item i from all the users.

Finally, the predicted ratings in each co-cluster can be aggregated (i.e., sum, avg, max, etc.) to obtain a final overall rating. With the predicted missing ratings for the group users on all the items, we get a dense user-item matrix, whose rating values combine user's preference over different features. We use M'_{UI} to denote this matrix. For each category, we get such a matrix.

B. Item Package Recommendation

We design Algorithm 2 to recommend item packages to a group of users.

The algorithm works in several steps. (1) The first step is to form candidate packages (Line 1). Recall that a package cover all the categories, and that a package contains one item from each category. For each category, we form a set of unique items which is a union of the *top-Z* items for each user. Then, we form candidate item packages by combining the items from the item-sets for different categories. (2) The second step is to calculate the preference of every user for each of the categories (Line 3). For this calculation, we need to extract from M'_{UI} (a) I_{uc}, which is the number of items rated by the user u in the category c, (b) I_u, the total number of items rated by the user u, and (c) $Pct_{I_u,c}$, the percentage of the items rated by user u in the category c . (3) The third step calculates user preference over a package, $Pref(u, P)$. Two models in Eq. (2) and Eq. (3) can be used to calculate this. (4) The last step recommends a set of packages $\mathcal{P}$ to the temporary user group G. We argue that the best packages are the most balanced ones when all the group members are equally satisfied. The packages with the K lowest balance values are recommended to the group.

Algorithm 2: PackageRecommendation

Input : M'_{UI} (user-item rating matrix for each category), C (set of categories), G (group of users), K (the number of packages to recommend), Z (the number of items to select in each category)

Output: T (set of top-K packages)

1 Generate candidate packages using user-item matrix (M'_{UI}) with Z;
2 **foreach** $c \in C$ **do**
3 | Calculate $aPOC(u, c)$ for every $u \in G$ using Eq. (1);
4 **end**
5 **foreach** *package* P **do**
6 | Calculate $Pref(G, P)$ using Eq. (4);
7 | Calculate $Balance(G, P)$ using Eq. (5)
8 **end**
9 $T = argmin\{Balance(G, P)\}$
10 Return T;

Time: Step 1 uses $O(Z^C)$ time since there are C categories and each category contains Z items with high preference.

V. Experimental Results

A. Datasets.

Datasets with multi-criteria ratings and having items from different categories are rare to find [24]. Datasets used for group recommenders are generally derived from data used for recommender systems for individual users [25]. We use three real datasets. The first is extracted from the TripAdvisor[1] website. The second is the Yahoo!Movies[2] multi-criteria dataset. The last one is the BeerAdvocate dataset[3].

For *TripAdvisor*, we collect user reviews for hotels and restaurants from Los Angeles, California. Users can give restaurants an overall rating and a rating for the *service, value, food, and atmosphere*. In the same way, the users give an overall rating for each hotel and for the *service, cleanliness, value, sleep quality, location, and rooms*. For both categories, the ratings are in a scale from 1 to 5, being 1 the worst rate and 5 the best rate. We pick users who have rated at least 10 items in both categories, for a final dataset containing 283 users, 266 hotels, and 2,073 restaurants.

In the *Yahoo!Movies* dataset, the films may belong to one or more of 18 existing genres. We pick the first genre of those movies as their category. We select 3,144 users who have rated at least 10 movies. The whole dataset contains 11,915 movies. Users give each movie an overall score, and scores on the *story, acting, directing, and visuals*. We use the

[1]https://www.tripadvisor.com/
[2]https://webscope.sandbox.yahoo.com/
[3]http://wwww.beeradvocate.com/

range 1 to 5 for the ratings. For package recommendations, we consider each movie's genre as a different category. Note that people may not agree that a genre is strictly a category. We use this dataset for method comparison.

The *BeerAdvocate* reviews include ratings on four aspects (*appearance, aroma, palate, taste*) in addition to an overall rating for different types of beers. The ratings are in a scale from 1 to 5. In total there are 104 types of beers, we use these types as categories. The dataset contains 33,387 users and 66,051 beers, with a total of more than 1.5 million reviews.

To form item packages using TripAdvisor, we consider both categories (*restaurants* and *hotels*). For Yahoo!Movies, without loss of generalization, we create packages with the two (*action, comedy*) and three categories having most movies (*action, comedy*, and *drama*). For BeerAdvocate we use the top two (*American IPA*, *Imperial IPA*) and three categories (*American IPA*, *Imperial IPA* and *American Pale Ale*) with the most number of ratings to form packages. We select users who have rated at least 10 beers in each category.

B. Parameters

Table II lists all the parameters that are used in our models and approaches. On each test, we vary one parameter's values and keep the other parameters to their default values. Each test computes the top-10 recommended packages to a random group of users, to simulate temporary groups.

Table II: Parameters (default values are in bold)

Parameter	Values
Size of a user group	2,3,**4**,5,6
α	0.4,0.5,**0.6**,0.7,0.8
I_{sub} (for SIM_{UI})	1, 10, **20**, 30, 40
Size of co-cluster matrix	4x4, 6x4, 8x4, **10x4**, 12x4
Z (for the top Z items)	5, 10, 15, **20**, 25

C. Baseline methods.

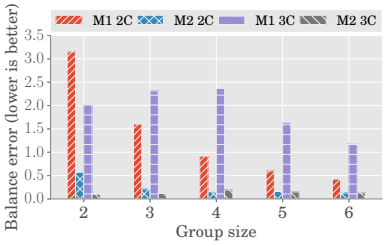


Figure 2: Comparing our models using Yahoo!Movies with two and three categories (2C: 2 categories, 3C: categories)

Theories [26] show that group happiness is either the averaged happiness (i.e., averaged ratings) of the members, or seeks for fairness by giving preference to people whose preferred item is not chosen in previous recommendations (i.e. items are ranked as if each member is choosing them in turn).

We create two baseline methods based on an average strategy [5] (denoted as *Average*) and a fairness strategy [27] (denoted as *Fairness*). We further implement the third baseline (denoted as GR) based on [3], which introduces a *Group Rating (GR)* model by leveraging item ratings, user impact in the group, package viability, and fairness to find packages of items. GR's two aspects (*user impact in the group* and *fairness*) are related to our problem definition.

D. Metrics.

Three evaluation metrics are used. The first two are **consensus** and **m-envy-freeness** [25], [28]. The *consensus* metric of a group of users G on one item i (Eq. (11)) is calculated using the pairwise distances of item preferences among all the group members. The consensus of G over a package P is the averaged consensus of all the items in P.

$$consensus(G,i) = 1 - \left(\frac{\sum_{u_j \in G \wedge u_k \in G \wedge j \neq k} |Pref(u_j,\{i\}) - Pref(u_k,\{i\})|}{|G||G-1|} \right) \quad (11)$$

$Pref(u,\{i\})$ is defined in Eq. (2) (for *M1* method) and Eq. (3) (for *M2* method) where $\{i\}$ represent a package with only one item i.

Let G_{ef} be the group of users having at least m items in the top $x\%$ item ratings (according to preferences). The *m-envy-freeness* metric is defined to be the proportion of G_{ef} in G (Eq. 12), where $|\,.\,|$ represent cardinality.

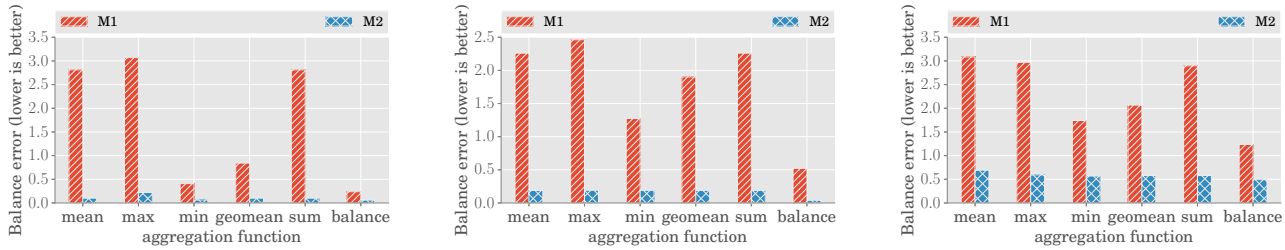
$$fairness_{m-envy}(G) = \frac{|G_{ef}|}{|G|} \quad (12)$$

The lower bound of $fairness_{m-envy}(G)$ captures the situation that the minimum number of users having their preferences in the top positions and is calculated as $\frac{\lceil x\% \times |G| \rceil}{|G|}$ where $\lceil \cdot \rceil$ represents the nearest integer function. The upper bound captures the situation that the top positions are spread among the maximum possible number of users and is calculated as $\frac{C \times \lceil x\% \times |G| \rceil}{|G|}$. This metric has the effect of averaging user's preferences because it evaluates the items based on a threshold ($x\%$) not based on their specific preference ranks. E.g., two items may have different preference ranks but are all within 25%, then there is no difference in counting users who like these two items. These two metrics capture to which extent group members agree on the recommended packages based on the user's preference for the items.

We also propose a new metric, **Balance Error** (*BE*), which is given by the following equation

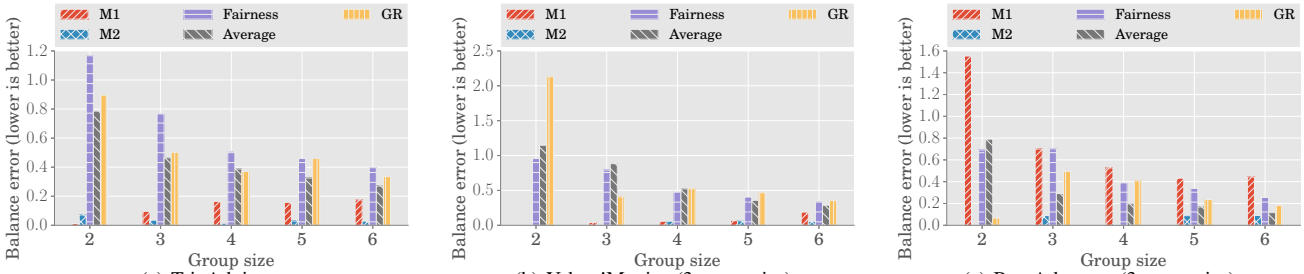
$$BE = \sum_{i=1}^{K} (Balance(G, P_i)) \quad (13)$$

This metric measures how far the recommended packages are from the ideal fairness when considering the users' preferences for both the items and categories.



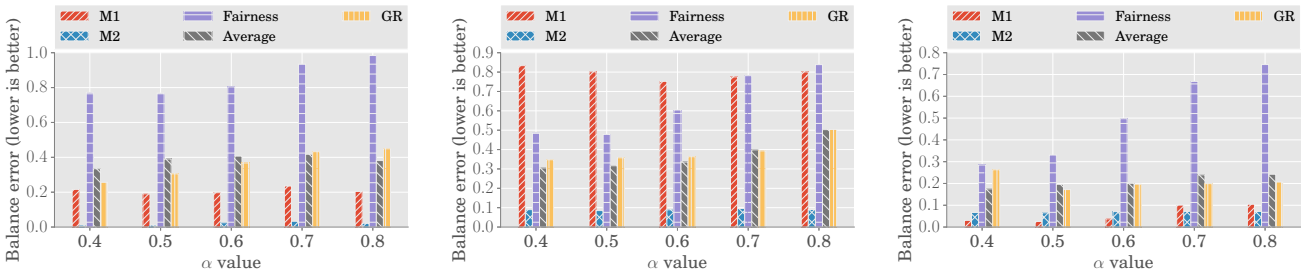
(a) TripAdvisor (b) Yahoo!Movies (3 categories) (c) BeerAdvocate (3 categories)

Figure 3: Balance error for different aggregation functions



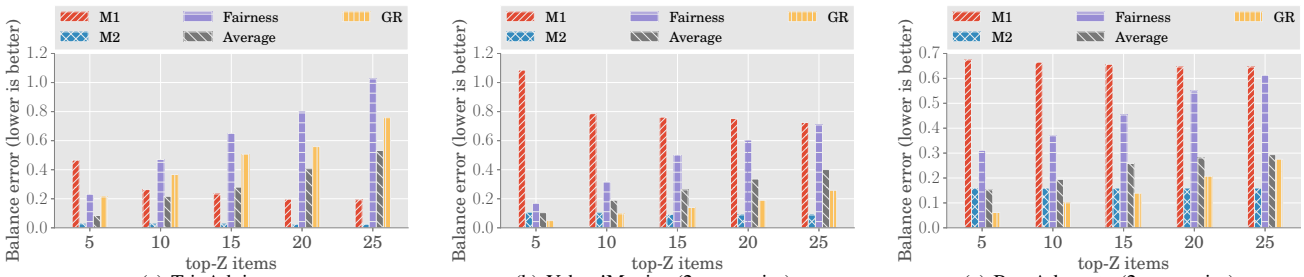
(a) TripAdvisor (b) Yahoo!Movies (3 categories) (c) BeerAdvocate (3 categories)

Figure 4: Balance error for different group sizes



(a) TripAdvisor (b) Yahoo!Movies (2 categories) (c) BeerAdvocate (2 categories)

Figure 5: Balance error for different α values



(a) TripAdvisor (b) Yahoo!Movies (2 categories) (c) BeerAdvocate (2 categories)

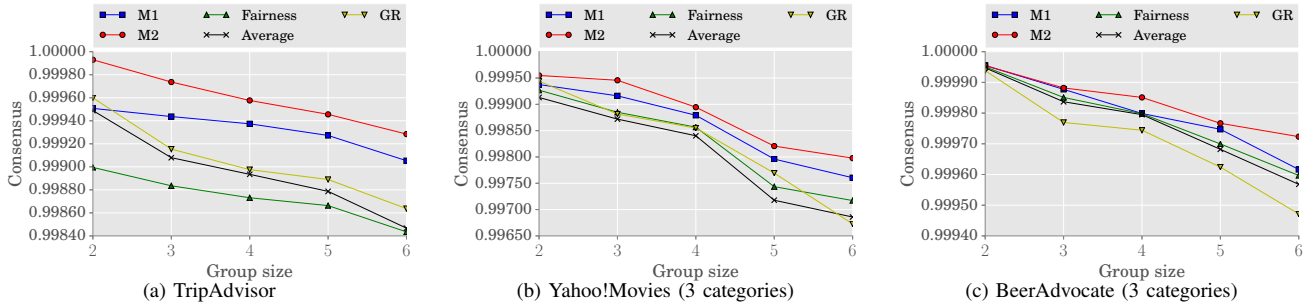
Figure 6: Balance error for different top-Z items

E. Results

We only present the figures for the most interesting results in following sections due to space limitations.

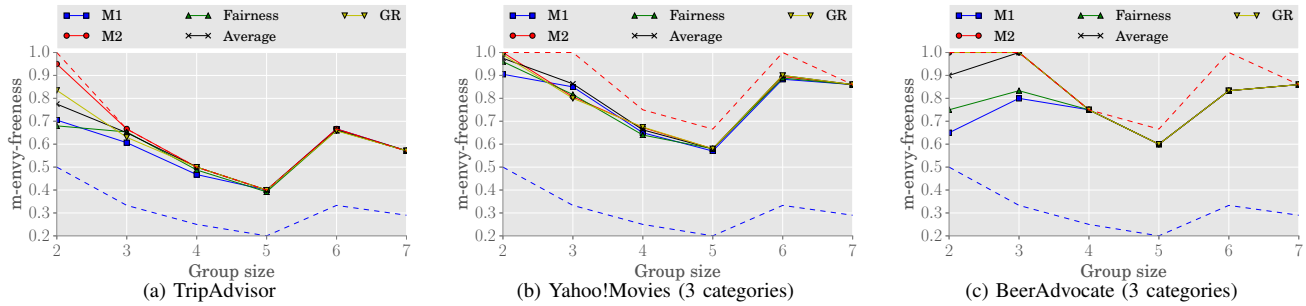
1) Comparing $M1$ and $M2$ models: Figure 2 shows the comparison of our two models when the item packages are formed with two and three categories. The results show that the $M2$ model, which considers the influence of group members, produces packages that are more balanced. This is consistent with the intuition behind the design.

2) Comparing the balance metric with other aggregation functions: Figure 3 shows that when utilizing the balance



(a) TripAdvisor (b) Yahoo!Movies (3 categories) (c) BeerAdvocate (3 categories)

Figure 7: Consensus for different group sizes



(a) TripAdvisor (b) Yahoo!Movies (3 categories) (c) BeerAdvocate (3 categories)

Figure 8: m-envy-freeness for different group sizes with $m = 1$(red dashed line: upper bound, blue dashed line: lower bound)

metric, instead of traditional aggregation functions (i.e. mean, min, max, sum, and geometric mean), our models can return packages with lower *BE*. Furthermore, these results show that $M2$ outperforms $M1$, producing packages that are more balanced for the group in all cases.

3) Sensitivity to group size: We create temporary user groups with 2 to 6 members. The users in each group are randomly chosen. Figure 4 shows again that $M2$ has lower *BE* than the other methods as the group size increases. This is mainly because the influence exerted by the group members makes them be more conformed when a larger number of candidate packages are generated. We also find that $M2$ error is higher than the baselines for the Yahoo!Movies dataset with two categories when only two users are in the group. We attribute this to the little influence exerted on each of the members.

4) Sensitivity to the size of co-cluster matrix: The calculation of the preference of the users for the items is achieved by co-clustering users, items, and features. The co-clustering technique we utilize needs to specify the size of co-cluster matrices (e.g. rows and columns). The column size is set to four because this is the maximum number of features for restaurants. Given the results, we conclude that the size of the co-cluster matrix is not a determining factor as the *BE* error remains stable in all cases.

5) Sensitivity to α value: Recall that α is a balancing parameter between the user similarity and the item similarity for the prediction of missing rates. Figure 5 shows the results for different α. The results show that varying this parameter has more influence on the baseline methods. This is because the baseline methods only consider user ratings, and not the adjustment from features, categories and influence.

6) Sensitivity to I_{sub}: For the missing ratings prediction, we need to calculate SIM_{UI} which contains the best I_{sub} items for the users in each user-feature co-cluster. When varying the values of I_{sub} we get a stable *BE*, indicating that this parameter does not have much influence on the results. As these results suggest we can use the smallest I_{sub} (which is 1) to improve the calculation efficiency.

7) Sensitivity to Z: We pick the top-Z items in each category for each group user. Figure 6 shows that different Zs do not affect the balance errors of our two models. Note $M2$ performs worse than some baselines for the Yahoo!Movies and BeerAdvocate datasets when using small Z values. This is attributed to that, when Z is small, there are fewer options to choose elements to form more balanced packages. Such behavior is also observed for the TripAdvisor dataset.

8) Consensus and m-envy-freeness metrics: Figure 7 plots the consensus results, which shows that our models ($M1$ and $M2$) outperform the baseline methods. This is because the users have more similar preferences for the package items using models $M1$ and $M2$. The absolute consensus values are high because preferences $Pref(u, \{i\})$ are small numbers.

For the experiments on evaluating the methods using *m-envy-freeness*, we used $m = 1$, and $x = 25$. We select

these values considering that the group sizes and the number of categories are small. Figure 8 shows the results and the upper bounds and lower bounds. The figure shows that our $M2$ model is slightly better than other models in most cases and is closer to the theoretical upper bound than the baseline models. This metric does not differentiate the different methods much because of its effect of averaging items' preferences. We note that the trend of this metric decreases before $G = 5$, bumps up at $G = 6$, and decreases again. This is because $x\% \times |G|$ for $G \leq 5$ is rounded to 1 while $x\% \times |G|$ for $G = 6, 7$ is rounded to 2.

We can see that our models have good performance for these two metrics, and in most cases $M2$ model performs better, which indicates that it can recommend packages that are more balanced for the group members.

VI. Conclusions

This paper studied the problem of recommending packages of multi-criteria items from different categories to a temporary group of users. We proposed a framework to solve this problem, which consists of two steps, estimating the missing ratings of users and recommending packages. In both steps, we need to calculate user's preference over packages.

We introduced two models to do the calculation, where the second model considers the influence of group members. Considering that the ratings for items are on multiple criteria, it is important to leverage the similarity of each criterion. To address this issue, we co-clustered users and items, items and features, and users and features. Utilizing the co-clusters, we calculated user-item missing ratings.

Finally, we proposed a balance metric to evaluate the goodness of the recommended packages. Our extensive experiments on three real datasets show that our second model outperforms the first model and the baselines for different parameter settings and different metrics.

Acknowledgment

This work has been supported by the National Council of Science and Technology of Mexico (CONACYT) #602434/440684, and National Science Foundation of USA (NSF) #1633330, #1345232, and #1757207.

References

[1] N. Manouselis and C. Costopoulou, "Analysis and classification of multi-criteria recommender systems," *World Wide Web*, vol. 10, pp. 415–441, 2007.

[2] A. Mikeli, D. Sotiros, D. Apostolou, and D. Despotis, "A multi-criteria recommender system incorporating intensity of preferences," in *Information, Intelligence, Systems and Applications (IISA), 2013 Fourth International Conference on*. IEEE, 2013, pp. 1–6.

[3] S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Recommending packages to groups," in *ICDE*, 2016, pp. 449–458.

[4] K. Li, W. Lu, S. Bhagat, L. V. Lakshmanan, and C. Yu, "On social event organization," in *SIGKDD*. ACM, 2014, pp. 1206–1215.

[5] H. Mengash and A. Brodsky, "Tailoring group package recommendations to large heterogeneous groups based on multi-criteria optimization," in *System Sciences (HICSS), 2016 49th Hawaii International Conference on*. IEEE, 2016, pp. 1537–1546.

[6] H. Mengash and A. Brodsky, "Dg-gpr: A decision-guided group package recommender with hybrid condorcet-instant runoff voting." *DSS*, vol. 2, pp. 317–328, 2014.

[7] H. Mengash and A. Brodsky, "Gcar: A group composite alternatives recommender based on multi-criteria optimization and voting," in *System Sciences (HICSS), 2014 47th Hawaii International Conference on*. IEEE, 2014, pp. 1113–1121.

[8] S. Board, "Preferences and utility," 2009.

[9] A. Blum, J. Jackson, T. Sandholm, and M. Zinkevich, "Preference elicitation and query learning," *Journal of Machine Learning Research*, vol. 5, pp. 649–667, 2004.

[10] B. Marlin, R. S. Zemel, S. Roweis, and M. Slaney, "Collaborative filtering and the missing at random assumption," *arXiv preprint arXiv:1206.5267*, 2012.

[11] A. Pujahari and V. Padmanabhan, "Group recommender systems: Combining user-user and item-item collaborative filtering techniques," in *Information Technology (ICIT), 2015 Intl. Conf. on*. IEEE, 2015, pp. 148–152.

[12] S. Feng, J. Cao, J. Wang, and J. He, "Group recommendations based on comprehensive latent relationship discovery," in *Web Services (ICWS), 2016 IEEE International Conference on*. IEEE, 2016, pp. 9–16.

[13] Q. Zhu, M. Zhou, J. Liang, T. Yan, and S. Wang, "Efficient promotion algorithm by exploring group preference in recommendation," in *2016 IEEE International Conference on Web Services (ICWS)*. IEEE, 2016, pp. 268–275.

[14] S. A. P. Parambath, N. Vijayakumar, and S. Chawla, "Saga: A submodular greedy algorithm for group recommendation," *arXiv preprint arXiv:1712.09123*, 2017.

[15] G. Adomavicius and Y. Kwon, "New recommendation techniques for multicriteria rating systems," *IEEE Intelligent Systems*, vol. 22, 2007.

[16] L. Liu, N. Mehandjiev, and D.-L. Xu, "Multi-criteria service recommendation based on user criteria preferences," in *Proc. of the fifth ACM conference on Recommender systems*. ACM, 2011, pp. 77–84.

[17] D. Jannach, F. Gedikli, Z. Karakaya, O. Juwig *et al.*, *Recommending hotels based on multi-dimensional customer ratings*. na, 2012.

[18] R. S. Nickerson, "Confirmation bias: A ubiquitous phenomenon in many guises." *Review of general psychology*, vol. 2, p. 175, 1998.

[19] R. B. Cialdini and N. J. Goldstein, "Social influence: Compliance and conformity," *Annu. Rev. Psychol.*, vol. 55, pp. 591–621, 2004.

[20] M. Kompan and M. Bielikova, "Group recommendations: survey and perspectives," *Computing and Informatics*, vol. 33, pp. 446–476, 2014.

[21] R. Burke, "Hybrid recommender systems: Survey and experiments," *User modeling and user-adapted interaction*, vol. 12, pp. 331–370, 2002.

[22] Y. Kluger, R. Basri, J. T. Chang, and M. Gerstein, "Spectral biclustering of microarray data: coclustering genes and conditions," *Genome research*, vol. 13, pp. 703–716, 2003.

[23] J. Wang, A. P. De Vries, and M. J. Reinders, "Unifying user-based and item-based collaborative filtering approaches by similarity fusion," in *ACM SIGIR*. ACM, 2006, pp. 501–508.

[24] G. Adomavicius, N. Manouselis, and Y. Kwon, "Multi-criteria recommender systems." *Recommender systems handbook*, vol. 1, pp. 769–803, 2011.

[25] A. Felfernig, L. Boratto, M. Stettinger, and M. Tkalčič, "Evaluating group recommender systems," in *Group Recommender Systems*. Springer, 2018, pp. 59–71.

[26] M. O'connor, D. Cosley, J. A. Konstan, and J. Riedl, "Polylens: a recommender system for groups of users," in *ECSCW 2001*. Springer, 2002, pp. 199–218.

[27] J. Masthoff, "Group recommender systems: Combining individual models," in *Recommender systems handbook*. Springer, 2011, pp. 677–702.

[28] D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, and P. Tsaparas, "Fairness in package-to-group recommendations," in *Proceedings of the 26th International Conference on World Wide Web*. International World Wide Web Conferences Steering Committee, 2017, pp. 371–379.