SPRINGER BRIEFS IN
ELECTRICAL AND COMPUTER ENGINEERING

Alexander Felfernig · Ludovico Boratto
Martin Stettinger · Marko Tkalčič

Group Recommender Systems

An Introduction

Springer

SpringerBriefs in Electrical and Computer Engineering

More information about this series at http://www.springer.com/series/10059

Alexander Felfernig • Ludovico Boratto
Martin Stettinger • Marko Tkalčič

Group Recommender Systems

An Introduction

Springer

Alexander Felfernig
Institute for Software Technology
Graz University of Technology
Graz, Austria

Ludovico Boratto
EURECAT
Centre Tecnológico de Catalunya
Barcelona, Spain

Martin Stettinger
Institute for Software Technology
Graz University of Technology
Graz, Austria

Marko Tkalčič
Faculty of Computer Science
Free University of Bozen-Bolzano
Bolzano, Italy

ISSN 2191-8112 ISSN 2191-8120 (electronic)
SpringerBriefs in Electrical and Computer Engineering
ISBN 978-3-319-75066-8 ISBN 978-3-319-75067-5 (eBook)
https://doi.org/10.1007/978-3-319-75067-5

Library of Congress Control Number: 2018930531

Printed on acid-free paper

This Springer imprint is published by the registered company Springer International Publishing AG part of Springer Nature.
The registered company address is: Gewerbestrasse 11, 6330 Cham, Switzerland

Preface

Recommender systems have become a fundamental means for providing personalized guidance to users in their searches for interesting or useful objects such as movies, songs, restaurants, software requirements, and digital cameras. Although most existing recommender systems support single users, there are many scenarios where items are used by groups. The increased interest in recommendation technologies for groups motivated us to write this book on *Group Recommender Systems*. The overall purpose of this book is to provide an easy to understand introduction to the field of group recommender systems. It is intended for persons new to the field, and also as reference material for researchers and practitioners that provides an overview of the existing state of the art and issues for future work. Included are contributions related to algorithms, user interfaces, psychological issues, and research challenges. The book entails an introductory presentation of different group recommendation algorithms. Beyond algorithms, it deals with additional relevant aspects such as group recommender user interfaces, evaluation techniques, approaches to the handling of preferences, different ways to include explanations into a group recommendation process, and psychological factors that have to be taken into account when building group recommender systems. The book also provides an overview of group recommendation scenarios that go beyond the basic ranking of alternatives. Related examples include group-based configuration, recommendation of sequences to groups, resource balancing for groups, and release planning for groups.

Graz, Austria — Alexander Felfernig
Barcelona, Spain — Ludovico Boratto
Graz, Austria — Martin Stettinger
Bolzano, Italy — Marko Tkalčič
Jan 2018

Acknowledgements

We want to thank the co-authors of this book for their contributions which we hope will help to advance the state of the art in group recommender systems. Furthermore, we want to express our thanks to *Nava Tintarev*, *Müslüm Atas*, *Gerhard Leitner*, *Andreas Falkner*, *Thi Ngoc Trang Tran*, *John Brown*, and *Ralph Samer* who helped with their feedback to improve the overall quality of individual chapters. Finally, a special thanks goes to *Anthony Jameson* who provided insightful feedback and comments which helped to improve the overall quality of the book. Parts of the work presented in this book have been conducted within the scope of the Horizon 2020 project OPENREQ (Intelligent Recommendation and Decision Technologies for Community-based Requirements Engineering) and the WEWANT project (Enabling Technologies for Group-based Configuration) funded by the Austrian Research Promotion Agency.

Contents

Contributors

Müslüm Atas, Graz University of Technology, Austria
Ludovico Boratto, EURECAT - Centro Tecnológico de Catalunya, Spain
Amra Delić, Vienna University of Technology, Austria
Alexander Felfernig, Graz University of Technology, Austria
Denis Helic, Graz University of Technology, Austria
Gerhard Leitner, Alpen-Adria University Klagenfurt, Austria
Stefan Reiterer, SelectionArts, Austria
Alan Said, University of Skövde, Sweden
Ralph Samer, Graz University of Technology, Austria
Martin Stettinger, Graz University of Technology, Austria
Nava Tintarev, Delft University of Technology, The Netherlands
Marko Tkalčič, Free University of Bozen-Bolzano, Italy
Thi Ngoc Trang Tran, Graz University of Technology, Austria
Christoph Trattner, University of Bergen, Norway
Martijn Willemsen, Eindhoven University of Technology, The Netherlands

Part I
Group Recommendation Techniques

Part I of this book focuses on different *recommendation techniques for groups*. In Chap. 1, recommender systems are introduced as a specific type of decision support system. In this context, the basic approaches of *collaborative filtering*, *content-based filtering*, *constraint-based*, *critiquing-based*, and *hybrid recommendation* are introduced with a working example from the domain of travel. In Chap. 2, we introduce different concepts of group recommender systems and show how such systems can be built on the basis of the recommendation approaches introduced in Chap. 1. Finally, in Chap. 3 we discuss techniques that can be used to evaluate group recommender systems.

Chapter 1
Decision Tasks and Basic Algorithms

Alexander Felfernig, Müslüm Atas, and Martin Stettinger

Abstract Recommender systems are *decision support systems* helping users to identify one or more items (solutions) that fit their wishes and needs. The most frequent application of recommender systems nowadays is to propose items to *individual users*. However, there are many scenarios where a *group of users* should receive a recommendation. For example, think of a group decision regarding the *next holiday destination* or a group decision regarding a *restaurant to visit for a joint dinner*. The goal of this book is to provide an introduction to *group recommender systems*, i.e., recommender systems that determine recommendations for groups. In this chapter, we provide an introduction to basic types of recommendation algorithms for individual users and characterize related decision tasks. This introduction serves as a basis for the introduction of group recommendation algorithms in Chap. 2.

1.1 Introduction

A recommender system is *a specific type of advice-giving or decision support system that guides users in a personalized way to interesting or useful objects in a large space of possible options or that produces such objects as output* [14, 17, 23, 29, 43, 66, 76, 81, 82]. A decision problem/task emerges if a person or a group of persons have an idea about a desired state [33]. If there are different options to achieve the desired state, the *decision task* to be solved is to identify items or actions that help to approach the target state in a suitable fashion. Recommender systems can provide help in such a context by trying to find the suitable items or actions that help to best reach the envisioned target. Arriving at a choice can be seen as the result of a collaboration between the user and the recommender system. Recommender systems support "good" choices within reasonable time spans including corresponding justifications provided in terms of explanations [42, 80].

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_1

There are different ways in which a recommender can support users in decision making processes. First, it can act as a *supporter* to figure out *candidate items*, i.e., a large number of alternatives is reduced to a so-called *consideration set*—selecting the favorite option is left to the user. Second, the recommender can help the user select from the items in the consideration set, for example, by representing them in convenient ways and providing explanations of why they have been recommended. Only in "extreme" cases is the *decision making authority taken over* by the recommender itself. Examples include music recommendation in a fitness studio, the recommendation of information units on a public display, and the automated adaptation of parameter settings such as light intensity in a smarthome [50].

An example of a single-user recommendation scenario is the following: when navigating an online sales platform with the goal to find a book related to the topic of, for example, *deep learning*, a recommender system will identify related books and propose these to the user of the online platform. In this scenario, the envisioned target state is to find a suitable book on the mentioned topic and the options to achieve this state are the different existing books on the topic of *deep learning*. Finding a book in an online sales platform is typically a *single user decision scenario*. However, there are also scenarios where a group of users has to make a decision. In this context, a recommender system must take into account the potentially conflicting preferences of different group members. Such a situation makes the recommendation task different and often more challenging.

The main focus of this book is *recommendation techniques that provide help in scenarios where a group of users is engaged*. In many scenarios, the presentation of recommendations to groups is a more natural approach than trying to address individual users [40, 56, 58]. For example, music recommendations in fitness studios have to take into account the preferences of all individuals currently present in the studio [59]. Stakeholders in a software project have to establish agreement regarding the requirements/features that have to be developed within the scope of the next release [64]. Personnel decisions are often taken in groups, i.e., a group has to decide which job applicant will be hired [78]. Groups of friends have to decide about the hotel for the next summer holidays or a skiing resort for the next winter holidays [39, 61]. A public display should be personalized in order to be able to display information to persons currently in the surrounding [40]. Finally, travel groups should receive a personalized museum guidance in such a way that the personal preferences of group members are fulfilled [28, 47].

1.2 Characteristics of Decision Tasks

Decision tasks differ with regard to various aspects [69]. In the following, we introduce *basic dimensions of decision tasks* (see [33]) which will help to better understand which decisions are supported by group recommenders (see Table 1.1).[1]

[1] As mentioned in [33], this characterization of decision tasks is not complete but a good basis for discussing properties relevant for group recommenders.

Table 1.1 Characteristics of decision tasks [33]

Dimension	Characteristic
Complexity	low .. high
Structuredness	low .. high
Decision type	choice .. design
Sentiment	opportunity .. threat
Dependence	yes .. no
Level	original .. meta
Actor	person .. group

Complexity We interpret complexity of decision tasks in terms of the *number of decision dimensions* and the *degree of item involvement* [33, 38]. Depending on the underlying item domain, humans will invest more or less time to come to a decision, i.e., to achieve an acceptable trade-off between decision effort and accuracy [67]. Items with higher related decision efforts are often denoted as *high-involvement items* whereas items with less related decision efforts are denoted as *low-involvement items* [70]. Suboptimal decisions have a much higher negative impact in the context of high-involvement items. For example, when purchasing an *apartment*, a suboptimal decision manifests in search efforts for a new apartment, unnecessary payments, relocation costs, and additional time efforts [27]. In contrast, risks related to the purchasing of a low-quality book are negligible—in the worst case, the user will provide negative feedback on the book and try to find other alternatives that better fit his/her wishes and needs. When purchasing a book, the number of decision dimensions is low—examples of dimensions are *price* and *quality*. The number of decision dimensions of *apartments* is much higher (e.g., price, quality of public transport, neighborhood, schools in the neighborhood, etc.). Aspects that further increase decision complexity especially in group decision scenarios are, for example, *contradicting preferences of group members*, *personal relationships*, *personality factors*, and *emotion-related aspects* (see Chap. 9).

Structuredness We interpret structuredness of decision tasks as the degree to which underlying processes and decision policies are defined. Decision tasks are often characterized by undefined processes and related decision policies are not predefined but developed and adapted in the course of the decision process. If a group of users has to decide on a holiday destination for the next summer, a recommender system can propose alternative destinations but it is unclear which of the alternatives will be chosen by the group. The final decision is something that has to be made by group members (or an individual user) and is in many cases not handled by the underlying decision support environment. There are exceptions to the rule, for example, music recommendations in fitness centers and information units shown on public displays. Specific decision types follow a *formalized process*. For example, electoral systems are defined by precise rules that determine how elections and referendums have to be conducted (the process) and how the results are determined (decision making).

Decision Type Decision tasks are often defined on the basis of known alternatives or parameters out of which one or more alternatives (values) should be selected. If alternatives (parameters) are predefined, the underlying decision task can be regarded as a basic *choice task* [42]. Choice problems are considered as central application area for recommender systems [42]. The other extreme are so-called *design tasks*, where alternatives are not predefined but created throughout a decision process. Design tasks are often related to creative acts where persons develop ideas and solutions. In "pure" design scenarios, the application of recommendation technologies is not widespread [75]. However, there are many scenarios located in-between basic choice tasks and "pure" design tasks. For example, knowledge-based configuration [25] can be considered as a simpler type of design task where a solution is identified (configured) out of a set of pre-defined component types. In this context, the alternatives (parameter values) are known beforehand; due to the large option space, not all potential alternatives can be enumerated for performance reasons—billions of alternatives would have to be managed and the corresponding recommendation algorithms would become inefficient [13] (see Chap. 7).

Sentiment Decision support with included recommendation support is very often *opportunity-related*, i.e., the goal is related to an opportunity and the best solution to achieve a goal should be identified. Examples thereof are purchasing a book to better understand a certain topic or choosing a holiday destination to spend unforgettable days somewhere abroad. A similar argument holds for item domains such as songs, digital equipment, food, and financial services. Certainly, decision problems also exist in contexts where alternative outcomes can be considered as negative ones. For example, choosing between alternative options to liquidate a company—in this scenario, every outcome can be considered as a negative one (the company gets liquidated). However, recommender systems can help to minimize damage, for example, on the basis of a structured utility analysis [79].

Dependence We regard decision tasks as dependent if the outcome of a decision has an impact on another decision. For example, the purchase of a movie typically does not require a follow-up decision regarding the purchase of the next movie or different item. Dependent decision tasks occur when one decision at an earlier point of time leads to follow-up decision tasks at a later point of time. An example of such a decision task is *requirements release planning* where for each software requirement it has to be decided when the requirement should be implemented [64]. Consequently, decisions taken in early phases of a software project can have an impact on or trigger decisions later in the project. For example, a decision that a requirement should be implemented could trigger decisions regarding additional resources in order to be able to provide the promised software functionalities in time.

Level We can differentiate between *original decisions* operating on the object level and decisions on the *meta-level* [33]. The first type is omni-present in many recommendation-supported scenarios—the underlying task is to identify and choose items of relevance. In contrast, meta-decisions are decisions about the qualities of

a decision process and the way decisions are taken. For example, a group could decide to use *majority voting* when it comes to the election of the next chairman. A meta-decision in this context is to decide about the election formalism—related alternatives can be, for example, *relative majority and a single-shot election* or *absolute majority in a potentially multi-level election process* [51]. In many decision scenarios—especially in the context of group decision making—recommendations have an advisory function but are not considered imperative.

Actor Many recommendation approaches support individual decision making where recommendation algorithms are focusing on determining recommendations for individual users. The focus of this book are recommender systems that support *decision making for groups of users*. The following types of groups are introduced in [7]: (1) *established groups* with shared and long-term common interests (e.g., *conference committees* taking decisions about conference venues or *families* taking a decision about purchasing a house), (2) *occasional groups* with a common aim in a particular moment (e.g., a group of persons jointly participating in a museum tour), (3) *random groups* (e.g., persons in a fitness center or persons around a public display), and (4) *automatically identified groups* where individuals with similar preferences have to be grouped (e.g., distribution of seminar papers to students and distribution of conference papers to reviewers).

1.3 Recommendation Algorithms for Individual Users

Recommender systems [43, 58] propose items of potential interest to an individual user or a group of users.[2] They are applied in item domains such as books [52], web sites [68], financial services [16], and software artifacts [20, 24]. In the following, we introduce *collaborative filtering* [31, 48], *content-based filtering* [68], *constraint-based* [8, 14], *critiquing-based* [10, 11], and *hybrid recommendation* [9] which are basic recommendation approaches. The items in Table 1.2 (travel destinations) serve as examples to demonstrate how basic recommendation algorithms operate. In Chap. 2, we show how these approaches can be integrated into corresponding group recommendation scenarios.

Collaborative Filtering

Collaborative filtering is based on the idea of word-of-mouth promotion where opinions of relatives and friends play a major role when taking a decision [12, 48, 52]. In online scenarios, family members and friends are replaced by *nearest neighbors* who

[2] Parts of this section are based on a discussion of recommendation technologies given in [24].

Table 1.2 Example set of travel destinations

Travel destination (item)	Name	Beach	City tours	Nature	Entertainment
t_1	Vienna		x		x
t_2	Yellowstone			x	
t_3	New York	x	x		x
t_4	Blue Mountains			x	
t_5	London		x		x
t_6	Beijing		x		x
t_7	Cape Town	x	x	x	x
t_8	Yosemity			x	
t_9	Paris		x		x
t_{10}	Pittsburgh		x		x

These items will be used across the following sections for demonstration purposes. Each travel destination is described by a set of meta-characteristics (categories), for example, travel destination *Yellowstone* is famous for its experience of *nature*

are users with preferences similar to the ones of the current user. In collaborative filtering, a *user* × *item* rating specifies to which extent a user likes an item. *Rating predictions* are determined by a recommender algorithm to estimate the extent a user will like an item he/she did not consume/evaluate up to now. A collaborative filtering recommender first determines k nearest neighbors ($k - NN$).[3] The preferences of nearest neighbors are used to extrapolate future item ratings of the current user. A *user* × *item* rating matrix that will be used in the following for explaining collaborative filtering is shown in Table 1.3. In this example, all users u_i visited travel destinations and provided a corresponding rating. In collaborative filtering, *user* × *item* ratings serve as input for the recommender.

Collaborative filtering identifies the k-nearest neighbors of the current user u_a (see Formula 1.1)[4] and—based on the nearest neighbors—calculates a prediction for the current user's rating. When applying Formula 1.1, user u_2 is identified as the nearest neighbor of user u_a (see also Table 1.3). The similarity between u_a and another user u_x can be determined, for example, using the Pearson correlation coefficient [43] (see Formula 1.1) where TD_c is the set of items that have been rated by both users (u_a and u_x), r_{x,t_i} is the rating of user x for item t_i, and $\overline{r_x}$ is the average rating of user x. Similarity values resulting from Formula 1.1 can take values on a scale of $[-1.. + 1]$. Sometimes, "neighbor" users with low or negative correlations with the current user are filtered out [1].

[3] We focus on *user-based collaborative filtering* which is a memory-based approach that—in contrast to model-based ones—operates on an uncompressed version of a user/item matrix [6, 12].

[4] We assume k = 2 in our example.

Table 1.3 Example of collaborative filtering rating matrix: travel destinations (items) t_i and ratings (we assume a rating scale of 1..5)

Item	Name	u_1	u_2	u_3	u_4	u_a
t_1	Vienna	5.0			4.0	
t_2	Yellowstone	4.0				
t_3	New York		3.0	4.0	3.0	
t_4	Blue Mountains		5.0	5.0		4.0
t_5	London			3.0		
t_6	Beijing		4.5	4.0		4.0
t_7	Cape Town	4.0				
t_8	Yosemity		2.0			
t_9	Paris				3.0	
t_{10}	Pittsburgh				5.0	3.0
	Average rating ($\overline{r_x}$)	4.33	3.625	4.0	3.75	3.67

Table 1.4 Similarity between user u_a and the users $u_j \neq u_a$ determined based on Formula 1.1

	u_1	u_2	u_3	u_4
u_a	–	0.97	0.70	–

If the number of commonly rated items is below 2, no similarity between the two users is calculated

$$similarity(u_a, u_x) = \frac{\sum_{t_i \in TD_c}(r_{a,t_i} - \overline{r_a}) \times (r_{x,t_i} - \overline{r_x})}{\sqrt{\sum_{t_i \in TD_c}(r_{a,t_i} - \overline{r_a})^2} \times \sqrt{\sum_{t_i \in TD_c}(r_{x,t_i} - \overline{r_x})^2}} \tag{1.1}$$

The similarity values for u_a calculated based on Formula 1.1 are shown in Table 1.4. For the purpose of our example, we assume the existence of at least two items per user pair (u_i, u_j) (i ≠ j) in order to be able to determine a similarity. This criterion holds for users u_2 and u_3.

A challenge when determining the similarity between users is the *sparsity* of the rating matrix. Users typically provide ratings for only a very small subset of the offered items. For example, given a large movie dataset that contains thousands of entries, a user will typically be able to rate only a few dozen. One approach to this problem is to take into account the number of commonly rated items as a *correlation significance* [37], i.e., the higher the number of commonly rated items, the higher is the significance of the corresponding correlation. For further information regarding the handling of sparsity, we refer to [37, 43].

The information about the set of users with a rating behavior similar to that of the current user (nearest neighbors NN) is the basis for predicting the rating of user u_a for an *item t* that has so far not been rated by u_a (see Formula 1.2).

$$prediction(u_a, t) = \hat{r}(u_a, t) = \overline{r_a} + \frac{\sum_{u_j \in NN} similarity(u_a, u_j) \times (r_{j,t} - \overline{r_j})}{\sum_{u_j \in NN} similarity(u_a, u_j)} \tag{1.2}$$

Table 1.5 Collaborative filtering based recommendations (predictions) for items that have not been rated by user u_a up to now

	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
u_2	–	–	3.0	5.0	–	4.5	–	2.0	–	–
u_3	–	–	4.0	5.0	3.0	4.0	–	–	–	–
u_a	–	–	–	4.0	–	4.0	–		–	3.0
Prediction for u_a	–	–	3.30 $\surd$	–	2.66	–	–	2.04	–	–

Based on the ratings of the nearest neighbors of u_a, we are able to determine a prediction for u_a (see Table 1.5). The nearest neighbors of u_a are assumed to be u_2 and u_3 (see Table 1.4). The travel destinations rated by the nearest neighbors but not rated by u_a are t_3, t_5, and t_8. Due to the determined predictions (Formula 1.2), item t_3 would be ranked higher than the items t_5 and t_8 in a recommendation list. For a discussion of advanced collaborative recommendation approaches, we refer the reader to [49, 74].

Content-Based Filtering

This approach is based on the assumption of monotonic personal interests [68]. For example, users interested in *political news* are typically not changing their interest profile from one day to another. On the contrary, they will also be interested in the topic in the (near) future. In online scenarios, content-based recommenders are applied, for example, when it comes to the recommendation of websites [68].

Content-based filtering is based on (a) a set of users and (b) a set of categories (or keywords) that have been assigned to (or extracted from) the set of items. It compares the content of already consumed items with new items, i.e., it finds items that are similar to those already consumed (positively rated) by the user (u_a). The basis for determining such a similarity are *keywords* extracted from the item content descriptions (e.g., keywords extracted from news articles) or *categories* if items have been annotated with the relevant meta-information. Readers interested in the principles of keyword extraction are referred to [43]. In this book, we focus on content-based recommendation which exploits item categories (see Table 1.2).

Content-based filtering will now be explained using Tables 1.2, 1.6, and 1.7. Table 1.2 provides an overview of the relevant items and the assignments of items to categories. Table 1.6 provides information on which categories are of relevance for our example users. For example, user u_1 is interested in items related to all categories. Since user u_a rated the items t_4, t_6, and t_{10} (see Table 1.3), we can infer that u_a is interested in the categories *City Tours*, *Nature*, and *Entertainment* (see Table 1.6) where items related to the categories *City Tours* and *Entertainment* have been evaluated twice and items related to *Nature* have been evaluated once by u_a.

If we are interested in an item recommendation for the user u_a, we have to search for those items which are most similar to the items that have already been consumed

Table 1.6 Example category interests of users: users u_1 – u_3 visited travel destinations that cover all available categories

Category	u_1	u_2	u_3	u_4	u_a
Beach	x	x	x	x	–
City tours	x	x	x	x	x
Nature	x	x	x	–	x
Entertainment	x	x	x	x	x

If a user accessed an item at least once with a rating ≥ 3 (see Table 1.3), it is inferred that the user is interested in this item

Table 1.7 Example of content-based filtering: *user* u_a has already consumed the items t_4, t_6, and t_{10} (see Table 1.3)

Item	Rating (u_a)	Name	Beach	City tours	Nature	Entertainment	Similarity(u_a, t_i)
t_1		Vienna	–	x	–	x	$\frac{4}{5}$
t_2		Yellowstone	–	–	x	–	$\frac{1}{2}$
t_3		New York	x	x	–	x	$\frac{2}{3}$
t_4	4.0	Blue Mountains	–	–	x	–	–
t_5		London	–	x	–	x	$\frac{4}{5}$
t_6	4.0	Beijing	–	x	–	x	–
t_7		Cape Town	x	x	x	x	$\frac{6}{7}\surd$
t_8		Yosemity	–	–	x	–	$\frac{1}{2}$
t_9		Paris	–	x	–	x	$\frac{4}{5}$
t_{10}	3.0	Pittsburgh	–	x	–	x	–
user u_a				x	x	x	

The item most similar (see Formula 1.3) to the preferences of u_a is t_7

(evaluated) by u_a. This relies on the simple similarity metric shown in Formula 1.3 (Dice coefficient which is a variation of the Jaccard coefficient "intensively" taking into account category commonalities—see also [43]). The major difference to the similarity metrics introduced in the context of collaborative filtering is that similarity is measured using categories (in contrast to ratings).

$$similarity(u_a, item) = \frac{2 * |categories(u_a) \cap categories(item)|}{|categories(u_a)| + |categories(item)|} \tag{1.3}$$

Constraint-Based Recommendation

Compared to the approaches of collaborative filtering and content-based filtering, *constraint-based recommendation* [14, 16, 63] does not primarily rely on item ratings and textual item descriptions but on deep knowledge about the offered items. Such deep knowledge (semantic knowledge [16]) describes an item in more detail and thus allows for a different recommendation approach (see Table 1.8).

Table 1.8 Slightly adapted travel destinations described based on *season* (digit 1 indicates a recommended season and 0 indicates a non-recommended one; seasons start with *spring*), associated *topics*, and average user rating (*eval*)

Item	Name	Season	Topics	Eval
t_1	Vienna	1110	City tours, entertainment	4.5
t_2	Yellowstone	1110	Nature	4.0
t_3	New York	1011	City tours, entertainment	3.3
t_4	Blue Mountains	1001	Nature	5.0
t_5	London	1010	City tours, entertainment	3.0
t_6	Beijing	1010	City tours, entertainment	4.7
t_7	Cape Town	1111	Beach, city tours, nature, entertainment	4.0
t_8	Yosemity	1110	Nature	2.0
t_9	Paris	1011	City tours, entertainment	3.0
t_{10}	Pittsburgh	1010	City tours	5.0

Constraint-based recommendation relies on (a) a set of rules (constraints) and (b) a set of items. Depending on the user requirements (a set of search criteria), rules (constraints) describe which items should be recommended. The current user (u_a) articulates his/her requirements (preferences) in terms of item property specifications which are internally represented as rules (constraints). In our example, constraints are represented solely by user requirements, no further constraint types are included. An example of a constraint is the following: *topics = city tours* (the user is primarily interested in travel destinations allowing city tours). For a detailed discussion of further constraint types, we refer the reader to [16]. Constraints are interpreted and the resulting items are presented to the user. A detailed discussion of reasoning mechanisms that are used in constraint-based recommendation can be found in [14, 19, 22]. In order to determine a recommendation in a constraint-based recommendation scenario, a *recommendation task* has to be solved.

Definition (Recommendation Task) A recommendation task can be defined by the tuple (R, I) where R represents a set of user requirements and I represents a set of items (in our case: travel destinations $t_i \in I$). The goal is to identify those items in I which fulfill the given user requirements (preferences).

A solution for a recommendation task (also denoted as recommendation) can be defined as follows.

Definition (Solution for a Recommendation Task) A solution for a recommendation task (R, I) is a set $S \subseteq I$ such that $\forall t_i \in S : t_i \in \sigma_{(R)} I$ where σ is the selection operator of a conjunctive query [19], R represents a set of selection criteria (represented as constraints), and I represents an item table (see, e.g., Table 1.8). If we want to restrict the set of item properties shown to the user in a result set (recommendation), we have to additionally include projection criteria π as follows: $\pi_{(attributes(I))}(\sigma_{(R)} I)$.

In our example, we show how to determine a solution for a given recommendation task based on a conjunctive query where user requirements are used

Table 1.9 Travel destinations described with regard to the dimensions *security* (high evaluation represents a high security), *attractiveness* (high evaluation represents a high attractiveness), and *crowdedness* (high evaluation represents a low crowdedness)

Item	Name	Security	Attractiveness	Crowdedness
t_1	Vienna	5	5	2
t_2	Yellowstone	4	4	4
t_3	New York	3	5	1
t_4	Blue Mountains	4	3	5
t_5	London	3	4	1
t_6	Beijing	3	3	1
t_7	Cape Town	2	3	3
t_8	Yosemity	4	4	4
t_9	Paris	3	5	1
t_{10}	Pittsburgh	3	3	3

For example, *security* = *5* for the item *Vienna* indicates the highest possible *contribution* to the dimension *security* (scale 1..5)

as selection criteria (constraints) on an item table I. If we assume that the user requirements are represented by the set $R = \{r_1 : season = winter, r_2 : topics = city\ tours\}$ and the item table I consists of the elements shown in Table 1.8, then $\pi_{(item)}(\sigma_{(season=winter \wedge topics=city\ tours)}I) = \{t_3, t_7, t_9\}$, i.e., these three items are consistent with the given set of requirements.

Ranking Items Up to now we know which items can be recommended to a user. One widespread approach to rank items is to define a utility scheme which serves as a basis for the application of *Multi Attribute Utility Theory* (MAUT).[5] Items can be evaluated and ranked with respect to a set of *interest dimensions*. In travel destinations, example interest dimensions are *security* (security level of the travel destination), *attractiveness* (estimated attractiveness of the travel destination), and *crowdedness* (degree of crowdedness of the travel destination). The first step to establish a MAUT scheme is to relate the interest dimensions with the given set of items. An example thereof is shown in Table 1.9 where the city of *Vienna* receives the highest evaluations with regard to the dimensions *security* and *attractiveness* and a low evaluation with regard to *crowdedness*.

We are now able to determine the user-specific utility of each individual item. The calculation of *item* utilities for a specific user u_a can be based on Formula 1.4.

$$utility(u_a, item) = \sum_{d \in Dimensions} contribution(item, d) \times weight(u_a, d) \tag{1.4}$$

If we assume that the current user u_a assigns a weight of 0.6 to the dimension *security* (weight(u_a,security)=0.6), a weight of 0.3 to the dimension *attractiveness*

[5] A detailed discussion of MAUT in constraint-based recommendation is given in [2, 16, 18].

Table 1.10 Item-specific *utility* for user u_a ($utility(u_a, t_i)$) assuming the personal preferences {weight(u_a,security)=0.6, weight(u_a,attractiveness)=0.3, weight(u_a,crowdedness)=0.1}, item t_1 has the highest utility for user u_a

Item	Security	Attractiveness	Crowdedness	Utility
t_1	3.0	1.5	0.2	4.7 ✓
t_2	2.4	1.2	0.4	4.0
t_3	1.8	1.5	0.1	3.4
t_4	2.4	0.9	0.5	3.8
t_5	1.8	1.2	0.1	3.1
t_6	1.8	0.9	0.1	2.8
t_7	1.2	0.9	0.3	2.4
t_8	2.4	1.2	0.4	4.0
t_9	1.8	1.5	0.1	3.4
t_{10}	1.8	0.9	0.3	3.0

(weight(u_a,attractiveness)=0.3), and a weight of 0.1 to the dimension *crowdedness* (weight(u_a,crowdedness)=0.1), then the user-specific utilities of the individual items (t_i) are the ones shown in Table 1.10.

Managing Inconsistencies In constraint-based recommendation scenarios, we have to deal with situations where no solution (recommendation) can be identified for a given set of user requirements, i.e., $\sigma_{(R)}I = \emptyset$. In such situations, we are interested in proposals for requirements changes such that a solution can be found. For example, if a user is interested in travel destinations for *entertainment* with an overall *evaluation of 5.0* in the summer season, then no solution can be provided for the given set of requirements $R = \{r_1 : season = summer, r_2 : topics = entertainment, r_3 : eval = 5.0\}$.

User support in such situations can be based on the concepts of conflict detection [44] and model-based diagnosis [13, 15, 72]. A conflict (or conflict set) with regard to an item set I in a given set of requirements R can be defined as follows.

Definition (Conflict Set) A conflict set is a set $CS \subseteq R$ such that $\sigma_{(CS)}I = \emptyset$. CS is minimal if there does not exist a conflict set CS' with $CS' \subset CS$.

In our example, we are able to determine the following minimal conflict sets CS_i: $CS_1 : \{r_1, r_3\}$, $CS_2 : \{r_2, r_3\}$. We will not discuss algorithms that support the determination of minimal conflict sets but refer the reader to the work of Junker [44] who introduces a divide-and-conquer based algorithm with a logarithmic complexity in terms of the needed number of consistency checks.

Based on the identified minimal conflict sets, we are able to determine the corresponding (minimal) diagnoses. A diagnosis for a given set of requirements which is inconsistent with the underlying item table can be defined as follows.

Definition (Diagnosis) A diagnosis for a set of requirements $R = \{r_1, r_2, \ldots, r_n\}$ is a set $\Delta \subseteq R$ such that $\sigma_{(R-\Delta)}I \neq \emptyset$. A diagnosis Δ is minimal if there does not exist a diagnosis Δ' with $\Delta' \subset \Delta$.

In other words, a diagnosis (hitting set) is a minimal set of requirements that have to be deleted from R such that a solution can be found for R - Δ. The determination

of the *complete set of minimal diagnoses* for a set of requirements inconsistent with the underlying item table (the corresponding conjunctive query results in $\emptyset$) is based on the construction of hitting set trees [72].

There are two possibilities of resolving the conflict set CS_1. If we decide to delete the requirement r_3, $\sigma_{(\{r_1,r_2\})}I \neq \emptyset$, i.e., a diagnosis has been identified ($\Delta_1 = \{r_3\}$) and—as a consequence—all CS_i have been resolved. Choosing the other alternative and resolving CS_1 by deleting r_1 does not result in a diagnosis since the conflict CS_2 is not resolved. Resolving CS_2 by deleting r_3 does not result in a minimal diagnosis, since r_3 already represents a diagnosis. The second (and last) minimal diagnosis that can be identified in our running example is $\Delta_2 = \{r_1, r_2\}$. For a detailed discussion of the underlying algorithm and analysis, we refer to [21, 72]. Note that a diagnosis provides a hint to which requirements have to be changed. For a discussion of how requirement repairs (changes) are calculated, we refer to Felfernig et al. [19].

Critiquing-Based Recommendation

Critiquing-based recommender systems support the navigation in the item space where in each critiquing cycle a reference item is presented to the user and the user either accepts the (recommended) item or searches for different solutions by specifying *critiques* (see Fig. 1.1). Critiquing-based recommender systems are useful in situations where users are not experts in the item domain and prefer to specify their requirements on the level of critiques [46]. Critiques are basic criteria that are used for determining new recommendations which take into account the (changed) preferences of the current user. Examples of such critiques in the context of our running example are *higher level of security* and *higher attractiveness*. Critiques are used as search criteria to identify corresponding candidate items, i.e., items that are shown to the user if he/she has specified critiques on the current reference item. Critiquing-based recommenders often search for items that are similar to the current reference item and additionally take into account the new

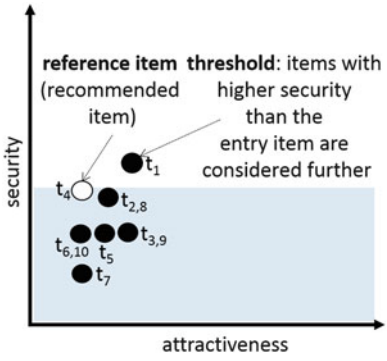


Fig. 1.1 Example of a critiquing scenario: an item (t_4) is shown as *reference item* to the user. The user specifies the critique "higher security." The new reference item is t_1 since it is consistent with the critique and the item most similar to t_4 (here, it is also the only remaining alternative)

criteria defined as critiques. When searching for a new reference item, similarity and diversity metrics are applied to systematically guide the navigation in the item space [43].

Similarity metrics in critiquing-based approaches are used to determine the similarity between a reference item currently shown to the user and a set of candidate items to be shown to the user in the next critiquing cycle. Example similarity metrics often used in the context of critiquing-based recommendation scenarios are represented by Formulae 1.5–1.9. In this context, $sim(r, c)$ denotes the similarity between the reference item r and the candidate item c, the subroutine $s(r.i, c.i)$ is represented by different *attribute-level similarity metrics* (MIB, LIB, NIB, and EIB). If a higher attribute value is better than a lower one, *More-Is-Better* (MIB) (Formula 1.6) is used to evaluate the attribute of a candidate item $(c.i)$. Vice versa, if low attributes values are considered better, the *Less-Is-Better* (LIB) similarity metric is used (Formula 1.7). Furthermore, *Nearer-Is-Better* (NIB) (Formula 1.8) is used if the attribute value of the candidate item should be as near as possible to the attribute $r.i$. Finally, *Equal-Is-Better* (EIB) is used in situations where attribute values should be equal (Formula 1.9). Besides taking into account the similarity between the reference item and candidate items, some critiquing-based systems also take into account the *compatibility* of a candidate item with regard to the complete critiquing history [61]. Thus, a trade-off between similarity to the reference item and critique compatibility can be achieved.

$$sim(r, c) = \sum_{i \in attributes} \mathrm{s}(r.i, c.i) * w(i) \quad (\Sigma_{i \in attributes} w(i) = 1) \tag{1.5}$$

$$MIB : s(r.i, c.i) = \frac{val(c.i) - minval(r.i)}{maxval(r.i) - minval(r.i)} \tag{1.6}$$

$$LIB : s(r.i, c.i) = \frac{maxval(r.i) - val(c.i)}{maxval(r.i) - minval(r.i)} \tag{1.7}$$

$$NIB : s(r.i, c.i) = 1 - \frac{|val(r.i) - val(c.i)|}{maxval(r.i) - minval(r.i)} \tag{1.8}$$

$$EIB : s(r.i, c.i) = \begin{cases} 1 & \text{if } r.i = c.i \\ 0 & \text{otherwise} \end{cases} \tag{1.9}$$

If users are knowledgeable in the item domain, the application of search-based approaches such as constraint-based recommendation makes more sense. Different types of critiquing-based approaches primarily differ in terms of the way in which user preferences can be specified. *Unit critiquing* [10, 54] only supports the definition of critiques (change requests) that are related to a single item property (attribute). *Compound critiques* allow the specification of change requests over multiple item properties and thus allow to reduce the number of needed critiquing cycles [60]. Finally, *experience-based critiquing* takes into account critiquing

histories of previous users to better predict reference items and thus to reduce the number of needed interaction cycles [54, 62]. For an in-depth discussion of different (additional) variants of critiquing-based recommendation, we refer to [10, 11, 32, 54, 60, 73].

Hybrid Recommendation

After having discussed the basic recommendation approaches of *collaborative filtering*, *content-based filtering*, *constraint-based*, and *critiquing-based recommendation*, we will now present some possibilities to combine these.

A major motivation for the development of hybrid recommender systems is the opportunity to achieve a better accuracy [9]. There are different approaches to evaluate the accuracy of recommendation algorithms. These approaches can be categorized into *predictive accuracy metrics* such as the mean absolute error (MAE), *classification accuracy metrics* such as precision and recall, and *rank accuracy metrics* such as Kendall's Tau (see Chap. 3). For a discussion of accuracy metrics in recommendation scenarios for individual users, we refer to Gunawardana and Shani [34] and Jannach et al. [43].

We now take a look at *example design types* of hybrid recommenders [9, 43]. These are *weighted*, *mixed*, and *cascade* (see Table 1.11). The basic assumption in the following example is that individual recommendation approaches return a list of *five* recommended items where each item has an assigned (recommender-individual) prediction *out of* {1.0, 2.0, 3.0, 4.0, 5.0}. For a more detailed discussion of hybridization strategies, we refer the reader to Burke [9] and Jannach et al. [43].

Weighted Weighted hybrid recommendation is based on the idea of deriving recommendations by combining the results (predictions) computed by individual recommenders. An example thereof is depicted in Table 1.12 where the individual

Table 1.11 Examples of hybrid recommendation approaches (*RECS* = set of recommenders, *s* = recommender-individual prediction, *score* = item score)

Method	Description	Calculation
Weighted	Predictions (s) of individual recommenders are summed up	score(item)=$\Sigma_{rec \in RECS}\ s(item, rec)$
Mixed	Recommender-individual predictions (s) are combined into one recommendation result	score(item) = zipper-function(item, RECS)
Cascade	The prediction of one recommender $(n-1)$ is used as input for the next recommender (n)	score(item) = s(item, rec_n) $\leftarrow$ s(item,rec_{n-1}) $\leftarrow \ldots$

Table 1.12 Example of *weighted* hybrid recommendation: individual predictions are integrated into a score

Items	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
s(t_i, collaborative filtering)	1.0	3.0	–	5.0	–	2.0	–	4.0	–	–
s(t_i, content−based filtering)	–	1.0	2.0	–	–	3.0	4.0	5.0	–	–
score(t_i)	1.0	4.0	2.0	5.0	0.0	5.0	4.0	9.0	0.0	0.0
ranking(t_i)	7	4	6	2	8	3	5	1	9	10

Item t_8 receives the best overall score (9.0)

Table 1.13 Example of *mixed* hybrid recommendation: individual predictions are integrated into one score conform the zipper principle (best collaborative filtering prediction receives score=10.0, best content-based filtering prediction receives score=9.0 and so forth)

Items	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
s(l_i, collaborative filtering)	1.0	3.0	–	5.0	–	2.0	–	4.0	–	–
s(l_i, content−based filtering)	–	1.0	2.0	–	–	3.0	4.0	5.0	–	–
score(l_i)	4.0	8.0	5.0	10.0	0.0	6.0	7.0	9.0	0.0	0.0
ranking(l_i)	7	3	6	1	8	5	4	2	9	10

item scores of a collaborative and a content-based recommender are summed up. Item t_8 receives the highest overall score (9.0) and is ranked highest by the weighted hybrid recommender.[6]

Mixed Mixed hybrid recommendation is based on the idea that predictions of individual recommenders are shown in one integrated result. For example, the results of a collaborative filtering and a content-based recommender can be ranked as sketched in Table 1.13. Item scores can be determined, for example, on the basis of the zipper principle, i.e., the item with the highest collaborative filtering prediction value receives the highest overall score (10.0), the item with the best content-based filtering prediction value receives the second best overall score, and so forth.

Cascade The basic idea of cascade-based hybridization is that recommenders in a pipeline of recommenders exploit the recommendation of the upstream recommender as a basis for deriving their own recommendation. The constraint-based recommendation approach is an example of a cascade-based hybrid recommendation approach. First, items that are consistent with the given requirements are preselected by a conjunctive query Q. We can assume, for example, that s(item,Q) = 1.0 if the item has been selected and s(item,Q) = 0.0 if the item has not been selected. In our case, the set of requirements $\{r_1 : topics = nature\}$ in the running example leads to the selection of the items $\{t_2, t_4, t_7, t_8\}$. Thereafter, these items are ranked conform to their utility for the current user (utility-based

[6] If two or more items have the same overall score, a possibility is to force a decision by lot; where needed, this approach can also be applied by other hybrid recommendation approaches.

ranking U). Based on the entries of Table 1.10, a utility-based ranking (U) would determine the item order utility(t_2) $\geq$ utility(t_8) $>$ utility(t_4) $>$ utility(t_7), assuming that the current user assigns a weight of 0.6 to the interest dimension *security* (weight(u_a,security) = 0.6), a weight of 0.3 to the interest dimension *attractiveness* (weight(u_a,attractiveness) = 0.3), and a weight of 0.1 to the interest dimension *crowdedness* (weight(u_a,crowdedness) = 0.1). In this example, the recommender Q is the first one and the results of Q are forwarded to the utility-based recommender.

Further Approaches Further hybrid recommendation approaches are the following [9]. *Switching* denotes an approach where—depending on the current situation—a specific recommendation approach is chosen. For example, if a user has a low level of product knowledge, then a critiquing-based recommender will be chosen. Vice-versa, if the user is an expert, an interface will be provided where the user is enabled to explicitly state his/her preferences on a detailed level. *Feature combination* denotes an approach where different data sources are exploited by a single recommender. For example, a recommendation algorithm could exploit semantic item knowledge in combination with item ratings (see Table 1.8).

Outlook Due to the increasing popularity of social platforms and online communities, group recommender systems are becoming an increasingly important technology [36, 58]. Example application domains of group recommendation technologies include tourism [61] (e.g., *which hotels or travel destinations should be visited by a group?*) and interactive television [57] (*which sequence of television programs will be accepted by a group?*). For the most part, group recommendation algorithms are operating on *simple items* such as hotels, travel destinations, and television programs. Examples of *complex items* are cars, round trips, and software release plans. In the remainder of this book, we will discuss different approaches that determine item recommendations for groups.

1.4 Relationship Between Algorithms and Choice Patterns

As already mentioned, recommender systems can be regarded as important support for human decision making [42]. In the following, we will briefly discuss different patterns of choice, i.e., approaches people use to solve a decision task/problem. Following the concepts introduced in [41, 42], we will explain these patterns and point out related recommendation approaches. Our major motivation for this discussion is the increasingly perceived relevance of human decision making aspects in recommendation contexts [42, 55].

Socially-Influenced Choice If someone is interested in purchasing a digital camera, has no experiences in the item domain, but knows friends who are photography enthusiasts, there is a high probability that the opinion of these friends can have an impact on the camera purchase decision. People are influenced by the opinions and advice of friends beyond item recommendation, for example, in terms of social

expectations of what is "cool" or what is "politically correct" [42]. From the viewpoint of recommender systems, *collaborative filtering* [48] simulates socially-influenced decisions where the preferences of nearest neighbors (social environment of the user) are taken as input for recommendations proposed to the current user. In such recommendation scenarios, *trust* plays an important role since only the preferences of trusted users should be taken into account by the recommendation algorithm [30, 65]. If available, an important aspect to be taken into account are *social networks* which can serve as a basis for representing a user's trust networks and also to determine the nearest neighbors relevant in the recommendation context [35]. In scenarios where groups of users have to make a decision, the personal opinion of a group member is in many cases influenced by his/her surroundings, i.e., other group members or even other groups [71]. Consequently, a key feature of group decisions is that social influence occurs not only in the environment but also within the group of decision makers themselves (see Chap. 9)—a fact that makes the socially-based pattern even more important than it is for individual choices.

Attribute-Based Choice This pattern is based on the idea that individual alternatives (items) are described by attributes which can be associated with importance values that reflect individual relevance. People can in principle evaluate an item by considering the values and importances of its attributes and computing something like a weighted sum for each item. More typically, they apply less effortful strategies. For example, *attributes* can serve as criteria to figure out whether an alternative should be taken into account (also denoted as *elimination by aspects* where users are kept in the loop), serve as a basis for the ranking of the available alternatives (utility-based ranking), or can be used for both purposes [5, 42, 67]. Due to the fact that importance of attributes and preferences for particular attribute levels are not stable and are not known in detail from the beginning [4], decision processes are iterative where users change or slightly adapt their preferences. Recommenders often integrated in such scenarios are *utility-based* and *constraint-based recommenders* [8, 14]. Constraint-based recommendation [14] is regarded as a special type of knowledge-based recommendation approach where the recommendation knowledge is represented in terms of explicitly defined attributes and constraints. The second type of knowledge-based recommendation (if we interpret item attributes and corresponding critiques as semantic knowledge) is represented by critiquing-based systems [11] where users specify change requests with regard to a reference alternative and the recommender system proposes a new candidate alternative that takes into account previous critiques. Critiques are typically defined on item attributes, therefore critiquing-based approaches can also be helpful to support attribute-based decision processes. There are situations where attribute-based decisions have to be supported for groups. Group members have to develop consensus with regard to a set of attributes, for example, arriving at a common set of importance weights or agreeing on the selection of an attribute value. If contradicting preferences occur, visualization and analysis methods can be applied to resolve inconsistencies [26].

Trial & Error Based Choice This choice pattern is often used in situations where the item domain is unknown and users want to better understand the domain and

analyze the pros and cons of the different existing options in more detail [41]. For example, customers purchase the trial version of a software, test different cars, or evaluate some potential holidays within the scope of a short-term visit. In the context of recommender systems, trial & error based decisions are supported, for example, by critiquing-based recommendation approaches [11] where a user analyzes individual recommendations and then defines criteria that should additionally be taken into account when presenting the next item. Critiquing-based approaches support explorative navigation in large search spaces which better helps to develop an understanding of the item domain [43]. Critiquing-based recommendation also plays a role when it comes to determining recommendations for groups. For example, a group makes a decision with regard to a skiing resort to visit in the next winter season [61].

Experience-Based Choice This type of choosing can be applied to situations where experiences of users from the past directly influence current decisions, often making it unnecessary to apply any of the other choice patterns (see, e.g., [3]). For example, if someone purchased a book from a specific author or a specific publisher, this experience can be exploited in future purchasing scenarios and books from one consideration set could be preferred based on these experience-based criteria. Another example is car purchasing: if a customer is completely satisfied with his/her car (specific brand), there is a good chance that he/she will stick with the brand when purchasing a new car. In this context, positive feelings from the past trigger positive feelings about specific alternatives in ongoing decision processes [42]. From the point of view of recommendation approaches, content-based recommendation implements experience-based decisions in the sense that positively-rated purchases in the past represent major criteria to recommend similar items in the future. For example, a user liked a book of a specific author, therefore a new book from the same author is recommended to the user. We want to point out that in the recommender systems community, critiquing-based recommendation [11] is *also* considered as specific type of case-based recommendation [53, 77] where items are regarded as cases and items similar to a given set of user requirements (the case description) are recommended. Since cases represent experiences from the past, critiquing-based recommendation can also be considered as representative of experience-based decisions. This becomes even clearer if we take a look at recent experience-based critiquing approaches where experiences from previous recommendation sessions are taken into account and users with similar critiquing histories receive similar recommendations [54, 62]. Here, the experience-based pattern is supplemented with the socially-based, since the experiences of other persons are being exploited as well.

Consequence-Based Choice An alternative decision strategy is to think about the potential consequences of choosing a specific alternative. In contrast to decision patterns such as attribute-based decisions and socially-influenced decisions, thinking about the consequences of making a specific decision (choosing a specific alternative) constitutes an additional dimension of a decision process. One challenge in this context is the uncertainty about the consequences of taking a decision

and how to best deal with this. Since anticipating and evaluating consequences is typically an effortful process, people are more likely to do so in some domains than in others. Low involvement items, i.e., items with lower risk impacts related to suboptimal decisions (e.g., movies, restaurants, and songs) will result in less investment in the analysis of the impact of choosing such items. In contrast, high-involvement items, i.e., items with higher associated risks, will result in higher investment in the analysis of the consequences of choosing a specific alternative. We want to point out that consequence-based decision patterns are often orthogonal, i.e., they play a role in combination with each of the mentioned recommendation approaches. Like all of the choice patterns, the consequence-based pattern can be combined with other patterns. For example, a chooser may apply the attribute-based pattern to quickly form a consideration set and then apply the consequence-based pattern to the consideration set. An important aspect of decision making is the relatively strong emphasis on the potential negative consequences of a decision—this aspect is taken into account in *prospect theory* [45], an asymmetric utility function which postulates that losses have a higher negative evaluation compared to equal gains.

Policy-Based Choice In the initial phase of a decision process, it can be the case that decision makers also define their decision policy. For example, during the weekend, if a movie recommender proposes a thriller and a cartoon, a family may apply a general policy of first consuming the movie that is suitable for children. They don't need to think in each case about the justification for this policy (i.e., that children need to go to bed earlier). When deciding about the next restaurant for a Christmas party, a company could have predefined the (meta-level) decision policy of *majority voting*, i.e., the majority defines which restaurant will be chosen for the next Christmas party. If two books with equal average ratings and high-quality reviews are in a consideration set, the customer could purchase the book from a publisher he/she previously found satisfying. A book customer may (for any of various reasons) have acquired the policy of buying a particular type of book from a particular publisher. In that case, the customer does not even need to consider books from other publishers. As these examples suggest, the policy-based decision pattern can play a role in the context of each basic recommendation scenario (*collaborative*, *content-based*, *constraint-based*, and *critiquing-based*).

1.5 Book Overview

In this chapter, we introduced recommender systems as a basic technology to support different decision scenarios. We gave an overview of recommendation algorithms and showed their application in the context of a scenario from the travel domain. Finally, we discussed the relationship between basic patterns of human choice and related supportive recommendation approaches. *The remainder of this book is organized as follows*. In Chap. 2, we provide an overview of group

recommendation algorithms with the goal of showing how basic recommendation algorithms (collaborative filtering, content-based filtering, constraint-based, and critiquing-based recommendation) can be tailored to group settings. In this context, we show the relationship of group recommenders to the algorithms discussed in Chap. 1. In Chap. 3, we sketch approaches to evaluate group recommender systems. An overview of existing applications of group recommendation technologies is provided in Chap. 4. In Chap. 5, we focus on a discussion of ways to elicit and manage user preferences. Chapter 6 deals with explanation approaches in the context of group recommendation. In Chap. 7, we introduce additional decision scenarios, for example, group-based configuration, group-based resource balancing, and group-based release planning. Chapter 8 analyzes the existence and ways to counteract decision biases that can occur in the context of group decision making. Chapter 9 focuses on the role of personality and emotion in group recommendation. Finally, the book is concluded with a summary and an outlook (see Chap. 10).

References

1. C. Aggarwal, *Recommender Systems: The Textbook* (Springer, New York, 2016)
2. L. Ardissono, A. Felfernig, G. Friedrich, A. Goy, D. Jannach, G. Petrone, R. Schäfer, M. Zanker, A framework for the development of personalized, distributed web-based configuration systems. AI Mag. **24**(3), 93–108 (2003)
3. T. Betsch, S. Haberstroh, *The Routines of Decision Making* (Lawrence Erlbaum Associates, Mahwah, NJ, 2005)
4. J. Bettman, M. Luce, J. Payne, Constructive consumer choice processes. J. Consum. Res. **25**(3), 187–217 (1998)
5. S. Bhatia, Associations and the accumulation of preference. Psychol. Rev. **120**(3), 522–543 (2013)
6. D. Billsus, M. Pazzani, Learning collaborative information filters, in *15th International Conference on Machine Learning (ICML'98)*, pp. 46–54 (1998)
7. L. Boratto, S. Carta, State-of-the-art in group recommendation and new approaches for automatic identification of groups, in *Information Retrieval and Mining in Distributed Environments*. Studies in Computational Intelligence, vol. 324 (Springer, Heidelberg, 2011), pp. 1–20
8. R. Burke, Knowledge-based recommender systems. Encycl. Library Inform. Syst. **69**(32), 180–200 (2000)
9. R. Burke, Hybrid recommender systems: survey and experiments. User Model. User-Adap. Inter. (UMUAI) **12**(4), 331–370 (2002)
10. R. Burke, K. Hammond, B. Young, The FindMe approach to assisted browsing. IEEE Expert: Intell. Syst. Appl. **12**(4), 32–40 (1997)
11. L. Chen, P. Pu, Critiquing-based recommenders: survey and emerging trends. User Model. User-Adap. Inter. (UMUAI) **22**(1–2), 125–150 (2012)
12. M. Ekstrand, J. Riedl, J. Konstan, Collaborative filtering recommender systems. Found. Trends Human-Comput. Interact. **4**(2), 81–173 (2011)
13. A. Falkner, A. Felfernig, A. Haag, Recommendation technologies for configurable products. AI Mag. **32**(3), 99–108 (2011)
14. A. Felfernig, R. Burke, Constraint-based recommender systems: technologies and research issues, in *ACM International Conference on Electronic Commerce (ICEC08)*, Innsbruck, Austria, pp. 17–26 (2008)

15. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner, Consistency-based diagnosis of configuration knowledge bases. Artif. Intell. **152**(2), 213–234 (2004)
16. A. Felfernig, G. Friedrich, D. Jannach, M. Zanker, An integrated environment for the development of knowledge-based recommender applications. Int. J. Electron. Commer. (IJEC) **11**(2), 11–34 (2006)
17. A. Felfernig, G. Friedrich, L. Schmidt-Thieme, Guest editors' introduction: recommender systems. IEEE Intell. Syst. **22**(3), 18–21 (2007)
18. A. Felfernig, B. Gula, G. Leitner, M. Maier, R. Melcher, E. Teppan, Persuasion in knowledge-based recommendation, in *3rd International Conference on Persuasive Technology*. Lecture Notes in Computer Science (Springer, Berlin, 2008), pp. 71–82
19. A. Felfernig, G. Friedrich, M. Schubert, M. Mandl, M. Mairitsch, E. Teppan, Plausible repairs for inconsistent requirements, in *Proceedings of the IJCAI'09*, Pasadena, CA, pp. 791–796 (2009)
20. A. Felfernig, W. Maalej, M. Mandl, M. Schubert, F. Ricci, Recommendation and decision technologies for requirements engineering, in *ICSE 2010 Workshop on Recommender Systems in Software Engineering (RSSE 2010)*, Cape Town, pp. 11–15 (2010)
21. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Des. Anal. Manuf. (AIEDAM) **26**(1), 53–62 (2012)
22. A. Felfernig, M. Schubert, S. Reiterer, Personalized diagnosis for over-constrained problems, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, Peking, China, pp. 1990–1996 (2013)
23. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, Toward the next generation of recommender systems, in *Multimedia Services in Intelligent Environments: Recommendation Services* (Springer, Heidelberg, 2013), pp. 81–98
24. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems, in *Recommendation Systems in Software Engineering* (Springer, Berlin, 2013), pp. 15–37
25. A. Felfernig, L. Hotz, C. Bagley, J. Tiihonen, *Knowledge-Based Configuration: From Research to Business Cases*, 1st edn. (Elsevier/Morgan Kaufmann, Waltham, MA, 2014)
26. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)*, pp. 69–72 (2016)
27. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, S. Polat-Erdeniz, An analysis of group recommendation heuristics for high- and low-involvement items, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*, Arras, pp. 335–344 (2017)
28. H. Garcia-Molina, G. Koutrika, A. Parameswaran, Information seeking: convergence of search, recommendations, and advertising. Commun. ACM **54**(11), 121–130 (2011)
29. M. Gasparic, A. Janes, What recommendation systems for software engineering recommend: a systematic literature review. J. Syst. Softw. **113**, 101–113 (2016)
30. J. Golbeck, *Computing with Social Trust* (Springer, London, 2009)
31. D. Goldberg, D. Nichols, B. Oki, D. Terry, Using collaborative filtering to weave an information tapestry. Commun. ACM **35**(12), 61–70 (1992)
32. P. Grasch, A. Felfernig, F. Reinfrank, ReComment: towards critiquing-based recommendation with speech interaction, in *7th ACM Conference on Recommender Systems* (ACM, New York, 2013), pp. 157–164
33. R. Grünig, R. Kühn, *Successful Decision-Making* (Springer, Heidelberg, 2013)
34. A. Gunawardana, G. Shani, A survey of accuracy evaluation metrics of recommendation tasks. J. Mach. Learn. Res. **10**, 2935–2962 (2009)
35. J. He, W. Chu, A social network-based recommender system (SNRS), in *Data Mining for Social Network Data*. Annals of Information Systems, vol. 12 (Springer, New York, 2010), pp. 47–74
36. T. Hennig-Thurau, A. Marchand, P. Marx, Can automated group recommender systems help consumers make better choices? J. Market. **76**(5), 89–109 (2012)

37. J. Herlocker, J. Konstan, A. Borchers, J. Riedl, An algorithmic framework for performing collaborative filtering, in *Conference on Research and Development in Information Retrieval*, Berkeley, CA, pp. 230–237 (1999)
38. C. Huffman, B. Kahn, Variety for sale: mass customization or mass confusion? J. Retail. **74**(4), 491–513 (1998)
39. A. Jameson, More than the sum of its members: challenges for group recommender systems, in *International Working Conference on Advanced Visual Interfaces*, pp. 48–54 (2004)
40. A. Jameson, B. Smyth, Recommendation to groups, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (Springer, Heidelberg, 2007), pp. 596–627
41. A. Jameson, B. Berendt, S. Gabrielli, F. Cena, C. Gena, F. Vernero, K. Reinecke, Choice architecture for human-computer interaction. Foundations and Trends in Human-Computer Interaction, vol. 7 (Now Publishers Inc., Hanover, MA, 2014)
42. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in *Recommender Systems Handbook*, 2nd edn., ed. by F. Ricci, L. Rokach, B. Shapira (Springer, New York, 2015), pp. 611–648
43. D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, *Recommender Systems – An Introduction* (Cambridge University Press, New York, 2010)
44. U. Junker, QUICKXPLAIN: preferred explanations and relaxations for over-constrained problems, in *19th International Conference on Artifical Intelligence, AAAI'04* (AAAI Press, San Jose, 2004), pp. 167–172
45. D. Kahneman, A. Tversky, Prospect theory: an analysis of decision under risk. Econometrica **47**(2), 263–291 (1979)
46. B. Knijnenburg, N. Reijmer, M. Willemsen, Each to his own: how different users call for different interaction methods in recommender systems, in *RecSys 2011*, Chicago, IL, pp. 141–148 (2011)
47. M. Kompan, M. Bielikova, Group recommendations: survey and perspectives. Comput. Inform. **33**(2), 446–476 (2014)
48. J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, J. Riedl, GroupLens: applying collaborative filtering to usenet news. Commun. ACM **40**(3), 77–87 (1997)
49. Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems. IEEE Comput. **42**(8), 30–37 (2009)
50. G. Leitner, A. Fercher, A. Felfernig, K. Isak, S. Polat Erdeniz, A. Akcay, M. Jeran, Recommending and configuring smart home installations, in *International Workshop on Configuration 2016 (ConfWS'16)*, pp. 17–22 (2016)
51. J. Levin, B. Nalebuff, An introduction to vote-counting schemes. J. Econ. Perspect. **9**(1), 3–26 (1995)
52. G. Linden, B. Smith, J. York, Amazon.com recommendations – item-to-item collaborative filtering. IEEE Internet Comput. **7**(1), 76–80 (2003)
53. F. Lorenzi, F. Ricci, Case-based recommender systems: a unifying view, in *International Conference on Intelligent Techniques for Web Personalization*, pp. 89–113 (2003)
54. M. Mandl, A. Felfernig, Improving the performance of unit critiquing, in *20th International Conference on User Modeling, Adaptation, and Personalization (UMAP 2012)*, Montreal, pp. 176–187 (2012)
55. M. Mandl, A. Felfernig, E. Teppan, M. Schubert, Consumer decision making in knowledge-based recommendation. J. Intell. Inform. Syst. **37**(1), 1–22 (2010)
56. A. Marchand, *Empfehlungssysteme für Gruppen* (EUL Verlag, 2011)
57. J. Masthoff, Group modeling: selecting a sequence of television items to suit a group of viewers. User Model. User-Adap. Inter. (UMUAI) **14**(1), 37–85 (2004)
58. J. Masthoff, Group recommender systems: combining individual models, in *Recommender Systems Handbook* (Springer, New York, 2011), pp. 677–702
59. J. McCarthy, T. Anagnost, MusicFX: an arbiter of group preferences for computer supported collaborative workouts, in *Conference on Computer Support Cooperative Work*, Seattle, WA, pp. 363–372 (1998)

60. K. McCarthy, J. Reilly, L. McGinty, B. Smyth, On the dynamic generation of compound critiques in conversational recommender systems, in *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems* (Springer, Berlin, 2004), pp. 176–184
61. K. McCarthy, L. McGinty, B. Smyth, M. Salamó, Social Interaction in the CATS Group Recommender, in *Workshop on the Social Navigation and Community based Adaptation Technologies* (2006)
62. K. McCarthy, Y. Salem, B. Smyth, Experience-based critiquing: reusing critiquing experiences to improve conversational recommendation, in *International Conference on Case-Based Reasoning (ICCBR 2010)*, Alessandria, pp. 480–494 (2010)
63. H. Mengash, A. Brodsky, A group recommender for investment in microgrid renewable energy sources, in *50th Hawaii International Conference on System Sciences*, Hawaii, pp. 1485–1494 (2017)
64. G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, W. Schanil, INTELLIREQ: Intelligent techniques for software requirements engineering, in *Prestigious Applications of Intelligent Systems Conference (PAIS)*, pp. 1161–1166 (2014)
65. J. O'Donovan, B. Smyth, Trust in recommender systems, in *ACM IUI 2005*, San Diego, CA, pp. 167–174 (2005)
66. D. Paraschakis, Recommender systems from an industrial and ethical perspective, in *10th ACM Conference on Recommender Systems*, Boston, MA, pp. 463–466 (2016)
67. J. Payne, J. Bettman, E. Johnson, *The Adaptive Decision Maker* (Cambridge University Press, New York, 1993)
68. M. Pazzani, D. Billsus, Learning and revising user profiles: the identification of interesting web sites. Mach. Learn. **27**(3), 313–331 (1997)
69. K. Peniwati, Criteria for evaluating group decision-making methods. Math. Comput. Model. **46**(7–8), 935–947 (2007)
70. R. Petty, J. Cacioppo, D. Schumann, Central and peripheral routes to advertising effectiveness: the moderating role of involvement. J. Consum. Res. **10**(2), 135–146 (1983)
71. L. Recalde, A social framework for set recommendation in group recommender systems, in *European Conference on Information Retrieval* (Springer, New York, 2017), pp. 735–743
72. R. Reiter, A theory of diagnosis from first principles. Artif. Intell. J. **32**(1), 57–95 (1987)
73. F. Ricci, Q. Nguyen, Acquiring and revising preferences in a critique-based mobile recommender systems. IEEE Intell. Syst. **22**(3), 22–29 (2007)
74. B. Sarwar, G. Karypis, J. Konstan, J. Riedl, Item-based collaborative filtering recommendation algorithms, in *10th WWW Conference*, pp. 285–295 (2001)
75. G. Sielis, C. Mettouris, G. Papadopoulos, A. Tzanavari, R. Dols, Q. Siebers, A context aware recommender system for creativity support tools. J. Universal Comput. Sci. **17**(12), 1743–1763 (2011)
76. B. Smith, G. Linden, Two decades of recommender systems at Amazon.com. IEEE Internet Comput. **21**(3), 12–18 (2017)
77. B. Smyth, Case-based recommendation, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (Springer, Berlin/Heidelberg, 2007), pp. 342–376
78. M. Stettinger, A. Felfernig, CHOICLA: Intelligent decision support for groups of users in context of personnel decisions, in *ACM RecSys'2014 IntRS Workshop*, Foster City, CA, pp. 28–32 (2014)
79. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, M. Jeran, Counteracting serial position effects in the CHOICLA group decision support environment, in *20th ACM Conference on Intelligent User Interfaces (IUI2015)*, Atlanta, GA, pp. 148–157 (2015)
80. N. Tintarev, J. Masthoff, Designing and evaluating explanations for recommender systems, in *Recommender Systems Handbook* (Springer, New York, 2011), pp. 479–510
81. N. Tintarev, J. O'Donovan, A. Felfernig, Human interaction with artificial advice givers. ACM Trans. Interact. Intell. Syst. **6**(4), 1–10 (2016)
82. A. Valdez, M. Ziefle, K. Verbert, A. Felfernig, A. Holzinger, Recommender systems for health informatics: state-of-the-art and future perspectives, in *Machine Learning for Health Informatics* (Springer, Cham, 2016), pp. 391–414

Chapter 2
Algorithms for Group Recommendation

Alexander Felfernig, Müslüm Atas, Denis Helic,
Thi Ngoc Trang Tran, Martin Stettinger, and Ralph Samer

Abstract In this chapter, our aim is to show how group recommendation can be implemented on the basis of recommendation paradigms for individual users. Specifically, we focus on collaborative filtering, content-based filtering, constraint-based, critiquing-based, and hybrid recommendation. Throughout this chapter, we differentiate between (1) *aggregated predictions* and (2) *aggregated models* as basic strategies for aggregating the preferences of individual group members.

2.1 Introduction

As discussed in Chap. 1, there are many real-world scenarios where recommendations have to be made to groups. The main task in these scenarios is to generate relevant recommendations from the preferences (evaluations) of individual group members. As illustrated in Table 2.1, group recommendation approaches can be differentiated with regard to the following characteristics [45, 46].

Preference Aggregation Strategy In group recommender systems, there are two basic aggregation strategies [34]. First, recommendations are determined for individual group members and then aggregated into a group recommendation.[1] Second, the preferences of individual users are aggregated into a *group profile* which is then used to determine a group recommendation. In this chapter, we show how both strategies can be applied with different recommendation algorithms.

Recommendation Algorithm The recommendation logic of group recommenders is in many cases based on single user recommenders (collaborative filtering, content-

[1] One can also distinguish between the *aggregation of items* and the *aggregation of evaluations* (e.g., *ratings* in collaborative filtering) [6, 34]—in this chapter we will provide examples of both.

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_2

Table 2.1 Characteristics to classify group recommenders [45, 46]

Characteristic	Description
Preference aggregation strategy	(1) Determination of *items/ratings* for individual group members, thereafter aggregation of these items/ratings to a group recommendation, or (2) aggregation of the preferences of group members into a *group profile*, thereafter determination of a recommendation for the group
Recommendation algorithm	One of the recommendation algorithms (i.e., collaborative, content-based, constraint-based, critiquing-based, and hybrid)
Preferences known beforehand?	For example, in collaborative filtering, ratings are known beforehand [2]. In conversational approaches, preferences are constructed over time [36]
Immediate item consumption?	Group recommenders can recommend (1) items that will be *consumed in the future* (e.g., holiday destinations as a basis for a final decision taken by (a) a responsible person or (b) a group on the basis of a discussion [35]), or (2) items *consumed immediately* (e.g., songs [45])
Active or passive group?	A group is *passive* if it does not actively influence the construction of a group profile [2]. *Active* groups negotiate the group profile [33, 50]
Number of Recommended Items	A group recommender can focus on the recommendation of (1) a single item as is the case with travel destinations [33] or (2) multiple items represented, for example, as a sequence (e.g., television items [45])
Type of Preference Acquisition	Preferences can be acquired by interpreting, for example, the ratings of items or by engaging users in a preference construction process [35]

based filtering, constraint-based, critiquing-based, and hybrid recommendation) [24] combined with selected *aggregation functions* from *social choice theory* [45, 52]. These functions will be discussed on the basis of examples from the travel domain introduced in Chap. 1.

Preferences Known Beforehand Consider the example of single-shot recommendations determined on the basis of collaborative filtering. Some user preferences are already known from previous recommendation sessions, and so do not need to be determined in an iterative process. In contrast, conversational recommender systems [10, 12, 18, 42, 47, 49] engage users in a dialog to elicit user preferences.

Immediate Item Consumption On the one hand, a pragmatics of a recommendation can be that a group directly experiences the recommended items. For example, consider songs consumed by members of a fitness studio or commercials shown on public screens. On the other hand, recommendations are often interpreted as proposals without the items being experienced immediately.

Active or Passive Group On the one hand, group profiles can be generated automatically if the preferences of the group members are known. On the other hand, especially when using constraint-based or critiquing-based recommenders,

preferences are constructed (i.e., not known beforehand) and thus are adapted and extended within the scope of negotiation processes. The more intensively group models are discussed and negotiated, the higher the degree of group activity.

Number of Recommended Items The output of a group recommender can be a single item (e.g., restaurant for a dinner or a movie), but also packages (e.g., travel packages), sequences (e.g., songs or travel plans), and even configurations (e.g., software release plans and cars).[2]

Type of Preference Acquisition Preferences can be collected *implicitly* (through observation, for example, of user's item consumption patterns) or *explicitly* by engaging users in a preference construction process. The latter is the case especially in conversational recommendation [10, 12, 18, 42, 47].

2.2 Preference Aggregation Strategies

Independent of the way preferences are acquired from individual group members (see Chap. 5), a group recommendation is determined by aggregating these preferences in one way or another [34]. In group recommender systems, the determination of recommendations depends on the chosen *preference aggregation strategy* [2, 6, 27, 34, 37, 43, 62].

There are two aggregation strategies (see Fig. 2.1): (1) *aggregating recommended items* (or *evaluations*) that were generated separately for each user profile $up(u_i)$ and (2) *aggregating individual user profiles* $up(u_i)$ into a group profile gp. In the first case, *the recommendation step precedes the aggregation step*—item evaluations or items recommended to individual group members are *aggregated* into a corresponding group recommendation. In the second case, *the aggregation step precedes the recommendation step*—group profiles aggregated from individual user profiles are the basis for determining a group recommendation. Following the discussions in [3, 34], we denote the first aggregation strategy as *aggregated predictions* and the second one as *aggregated models* (see Fig. 2.1).

Aggregated Predictions There are two basic approaches to aggregate predictions. *First*, recommendations (items) determined for individual group members can be merged. This approach can be used if a *a set of candidate solutions* should be presented and the group members are in charge of selecting one out of the candidate items. In this context, specific items which are not very appealing for some group members are not filtered out. Group members play an important role in the decision making process, since *no ranking* of the individual candidate items is provided. *Second*, group-member-specific predictions for candidate items are aggregated. The outcome of this approach is a *ranking of candidate items*.

[2] See Chap. 7.

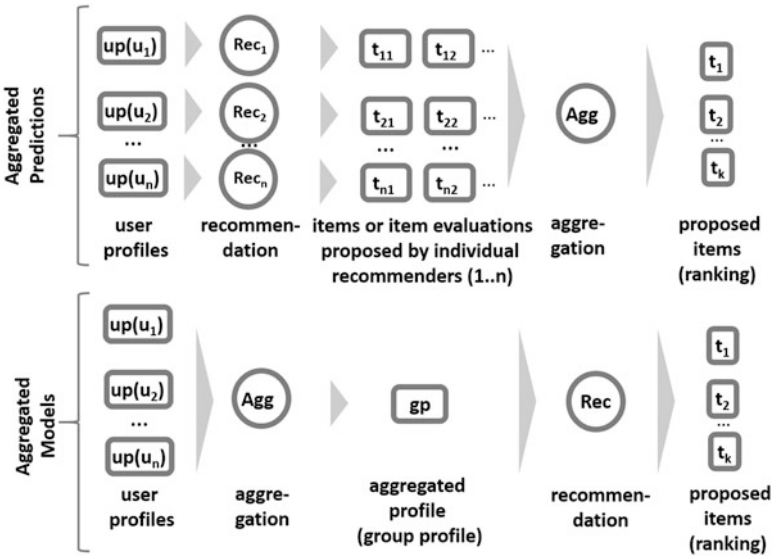


Fig. 2.1 Two basic aggregation strategies in group recommendation: (1) recommendation based on *single user profiles* with a downstream aggregation of items (or evaluations/ratings) recommended to group members/users (*aggregated predictions*) and (2) recommendation based on *aggregated models* (*group profiles*)

Aggregated Models Instead of aggregating recommendations for individual users, this approach constructs a *group preference model* (*group profile*) that is then used for determining recommendations. This is especially useful in scenarios where group members should have the opportunity to *analyze, negotiate, and adapt the preferences of the group* [34]. Another advantage of applying group preference models is that the *privacy concerns* of users can be alleviated, since there is no specific need to record and maintain individual user profiles.

Although studies exist that compare the predictive quality of the two basic aggregation approaches (*aggregated predictions* and *aggregated models*) [2, 3, 5, 13], more in-depth comparisons are needed that also focus on specific group properties such as size, homogeneity (e.g., similarity between group members can have a negative impact on the decision quality), the item domain (e.g., high-involvement vs. low-involvement items [26]), and also the ways in which individual and group rating behavior differs [56]. After introducing a couple of *social choice based preference aggregation functions* that help to implement aggregated predictions and aggregated models, we show how preference aggregation can be implemented in the context of *collaborative-* and *content-based filtering* as well as *constraint-based*, *critiquing-based*, and *hybrid recommendation*.

2.3 Social Choice Based Preference Aggregation Functions

A major issue in all of the mentioned group recommendation scenarios is how to adapt to the group as a whole, given information about the individual preferences of group members [1, 45]. As there is *no optimal way* to aggregate recommendation lists [1], corresponding approximations (in the following denoted as *aggregation functions*) have to be used to come up with a recommendation that takes into account "as far as possible" the individual preferences of group members. As mentioned in [46, 57], the aggregation functions can be categorized into *majority-based* (*M*), *consensus-based* (*C*), and *borderline* (*B*). Table 2.2 provides an overview of different kinds of *aggregation functions* taken from *social choice theory*[3] [11, 44, 45, 58] and their categorization into one of the three mentioned categories (*M*, *C*, and *B*).

Majority-Based Aggregation Functions (M) represent aggregation mechanisms that focus on those items which are the *most popular* [44, 58]. Examples of majority-based functions are *Plurality Voting* (PLU) (winner is the item with the highest number of *votes*), *Borda Count* (BRC) (winner is the item with the best *total ranking score* where each item rank[4] is associated with a score 0 .. #*items*−1), and *Copeland Rule* (COP) (winner is the item that outperforms other items in terms of pairwise *evaluation*[5] comparison) (see Table 2.3). Equal evaluations in BRC are handled as follows: in the example of Table 2.3, user u_2 provided the rating 2.5 for t_2 and t_3; both items receive the same *score* which is $\frac{0+1}{2} = 0.5$.

When comparing the items t_1 and t_2 in Table 2.3, t_1 outperforms t_2 two times and loses once in terms of user evaluations ($u_1 : 5.0$ vs. $u_2 : 3.0$, $u_1 : 4.5$ vs. $u_2 : 2.5$, and $u_1 : 3.5$ vs. $u_2 : 4.0$) which results in a win ("+") 2:1. Comparing items t_2 and t_3 results in a tie 1:1 which is indicated by "0" in Table 2.3. Such an evaluation has to be performed for each item in order to determine a winner on the basis of COP (see the *rhs* of Table 2.3). A further majority-based aggregation function is *Approval Voting* (APP) that recommends items with the highest number of supporting users. In this context, support is measured in terms of the number of item evaluations above a defined threshold.

Consensus-Based Functions (C)[6] represent aggregation mechanisms that take into account the preferences of *all group members* [58]. Examples are *Additive Utilitarian* (ADD) (winner is the item with the maximum sum of user-individual evaluations), *Average* (AVG) (winner is the item with the maximum average of the user-individual evaluations—in the line of ADD,[7] the function causes problems

[3] Also denoted as *group decision making*.

[4] The highest rank is assumed to be 1. For example, in collaborative filtering it is associated with the highest rating. The highest rank is associated with the score *#items-1*.

[5] For example, when using collaborative filtering, evaluations are denoted as *ratings*.

[6] Also denoted as *democratic functions*.

[7] ADD and AVG result in the same rankings.

Table 2.2 Basic *aggregation functions* for group recommender systems [11, 40, 45, 46, 57] where *argmax* is assumed to return a recommended item. Tie breaking rules such as *random selection* can be applied. M, C, and B denote the aggregation categories *majority-based*, *consensus-based*, and *borderline*; u represents a *user* (*group member*), G a *group*, t an *item*, and I a set of *items*

Aggregation strategy	Description	Recommendation
Additive Utilitarian (ADD) [C]	Sum of item-specific evaluations	$\underset{(t \in I)}{argmax}(\Sigma_{u \in G} eval(u,t))$
Approval Voting (APP) [M]	Number of item-specific evaluations above an approval threshold	$\underset{(t \in I)}{argmax}(\lvert\{u \in G : eval(u,t) \geq threshold\}\rvert)$
Average (AVG) [C]	Average of item-specific evaluations	$\underset{(t \in I)}{argmax}(\frac{\Sigma_{u \in G} eval(u,t)}{\lvert G \rvert})$
Average without Misery (AVM) [C]	Average of item-specific evaluations (if all evaluations are above a defined threshold)	$\underset{(t \in I : \nexists u \in G \lvert eval(u,t) \leq threshold)}{argmax}(\frac{\Sigma_{u \in G} rating(u,t)}{\lvert G \rvert})$
Borda Count (BRC) [M]	Sum of item-specific scores derived from item ranking	$\underset{(t \in I)}{argmax}(\Sigma_{u \in G} score(u,t))$
Copeland Rule (COP) [M]	Number wins (w)—number losses (l) in pair-wise evaluation comparison	$\underset{(t \in I)}{argmax}(\lvert w(t, I - \{t\})\rvert - \lvert l(t, I - \{t\})\rvert)$
Fairness (FAI) [C]	Item ranking as if individuals ($u \in G$) choose them one after the other	$\underset{(t \in I)}{argmax}(eval(u,t))$ [in each iteration]
Least Misery (LMS) [B]	Minimum item-specific evaluation	$\underset{(t \in I)}{argmax}(mineval(t))$
Majority Voting (MAJ) [B]	Majority of evaluation values per item	$\underset{(t \in I)}{argmax}(majorityeval(t))$
Most Pleasure (MPL) [B]	Maximum item-specific evaluation	$\underset{(t \in I)}{argmax}(maxeval(t))$
Most Respected Person (MRP) [B]	Item-evaluations of most respected user	$\underset{(t \in I)}{argmax}(eval(u_{mrp}, t))$
Multiplicative (MUL) [C]	Multiplication of item-specific evaluations	$\underset{(t \in I)}{argmax}(\Pi_{u \in G} eval(u,t))$
Plurality Voting (PLU) [M]	Item with the highest #votes from $u \in G$	$\underset{(t \in I)}{argmax}(votings(t))$ [in each iteration]

in the context of larger groups since the opinions of individuals count less), and *Multiplicative* (MUL) (winner is the item with the maximum product of the user-individual evaluations) (see Table 2.4). Further majority-based aggregation functions are *Average without Misery* (AVM) that recommends the average evaluation for items that do not have individual ratings below a defined threshold and *Fairness* (FAI) which ranks items as if individuals are choosing them in turn [45].

Table 2.3 Examples of *majority-based* aggregation: Plurality Voting (PLU), Borda Count (BRC), and Copeland Rule (COP, "+" indicates a win, "−" a loss, and "0" a tie)

	Votes				Evaluations (scores)				Evaluations			COP index			
Item	u_1	u_2	u_3	*PLU*	u_1	u_2	u_3	*BRC*	u_1	u_2	u_3	t_1	t_2	t_3	*COP*
t_1	1	1	0	2 √	5.0(2)	4.5(2)	3.5(1)	5 √	5.0	4.5	3.5	0	+	+	2 √
t_2	0	0	1	1	3.0(0)	2.5(0.5)	4.0(2)	2.5	3.0	2.5	4.0	–	0	0	−1
t_3	0	0	0	0	3.5(1)	2.5(0.5)	1.5(0)	1.5	3.5	2.5	1.5	–	0	0	−1

√ denotes the item t_i with the best evaluation, i.e., the *recommendation*

Table 2.4 Examples of *consensus-based* aggregation: Additive Utilitarian (ADD), Average (AVG), and Multiplicative (MUL)

	Evaluations					
Item	u_1	u_2	u_3	*ADD*	*AVG*	*MUL*
t_1	5	2	2	9	3	20
t_2	3	3	4	10 √	3.3 √	36 √
t_3	2	3	2	7	2.3	12

Table 2.5 Examples of *Borderline* aggregation: Least Misery (LMS), Most Pleasure (MPL), and Majority Voting (MAJ)

	Evaluations					
Item	u_1	u_2	u_3	*LMS*	*MPL*	*MAJ*
t_1	5	2	2	2	5 √	2
t_2	3	3	4	3 √	4	3 √
t_3	2	3	2	2	3	2

Borderline Functions (B) represent aggregation mechanisms that take into account only *a subset of the user preferences* [58]. Examples of borderline functions are *Least Misery* (LMS) (winner is the item with the highest of all lowest evaluations given to items—when using this function, items may be selected that nobody hates but also nobody really likes; furthermore, there is the danger that a minority dictates the group (especially in settings involving larger groups) [44]), *Most Pleasure* (MPL) (winner is the item with the highest of all individual evaluations—items may be selected that only a few persons really like),[8] and *Majority Voting* (MAJ) (item with the highest number of all evaluations representing the majority of item-specific evaluations) (see Table 2.5). A further borderline aggregation function is *Most Respected Person* (*MRP*) that recommends a rating (evaluation) proposed by the most respected individual.

The following discussions of group recommendation approaches will be based on a set of example items from the travel domain (see Chap. 1). Using these items, we will show how different recommendation approaches can determine group recommendations with *aggregated models* and *aggregated predictions*.

[8] Variants thereof can be considered [44], for example, *most pleasure without misery* where only items are considered that do not have evaluations below a predefined threshold.

2.4 Collaborative Filtering for Groups

Collaborative filtering (CF) [38, 41] is based on the idea of recommending items that are derived from the preferences of *nearest neighbors*, i.e., users with preferences similar to those of the current user. In the following, we show how *aggregated predictions* and *aggregated models* can be applied to CF for groups.

Aggregated Predictions When applying the *aggregated predictions* strategy in combination with collaborative filtering, ratings are determined for individual users and then aggregated into a recommendation for the group (see Fig. 2.2).

Following this approach, for each group member (and corresponding recommender) i and each item j not rated by this group member, a rating prediction $\hat{r}_{ij}$ is determined [4]. For simplicity, we assume that the items $\{t_1, .., t_{10}\}$ in Table 2.6 have not been previously consumed by the group members, i.e., the rating has been proposed by a collaborative filtering algorithm.[9] Thereafter, these predictions are aggregated on the basis of different aggregation functions (see Table 2.2). In the following example, we assume that some variant of collaborative filtering [17, 28, 53] has already been applied to predict ratings (e.g., a matrix factorization approach [51, 56] can be applied to infer *user* × *item* rating tables as shown in Table 2.6).

The result of the aggregation step is a *ranking of candidate items*. In our example, the majority of aggregation functions recommends the item t_6.

An alternative to the aggregation of ratings is to *aggregate predicted items* where *items* determined by individual recommenders are aggregated into a group recommendation (see Fig. 2.3).

Following this approach, items with the highest predicted rating for a specific user are considered as part of the recommendation. If we want to generate a recommendation consisting of, for example, at most 10 items, the two top-rated

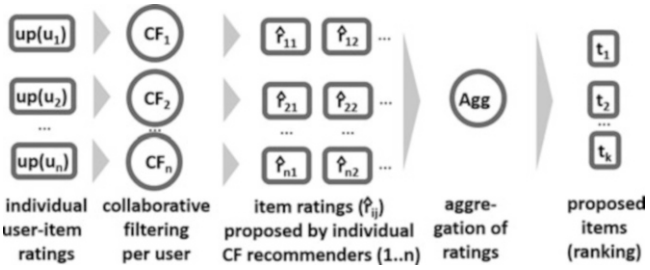


Fig. 2.2 Collaborative filtering for groups based on *aggregated predictions* (*ratings*). $\hat{r}_{ij}$ is the rating prediction for item j proposed by recommender i ($i = 1..n$)

[9] Item predictions for individual users can be based on collaborative recommendation approaches as introduced in Chap. 1.

Table 2.6 Rating predictions and corresponding *scores* (scores are used by *BRC*)

Name Item		Rating predictions $\hat{r}_{ij}$ (scores)					Aggregation		
		u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	Vienna	5.0(9)	3.5(2)	1.0(0)	4.5(7)	5.0(9)	3.8	27	1.0
t_2	Yellowstone	2.5(0)	4.0(4)	3.0(3)	2.0(0)	1.1(0)	2.5	7	1.1
t_3	New York	4.9(8)	3.8(3)	4.0(7)	3.3(4)	4.0(5)	4.0	27	3.3 $\surd$
t_4	Blue Mountains	3.1(2)	5.0(9)	4.2(8)	2.4(1)	4.4(8)	3.8	28	2.4
t_5	London	4.0(4)	4.3(7)	3.3(5)	4.1(6)	2.9(3)	3.7	25	2.9
t_6	Beijing	4.5(6)	4.1(5)	5.0(9)	3.2(3)	4.2(6)	4.2 $\surd$	29 $\surd$	3.2
t_7	Cape Town	4.2(5)	4.2(6)	3.4(6)	3.1(2)	3.8(4)	3.7	23	3.1
t_8	Yosemity	3.4(3)	2.6(0)	1.6(1)	5.0(9)	2.4(2)	3.0	15	1.6
t_9	Paris	4.7(7)	3.1(1)	2.7(2)	3.6(5)	2.2(1)	3.3	16	2.2
t_{10}	Pittsburgh	2.6(1)	4.5(8)	3.1(4)	4.6(8)	4.3(7)	3.8	28	2.6

Recommendations are derived on the basis of aggregation functions (*AVG*, *BRC*, *LMS*). The $\surd$ symbol indicates the item with the best evaluation

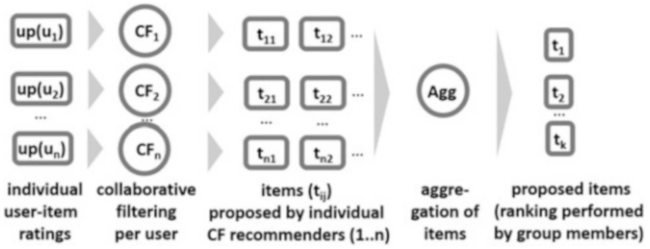


Fig. 2.3 Collaborative filtering for groups based on *aggregated predictions* (*items*)

items (upper bound) in each group member specific recommendation can be included in the group recommendation. In the example shown in Table 2.6, $\{t_1, t_3\}$ are the two top-rated items of user u_1, $\{t_4, t_{10}\}$ are chosen for user u_2, $\{t_4, t_6\}$ for user u_3, $\{t_8, t_{10}\}$ for user u_4, and $\{t_1, t_4\}$ for user u_5. The union of these group member individual recommendations is $\{t_1, t_3, t_4, t_6, t_8, t_{10}\}$ which represents the group recommendation—in this context, group members are in charge of item ranking. This way of constructing a group recommendation is similar to the idea of the *Fairness* (FAI) aggregation function (see Table 2.2).

Aggregated Models When using this aggregation approach, ratings of individual users are aggregated into a group profile *gp* (see Fig. 2.4). Based on the group profile (*gp*), collaborative filtering determines a ranking for each candidate item.

In the aggregated models approach, the group is represented by a *group profile* (*gp*) that includes item-specific evaluations (ratings) derived through aggregation functions applied to the item ratings of individual group members. Often, the aggregation is based on a weighted average function (see, e.g., [4]), however, the aggregation functions mentioned in Table 2.2 can be considered alternatives.

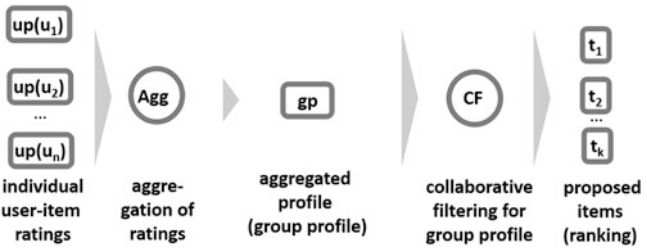


Fig. 2.4 Collaborative filtering for groups based on *aggregated models*

Table 2.7 Applying collaborative filtering (CF) to a group profile *gp* (*gp*-ratings have no relationships to earlier examples)

Item	Name	*gp*	*gx* ∈ *NN*	*gy* ∈ *NN*	Recommended ratings
t_1	Vienna	5.0	5.0	4	–
t_2	Yellowstone	–	4.0	4.5	4.49 √
t_3	New York	4.0	3.0	3.5	–
t_4	Blue Mountains	–	4.5	4	4.44
t_5	London	4.0	3.9	3.5	–
t_6	Beijing	–	3.5	3	3.44
t_7	Cape Town	–	4.7	3	3.99
t_8	Yosemity	3.0	3.8	3.2	–
t_9	Paris	4.0	3.9	2.9	–
t_{10}	Pittsburgh	–	5.0	3.3	4.28
Average		4.0	4.13	3.5	–

The √ symbol indicates the item with the best CF-based evaluation

Following the aggregated models strategy, collaborative filtering is applied to individual group profiles, i.e., for a given group profile (*gp*), similar group profiles (*k* nearest neighbors *k-NN*)[10] are retrieved and used for determining a recommendation. In our example, the item t_2 (*Yellowstone*) is *not* known to the current group *gp* but received the highest ratings from the nearest neighbor groups *gx* and *gy* (see Table 2.7) which makes it a recommendation candidate for *gp*.

The *similarity* between the group profile *gp* and another group profile *gx* (the nearest neighbor) can be determined, for example, using the Pearson correlation coefficient (see Chap. 1). Formula 2.1 is an adapted version that determines the similarity between a group profile and the profiles of other groups. In this context, TD_c represents the set of items that have been rated by both groups (*gp* and *gx*), r_{gx,t_i} is the rating of group *gx* for item t_i, and $\overline{r_{gx}}$ is the average rating of group *gx*.

[10] In our example, we assume $k = 2$.

$$similarity(gp, gx) = \frac{\sum_{t_i \in TD_c}(r_{gp,t_i} - \overline{r_{gp}}) \times (r_{gx,t_i} - \overline{r_{gx}})}{\sqrt{\sum_{t_i \in TD_c}(r_{gp,t_i} - \overline{r_{gp}})^2} \times \sqrt{\sum_{t_i \in TD_c}(r_{gx,t_i} - \overline{r_{gx}})^2}} \quad (2.1)$$

The information about groups with a similar rating behavior (i.e., nearest neighbors *NN*) compared to the current group *gp* is the basis for predicting the rating of *gp* for an *item t* that has not been rated by members of *gp* (see Formula 2.2).

$$prediction(gp, t) = \hat{r}(gp, t) = \overline{r_{gp}} + \frac{\sum_{gj \in NN} similarity(gp, gj) \times (r_{gj,t} - \overline{r_{gj}})}{\sum_{gj \in NN} similarity(gp, gj)} \quad (2.2)$$

Recommendations can also be determined on the basis of *ensemble voting* [59]: each aggregation function can represent a vote. The more such votes an item receives, the higher is its relevance for the group. In our running example, item t_6 is regarded as favorite item since it received the best evaluation by the majority of the used aggregation functions (see the *aggregated predictions* example in Table 2.6).

2.5 Content-Based Filtering for Groups

Content-based filtering (CBF) is based on the idea of recommending new items with *categories*[11] similar to those preferred by the current user. Categories preferred by a user (group member) are stored in a user profile; these categories are derived from descriptions of items already consumed by the user.

Aggregated Predictions When using this aggregation strategy, group member individual content-based recommenders determine the similarity between (a) items not consumed by him/her and (b) his/her user profile.[12] The identified item similarities (or items) are then aggregated and thus form the basis of a group recommendation (see Fig. 2.5).

Table 2.8 depicts example profiles of group members $u_1..u_5$. For each of these profiles, the similarity to the items included in Table 1.8 is determined (we assume that these items have not been consumed/evaluated by the group members). These similarity values are the basis for a group recommendation (see Table 2.9).

The user-item similarities of Table 2.9 are calculated by a content-based recommender (*similarity* metric 1.3 in Chap. 1). The calculation is based on the item categories included in Table 2.8, i.e., *beach*, *city tours*, *nature*, and *entertainment*. For example, $similarity(u_1, t_2) = \frac{2*|categories(u_1) \cap categories(t_2)|}{|categories(u_1)|+|categories(t_2)|} = \frac{2}{3} = 0.66$.

[11] Alternatively, *keywords* extracted from item descriptions.

[12] The determination of *user* × *item similarities* can be based on content-based recommendation approaches as discussed in Chap. 1.

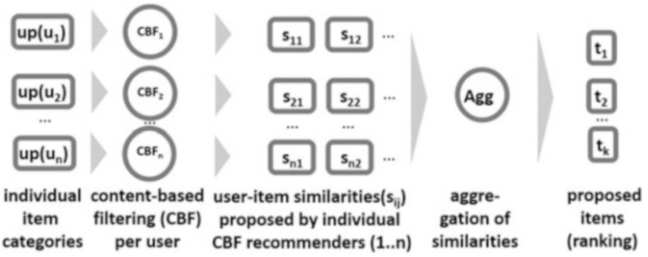


Fig. 2.5 Content-based filtering for groups based on *aggregated predictions*. Similarity s_{ij} denotes the similarity between user i and item j determined by recommender i ($i = 1..n$)

Table 2.8 Example profiles of group members (preferences regarding travel destinations)

	Individual item categories				
Category	u_1	u_2	u_3	u_4	u_5
Beach	x	x	x	x	x
City tours	–	x	–	x	–
Nature	x	–	x	–	x
Entertainment	–	–	–	–	–

If a group member u_i likes a category, this is denoted with "x"

Table 2.9 User × item similarities (and corresponding *scores* used by *BRC*) as input for *AVG*, *BRC*, *LMS* to derive a group recommendation

		User-item similarities (scores)					Aggregation		
Item	Name	u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	Vienna	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_2	Yellowstone	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_3	New York	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_4	Blue Mountains	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_5	London	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_6	Beijing	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_7	Cape Town	0.66(7.5)	0.66(8.5)	0.66(7.5)	0.66(8.5)	0.66(7.5)	0.66√	39.5√	0.66√
t_8	Yosemity	0.66(7.5)	0(1)	0.66(7.5)	0(1)	0.66(7.5)	0.4	24.5	0
t_9	Paris	0(2.5)	0.5(5)	0(2.5)	0.5(5)	0(2.5)	0.2	17.5	0
t_{10}	Pittsburgh	0(2.5)	0.66(8.5)	0(2.5)	0.66(8.5)	0(2.5)	0.26	24.5	0

The √ symbol indicates the item with the best evaluation

On the basis of a user × item similarity matrix, aggregation functions can determine a group recommendation. An alternative to the aggregation of similarities is to *aggregate items* proposed by individual content-based recommenders. If we want to generate a recommendation consisting of, for example, at most 5 items (upper bound), the highest rated item of each group member can be included in the

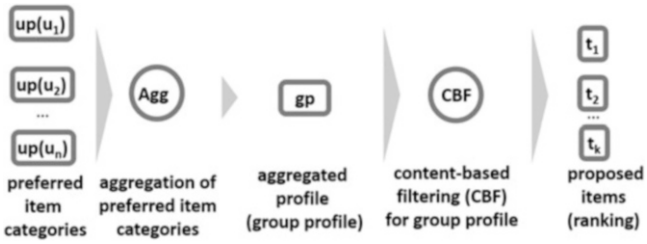


Fig. 2.6 Content-based filtering for groups based on *aggregated models*

Table 2.10 Aggregation of preferences (categories) of group members into a group profile *gp*

	Individual item categories					
Category	u_1	u_2	u_3	u_4	u_5	gp
Beach	x	x	x	x	x	x
City tours	–	x	–	x	–	x
Nature	x	–	x	–	x	x
Entertainment	–	–	–	–	–	–

group recommendation. In our example depicted in Table 2.9, $\{t_2\}$ is among the highest rated items of user u_1 (the other three are excluded due to the user-specific limit of one item), t_7 can be selected for user u_2, t_4 for user u_3, t_{10} for user u_4, and t_2 for user u_5. The group recommendation includes all of these items: $\{t_2, t_4, t_7, t_{10}\}$.

Aggregated Models When using this strategy, preferred categories of individual users are integrated into a group profile *gp*. Thereafter, content-based filtering determines recommendations by calculating the similarities between *gp* and candidate items (items not consumed by the group—see Fig. 2.6).

In our example (see Table 2.10), the derived group profile is represented by the union of the categories stored in the individual user profiles. Items are recommended that are similar to the categories in the group profile and have not been consumed by group members. In our example, the derived group profile *gp* entails the categories *Beach*, *City Tours*, and *Nature*.

The similarity between the group profile *gp* and candidate items can be determined using Formula 2.3 which is an adaption of Formula 1.3 to group settings. The similarities between *gp* and items t_i (taken from our example itemset shown in Table 1.8) are determined by comparing the categories *beach*, *citytours*, *nature*, and *entertainment* (see Table 2.11). For example, $similarity(gp, t_1) = \frac{2*|categories(gp) \cap categories(t_1)|}{|categories(gp)|+|categories(t_1)|} = \frac{2}{5} = 0.4$. In this context, we assume that the items of Table 2.11 have not been consumed by the group.

$$similarity(gp, item) = \frac{2 * |categories(gp) \cap categories(item)|}{|categories(gp)| + |categories(item)|} \tag{2.3}$$

Table 2.11 Applying content-based filtering (CBF) to a group profile *gp* (see Table 2.10)

Item	Name	Similarity(gp, t_i)
t_1	Vienna	$\frac{2}{5} = 0.4$
t_2	Yellowstone	$\frac{2}{4} = 0.5$
t_3	New York	$\frac{2}{5} = 0.4$
t_4	Blue Mountains	$\frac{2}{4} = 0.5$
t_5	London	$\frac{2}{5} = 0.4$
t_6	Beijing	$\frac{2}{5} = 0.4$
t_7	Cape Town	$\frac{6}{7} = 0.86$ $\surd$
t_8	Yosemity	$\frac{2}{4} = 0.5$
t_9	Paris	$\frac{2}{5} = 0.4$
t_{10}	Pittsburgh	$\frac{2}{4} = 0.5$

The $\surd$ symbol indicates the item with the best evaluation determined by CBF

2.6 Constraint-Based Recommendation for Groups

Taking into account groups in constraint-based recommendation [18] requires the extension of our definition of a recommendation task, as given in Chap. 1.

Definition (Recommendation Task for Groups) A recommendation task for groups can be defined by the tuple (G, $R = R_1 \cup .. \cup R_m$, I) where $G = \{u_1, u_2, .., u_m\}$ represents a group of users, $R_j = \{r_{1j}, r_{2j}, \ldots, r_{nj}\}$ represents a set of requirements (r_{ij} denotes the requirement i of group member j), and $I = \{t_1, .., t_k\}$ represents a set of items. The goal is to identify items in I which fulfill all requirements in R. A solution for a recommendation task can be defined as follows.

Definition (Recommendation Task for Groups—Solution) A solution for a recommendation task for groups (G, R, I) is a set $S \subseteq I$ such that $\forall t_i \in S : t_i \in \sigma_{[R]} I$ where σ is the selection operator of a conjunctive query, R represents requirements defined by group members, and I represents a collection of items.

In group recommendation settings, each group member should specify his/her *requirements* (in our example, these are hard constraints related to season and topics) and *preferences* (*weights* or *soft constraints*) with regard to a set of *interest dimensions* (in our example, *security*, *attractiveness*, and *crowdedness*)—see Table 2.12. Requirements are constraints that are used to pre-select items, preferences specify weights that are used to rank the pre-selected items.

In both scenarios, i.e., *aggregated predictions* and *aggregated models*, group members have to define their requirements and preferences.

Aggregated Predictions We will first show how to handle aggregated predictions in constraint-based recommendation for groups (see Fig. 2.7).

A constraint-based recommender derives user-specific recommendations (items and user-specific item utilities) on the basis of a set of requirements and preferences. Item utilities for specific group members can be determined with multi-attribute utility theory (MAUT) [60, 62] (see Formula 1.4). For example, on the basis of

Table 2.12 User-specific *requirements* and *preferences* (*weights*)

	Requirements		Preferences (weights)		
User	*Season*	*Topics*	*Security*	*Attractiveness*	*Crowdedness*
u_1	r_{11} :spring	–	0.5	0.4	0.1
u_2	r_{12} :spring	r_{22} :citytours	0.2	0.7	0.1
u_3	–	r_{13} :entertainment	0.3	0.3	0.4
u_4	r_{14} :spring	–	0.6	0.2	0.2
u_5	–	r_{15} :citytours	0.1	0.8	0.1

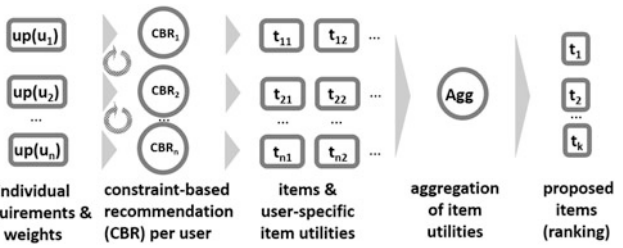


Fig. 2.7 Constraint-based recommendation for groups based on *aggregated predictions*. User preferences are constructed iteratively (conversational recommendation approach). Item t_{ij} represents item j (including corresponding item utilities) determined by recommender i

the user requirements defined in Table 2.12 and the example itemset of Table 1.8, the utility of item t_1 for user u_1 can be determined as follows: $utility(u_1, t_1) = \sum_{d \in Dimensions} contribution(t_1, d) \times weight(u_1, d) = contribution(t_1, security) \times weight(u_1, security) + contribution(t_1, attractiveness) \times weight(u_1, attractiveness) + contribution(t_1, crowdedness) \times weight(u_1, crowdedness) = 5.0 \times 0.5 + 5.0 \times 0.4 + 2.0 \times 0.1 = 2.5 + 2.0 + 0.2 = 4.7$. These user-specific item utilities are aggregated into a group recommendation (see Table 2.13).

If an entry of item t_i in *user-specific item utilities* in Table 2.13 > 0, this indicates that the item t_i fulfills all requirements of the corresponding group member. In contrast, table entries $= 0$ are used to indicate that an item does not completely fulfill the requirements of a group member. For example, the requirements of u_2 ($\{r_{12}, r_{22}\}$) are not completely fulfilled by t_2 (r_{22} : $topics = citytours$ is not supported). Even if an item does not completely fulfill the requirements of some users, it could be recommended. The lower the number of users with completely fulfilled requirements with regard to a specific item t_i, the lower the probability that t_i will be recommended. A set of individual user requirements can also be inconsistent with the effect that no fitting item can be identified. In such a case, diagnosis methods can help to guide the user out of the *no solution could be found dilemma* [20].[13]

[13] Issues related to conflict resolution will be discussed at the end of this section.

Table 2.13 User-specific item utilities (and corresponding *scores* used by *BRC*) with regard to *security*, *attractiveness*, and *crowdedness* determined by utility analysis (see Chap. 1)

	Item contribution			User-specific item utilities (scores)					Aggregation		
Item	*secur.*	*attr.*	*crowd.*	u_1	u_2	u_3	u_4	u_5	AVG	BRC	LMS
t_1	5.0	5.0	2.0	4.7(9)	4.7(9)	3.8(9)	4.4(9)	4.7(9)	4.46 $\surd$	45.0 $\surd$	3.8 $\surd$
t_2	4.0	4.0	4.0	4(7.5)	0.0	0.0	4(7)	0.0	1.6	14.5	0.0
t_3	3.0	5.0	1.0	3.6(4.5)	4.2(7.5)	2.8(7.5)	3(4)	4.4(7.5)	3.6	31.0	2.8
t_4	4.0	3.0	5.0	3.7(6)	0.0	0.0	4(7)	0.0	1.54	13.0	0.0
t_5	3.0	4.0	1.0	3.2(3)	3.5(6)	2.5(6)	2.8(2)	3.6(6)	3.12	23.0	2.5
t_6	3.0	3.0	1.0	2.8(1)	2.8(3.5)	2.2(5)	2.6(1)	2.8(3)	2.64	13.5	2.2
t_7	2.0	3.0	3.0	2.5(0)	2.8(3.5)	0.0	2.4(0)	2.9(4)	2.12	7.5	0.0
t_8	4.0	4.0	4.0	4(7.5)	0.0	0.0	4(7)	0.0	1.6	14.5	0.0
t_9	3.0	5.0	1.0	3.6(4.5)	4.2(7.5)	2.8(7.5)	3(4)	4.4(7.5)	3.6	31.0	2.8
t_{10}	3.0	3.0	3.0	3(2)	3(5)	0.0	3(4)	3(5)	2.4	16.0	3.0

The $\surd$ symbol indicates the item with the best evaluation

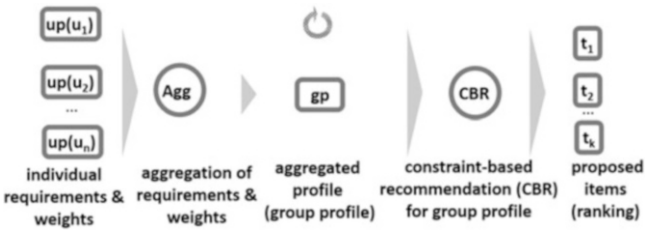


Fig. 2.8 Constraint-based recommendation for groups based on *aggregated models*. Group preferences are constructed iteratively (conversational recommendation)

Also in constraint-based recommendation, an alternative to the aggregation of user $\times$ item utilities (Table 2.13) is to *aggregate items* proposed by individual recommenders. If we want to generate a recommendation based on the *Fairness* (FAI) aggregation strategy and 5 is the upper bound for the number of proposed items, each group member would choose his/her favorite item (not already selected by another group member). In the example shown in Table 2.13, t_1 has the highest utility for user u_1, it also has the highest utility for user u_2, however, since u_1 already selected t_1, u_2 has to identify a different one, which is now t_3. Furthermore, we assume that u_3 selects t_9, u_4 selects t_8, and user u_5 selects t_5. The group recommendation resulting from this aggregation step is $\{t_1, t_3, t_5, t_8, t_9\}$.

Aggregated Models Another possibility of determining recommendations for groups in constraint-based recommendation scenarios is to first aggregate individual user preferences [33] (requirements and weights related to interest dimensions) into a *group profile gp* and then to determine recommendations (see Fig. 2.8).

The construction of a group profile *gp* is sketched in Table 2.14. Beside aggregating the user requirements $R = \{r_{11}, r_{12}, r_{22}, r_{13}, r_{14}, r_{15}\}$, we also have

Table 2.14 Construction of a group profile (*gp*)

Weights and requirements	u_1	u_2	u_3	u_4	u_5	*gp*
Security	0.5	0.2	0.3	0.6	0.1	0.34 (AVG)
Attractiveness	0.4	0.7	0.3	0.2	0.8	0.48 (AVG)
Crowdedness	0.1	0.1	0.4	0.2	0.1	0.18 (AVG)
Season	r_{11}: spring	r_{12}: spring	–	r_{14}: spring	–	r_{11}, r_{12}, r_{14}
Topics	–	r_{22}: citytours	r_{13}: entertainment	–	r_{15}: citytours	r_{22}, r_{13}, r_{15}

User-specific weights regarding the interest dimensions *security*, *attractiveness*, and *crowdedness* are aggregated into *gp* using AVG. Furthermore, user requirements r_{ij} are combined into $R = \{r_{11}, r_{12}, r_{22}, r_{13}, r_{14}, r_{15}\}$

Table 2.15 Item utilities determined on the basis of the weights defined in *gp* (see Table 2.14)

Item	Item contribution			Utility(gp, t_i)
	secur.	*attr.*	*crowd.*	
t_1	5	5	2	4.46 √
t_2	4	4	4	4.0
t_3	3	5	1	3.6
t_4	4	3	5	3.7
t_5	3	4	1	3.12
t_6	3	3	1	2.64
t_7	2	3	3	2.66
t_8	4	4	4	4.0
t_9	3	5	1	3.6
t_{10}	3	3	3	3

Only items t_i are taken into account that are consistent with the requirements in *gp* (others are shown greyed out). The √ symbol indicates the item with the highest utility

to aggregate user preferences specified in terms of weights related to the interest dimensions *security*, *attractiveness*, and *crowdedness*.

On the basis of the requirements defined in *gp* and the item definitions in Table 1.8, a conjunctive query $\sigma_{[r_{11},r_{12},r_{22},r_{13},r_{14},r_{15}]}I$ results in: $\{t_1, t_3, t_5, t_6, t_7, t_9\}$, i.e., these items are consistent with the requirements defined in *gp*. Formula 2.4 can be used then to determine item-specific utilities on the basis of the group profile *gp*. For example, $utility(gp, t_1) = \sum_{d \in Dimensions} contribution(t_1, d) \times weight(gp, d) = contribution(t_1, security) \times weight(gp, security) + contribution(t_1, attractiveness) \times weight(gp, attractiveness) + contribution(t_1, crowdedness) \times weight(gp, crowdedness) = 5 \times 0.34 + 5 \times 0.48 + 2 \times 0.18 = 1.7 + 2.4 + 0.36 = 4.46$. The resulting utilities are shown in Table 2.15.

$$utility(gp, item) = \sum_{d \in Dimensions} contribution(item, d) \times weight(gp, d) \qquad (2.4)$$

It can be the case that a set of user requirements is *inconsistent with all items* of an itemset. In such a situation, users of a constraint-based recommender have to adapt their requirements such that at least one solution can be identified. Related techniques will be discussed in the following section.

2.7 Handling Inconsistencies

Since item retrieval in constraint-based recommendation is based on semantic queries (e.g., conjunctive queries), situations can occur where no solution can be identified for the given set of requirements [22], i.e., $\sigma_{[R]}I = \emptyset$ (R represents the union of requirements specified by individual group members and I represents the example itemset shown in Table 1.8). An example of such a situation is the following (adapted version of the examples introduced in the previous sections): $R = \{r_{11} : season = summer, r_{21} : eval = 5.0, r_{12} : season = summer, r_{13} : topics = entertainment, r_{14} : topics = entertainment, r_{15} : eval = 5.0\}$ where $\sigma_{[R]}I = \emptyset$. Also in the context of group recommendation scenarios, we are interested in how to change the requirements defined by group members in order to be able to come up with a recommendation consistent with the requirements of all group members.

In the *aggregated predictions* scenario, inconsistencies induced by requirements occur on the "single user" level: a user specifies his/her requirements but no recommendation can be identified (see Chap. 1). In this context, diagnosis algorithms help to identify possible changes to the user requirements such that a recommendation can be identified. This way, it can be guaranteed that no user-specific inconsistent requirements are passed to the group level.

In the *aggregated models* scenario, the task of resolving inconsistent situations is a similar one: in the case of inconsistencies between requirements defined by a specific group member, diagnosis (see Chap. 1) can actively support him/her in restoring consistency.[14] However, even if the requirements of a user profile are consistent, integrating the requirements of individual users into a group profile gp can induce inconsistencies on the group level [25]. In the aggregated models scenario, diagnosis also supports the achievement of *global consistency*, i.e., all joint preferences defined by individual group members allow the derivation of at least one solution. Table 2.16 shows the user requirements specified in our example.

The conflict sets induced by our example requirements (R) are: $CS_1 : \{r_{11}, r_{21}\}$, $CS_2 : \{r_{11}, r_{15}\}$, $CS_3 : \{r_{12}, r_{21}\}$, $CS_4 : \{r_{12}, r_{15}\}$, $CS_5 : \{r_{13}, r_{21}\}$, $CS_6 : \{r_{13}, r_{15}\}$, $CS_7 : \{r_{14}, r_{21}\}$, and $CS_8 : \{r_{14}, r_{15}\}$. If we resolve the conflicts by deleting the requirements r_{21} and r_{15}, a corresponding diagnosis (hitting set) $\Delta_1 = \{r_{21}, r_{15}\}$ can be identified. The second diagnosis is $\Delta_2 = \{r_{11}, r_{12}, r_{13}, r_{14}\}$. The determination of the diagnoses Δ_1 and Δ_2 is shown on the basis of the HSDAG approach (Hitting Set Directed Acyclic Graph) [55] (see Fig. 2.9). Table 2.16 includes a third

[14] A discussion of algorithms for diagnosis determination can be found in [19–21, 55].

Table 2.16 Example user requirements and related diagnoses in the *aggregated models* scenario (r_{ij} = requirement i of user j): $\Delta_1 = \{r_{21}, r_{15}\}$, $\Delta_2 = \{r_{11}, r_{12}, r_{13}, r_{14}\}$, and $\Delta_3 = \{r_{11}, r_{21}, r_{15}\}$

Requirement	Δ_i		
	Δ_1	Δ_2	Δ_3
$r_{11(season=0100)}$		•	•
$r_{21(eval=5.0)}$	•		•
$r_{12(season=0100)}$		•	
$r_{13(topic=entertainment)}$		•	
$r_{14(topic=entertainment)}$		•	
$r_{15(eval=5.0)}$	•		•

Δ_3 is a non-minimal diagnosis included to show that aggregation functions prefer minimal diagnoses

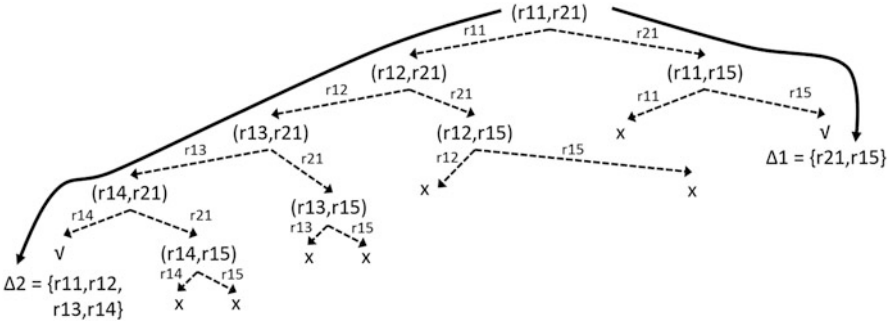


Fig. 2.9 Determination of the minimal diagnoses Δ_1 and Δ_2 using the HSDAG approach [55] (paths to minimal diagnoses are denoted with $\surd$)

diagnosis ($\Delta_3 = \{r_{11}, r_{21}, r_{15}\}$) which has been included to show that non-minimal diagnoses $\Delta_{\neg min}$ are not preferred by aggregation functions (see Tables 2.17 and 2.18). A corresponding subset ($\Delta \subset \Delta_{\neg min}$) exists that already fulfills the diagnosis properties. In our example, $\Delta_1 \subset \Delta_3$ holds, i.e., Δ_3 is a non-minimal diagnosis.

As different diagnosis candidates exist ($\Delta_1, \Delta_2, \Delta_3$), we have to figure out which one should be recommended to the group. Similar to the determination of recommendations, diagnosis candidates can be ranked on the basis of different aggregation functions. In Table 2.17 we sketch an approach to rank diagnoses depending on the number of requirements that have to be deleted/adapted by individual group members. Diagnosis Δ_1 has the *lowest number of needed changes* (*ADD*); consequently it can be recommended. *Least Misery* (LMS) recommends one out of $\{\Delta_1, \Delta_2\}$. As mentioned, we will not discuss diagnosis algorithms in this chapter; for a detailed discussion of diagnosis search and selection in group contexts, we refer to [25].

Diagnosis ranking can be better personalized, if we assume that requirements have importance weights learned, for example, on the basis of previous group decisions [20, 29]. Table 2.18 depicts an example of the determination of diagnosis

Table 2.17 Diagnosis recommendation in the *aggregated models* scenario based on (1) counting the needed changes per user and (2) *LMS*

	Changes per user					Aggregation	
Diagnosis	u_1	u_2	u_3	u_4	u_5	*ADD*	*LMS*
Δ_1	1	0	0	0	1	2 $\surd$	1 $\surd$
Δ_2	1	1	1	1	0	4	1 $\surd$
Δ_3	2	0	0	0	1	3	2

The $\surd$ symbol indicates recommended diagnosis candidates

Table 2.18 Utility-based diagnosis recommendation in the *aggregated models* scenario

Δ_i	Weighted requirements						Aggregation	
	$w(r_{11}) = 0.1$	$w(r_{21}) = 0.3$	$w(r_{12}) = 0.1$	$w(r_{13}) = 0.1$	$q(r_{14})0.1$	$w(r_{15}) = 0.3$	*utility*	*LMS*
Δ_1	0	0.3	0	0	0	0.3	1.67	0.3
Δ_2	0.1	0.0	0.1	0.1	0.1	0	2.5 $\surd$	0.1 $\surd$
Δ_3	0.1	0.3	0	0	0	0.3	1.42	0.3

The $\surd$ symbol indicates the highest rated diagnosis

utilities on the basis of weighted requirements—the utility of a diagnosis can be determined on the basis of Formula 2.5. That implements an additive aggregation strategy: the higher the sum of the individual weights $w(r_{ij})$, the higher the importance of the related requirements for the group members. Consequently, the lower the total importance of the included requirements, the higher the utility of the corresponding diagnosis (see Formula 2.5). In this setting, diagnosis Δ_2 outperforms Δ_1 (also Δ_3) since Δ_2 includes requirements less relevant for the individual group members. *Least Misery* (LMS) in this context analyzes (user-wise) attribute-specific estimated negative impacts of requirement deletions.

$$utility(\Delta) = \frac{1}{\sum_{r_{ij} \in \Delta} w(r_{ij})} \tag{2.5}$$

Remark An issue for future work in this context is to analyze the possibility of combining the group profile (*gp*) with local user profiles. This could serve to assure consensus in the group earlier, and avoid efforts related to conflict resolution on the group level. If parts of the group profile are integrated into individual user profiles, this could also help to take into account the requirements of other group members at the very beginning of the decision making process. Further details on how to determine personalized diagnoses on the basis of search heuristics can also be found in [20, 23].

2.8 Critiquing-Based Recommendation for Groups

Critiquing-based recommendation [9, 30] is based on the idea of showing *reference items* to users and allowing users to give feedback in terms of *critiques*. Critiques trigger a new critiquing cycle where *candidate items* (items that fulfill the critiques defined by the user[15]) are compared with regard to their utility as a new *reference item*. This utility is evaluated on the basis of (a) *similarity metrics* (see Chap. 1) that estimate the *similarity* between a reference item and a candidate item and (b) the degree of *support* of the critiques already defined by a user.[16] Intuitively, the more similar a candidate item is with regard to the reference item and the more critiques it supports, the higher its utility. In the following, we assume that the determination of candidate items for a specific group member takes into account his/her previous critiques and the similarity between reference and candidate item. The utility of a candidate item as the next reference item can be determined on the basis of Formulae 2.6–2.8. In this context, $utility(c, r, u)$ denotes the utility of a candidate item c to act as a reference item for user u taking into account the current reference item r. Furthermore, $sim(c, r)$ determines the similarity between r and c. Finally, $support(c, critiques(u))$ evaluates the support candidate item c provides for the critiques defined by user u. In this context, *support* is measured in terms of (a) *consistency* between candidate item and critiques and (b) the *weight* of individual critiques (for example, older critiques could have a lower weight).

$$utility(c, r, u) = sim(c, r) \times support(c, critiques(u)) \tag{2.6}$$

$$support(c, critiques) = \Sigma_{crit \in critiques} consistent(c, crit) \times weight(crit) \tag{2.7}$$

$$consistent(c, crit) = \begin{cases} 1 & \text{if } \sigma_{[crit]}\{c\} \neq \emptyset \\ 0 & \text{otherwise} \end{cases} \tag{2.8}$$

Let us assume that the first reference item (item r that is the first one shown to start a critiquing session) shown to each group member is t_1. Table 2.19 depicts

Table 2.19 A group-based critiquing scenario: each group member already specified two critiques (denoted as *critiquing history*)

User	First critique	Second critique
u_1	t_1:winter ∈ season (cr_{11})	t_3:eval > 3.3 (cr_{12})
u_2	t_1:nature ∈ topics (cr_{21})	t_2:winter ∈ season (cr_{22})
u_3	t_1:eval > 4.5 (cr_{31})	t_4:citytours ∈ topics (cr_{32})

The reference item for the first critiquing cycle is assumed to be $t_1(u_1, u_2, u_3)$, the reference items for the second critiquing cycle are $t_3(u_1)$, $t_2(u_2)$, and $t_4(u_3)$

[15] Different variants thereof exist in critiquing-based systems ranging from *taking into account only the most recent critique* to *all critiques in the critiquing history* (see Chap. 1).

[16] Also denoted as *compatibility score* [47].

example critiques defined thereafter on t_1 by the group members u_1, u_2, and u_3. We also assume that items used in the example correspond to the travel destinations itemset shown in Table 1.8. Finally, we assume *equal weights* for critiques.

The similarities between potential combinations of reference items (r) and candidate items (c) are depicted in Table 2.20. The attributes *season* (EIB), *topics* (EIB), and *eval* (NIB) are taken into account.[17] For example, $\text{sim}(t_1,t_2) = \text{s}(t_1.season.spring, t_2.season.spring)\times\frac{1}{9} + \text{s}(t_1.season.summer, t_2.season.summer)\times\frac{1}{9} + \text{s}(t_1.season.autumn, t_2.season.autumn)\times\frac{1}{9} + \text{s}(t_1.season.winter, t_2.season.winter)\times\frac{1}{9} + \text{s}(t_1.topics.citytours, t_2.topics.citytours)\times\frac{1}{9} + \text{s}(t_1.topics.entertainment, t_2.topics.entertainment)\times\frac{1}{9} + \text{s}(t_1.topics.nature, t_2.topics.nature)\times\frac{1}{9} + \text{s}(t_1.topics.beach, t_2.topics.beach)\times\frac{1}{9} + \text{s}(t_1.eval, t_2.eval)\times\frac{1}{9} = 0.66$.

The selection of a new reference item in the critiquing scenario shown in Table 2.19 is depicted in Table 2.21. In this context, reference items are not considered potential candidate items, since the same item should not be presented in follow-up critiquing cycles. Each table entry represents the utility of a specific candidate item (from Table 1.8) with regard to a reference item. For example, $utility(c : t_2, r : t_3, u : u_1) = sim(t_2, t_3) \times support(t_2, critiques(u_1)) = 0.43 \times (0 \times 0.5 + 1 \times 0.5) = 0.21$ (two critiques, i.e., equal weights = 0.5).

Table 2.20 Items of Table 1.8 (similarity with regard to *season*, *topics*, and *eval*)

Item	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
t_1	1.0	0.66	0.75	0.32	0.86	0.88	0.54	0.61	0.74	0.77
t_2	–	1.0	0.43	0.64	0.53	0.54	0.67	0.96	0.42	0.64
t_3	–	–	1.0	0.52	0.88	0.86	0.54	0.42	0.99	0.74
t_4	–	–	–	1.0	0.4	0.44	0.53	0.6	0.51	0.56
t_5	–	–	–	–	1.0	0.96	0.42	0.53	0.89	0.84
t_6	–	–	–	–	–	1.0	0.43	0.5	0.85	0.88
t_7	–	–	–	–	–	–	1.0	0.62	0.53	0.53
t_8	–	–	–	–	–	–	–	1.0	0.42	0.6
t_9	–	–	–	–	–	–	–	–	1.0	0.73
t_{10}	–	–	–	–	–	–	–	–	–	1.0

Table 2.21 Selection of new reference items based on the *utility* of candidate items t_i (calculation is based on Formula 2.6)

		$utility(t_i, r, u)$									
u	r	t_1	t_2	t_3	t_4	t_5	t_6	t_7	t_8	t_9	t_{10}
u_1	t_3	–	0.21	–	0.52	0	0.43	0.54$\surd$	0	0.49	0.37
u_2	t_2	–	–	0.21	0.64	0	0	0.67$\surd$	0.48	0.21	0
u_3	t_4	–	0	0.26	–	0.2	0.44	0.26	0	0.25	0.56$\surd$

We assume that previous reference items are not reference item candidates anymore (represented by "–" entries). The $\surd$ symbol denotes the selected new reference items

[17] Similarity metrics introduced in Chap. 1—we assume, *minval*=0 and *maxval*=5.

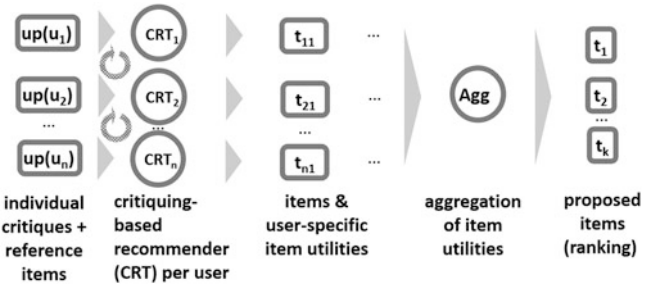


Fig. 2.10 Critiquing-based recommendation for groups with *aggregated predictions*. User preferences are constructed iteratively (conversational recommendation)

If a user interacts with a critiquing-based recommender in *standalone* mode (critiques of other users are not taken into account), he/she receives recommendations related to his/her preferences [47]. In parallel, critiques from individual group members can be forwarded to a group recommender. Different variants thereof are possible. For example, recommendations determined for a single user can also take into account the preferences of the whole group by simply taking into account some or all of the critiques stored in the group profile [47]. In this context, weights regarding the trade-offs between the importance of user-individual critiques and critiques on the group level have to be specified.

Aggregated Predictions The process of critiquing-based group recommendation using aggregated predictions is sketched in Fig. 2.10.

On the basis of an initial reference item, individual critiquing-based recommenders start the first critiquing cycle and—depending on user feedback—determine follow-up reference items. In other words, several interaction cycles precede a decision. After individual group members have completed their selection process, the corresponding results (see, e.g., Table 2.19) can be used to determine a group recommendation. Table 2.22 depicts user-specific utilities of new items (the similarity values are taken from Table 2.21).

An alternative to the aggregation of item utilities (Table 2.22) is to *aggregate items* proposed by individual critiquing-based recommenders. A group recommendation can be determined, for example, by taking the item with the highest utility value per group member (Formula 2.6). The group recommendation is $\{t_7, t_{10}\}$. As discussed in [30], items can be proposed by group members and group members can provide counter-proposals that—with some likelihood—are acceptable to other group members.

Aggregated Models Following this strategy, a group model (critiquing history on the group level) has to be generated (see Fig. 2.11).

On the basis of a group model (group profile—*gp*), a corresponding group recommendation can be determined. In order to build a group profile (*gp*), critiques defined by group members have to be aggregated. Table 2.23 depicts an example

Table 2.22 User-specific utilities of new items (see Formula 2.6)

	Utility(c,r,u) (score)			Aggregation		
c	$u_1(r:t_3)$	$u_2(r:t_2)$	$u_3(r:t_4)$	AVG	BRC	LMS
t_1	0 (1.5)	0 (2)	0 (1.5)	0	5	0
t_2	0.21 (4)	0 (2)	0 (1.5)	0.07	7.5	0
t_3	0 (1.5)	0.21 (5.5)	0.26 (6.5)	0.16	13.5	0
t_4	0.52 (8)	0.64 (8)	0 (1.5)	0.39	17.5	0
t_5	0 (1.5)	0 (2)	0.2 (4)	0.07	7.5	0
t_6	0.43 (6)	0 (2)	0.44 (8)	0.29	16	0
t_7	0.54 (9)	0.67 (9)	0.26 (6.5)	0.49 √	24.5 √	0.26 √
t_8	0 (1.5)	0.48 (7)	0 (1.5)	0.16	10	0
t_9	0.49 (7)	0.21 (5.5)	0.25 (5)	0.32	17.5	0.21
t_{10}	0.37 (5)	0 (2)	0.56 (9)	0.31	16	0

√ indicates the item with the best evaluation determined by the corresponding aggregation function

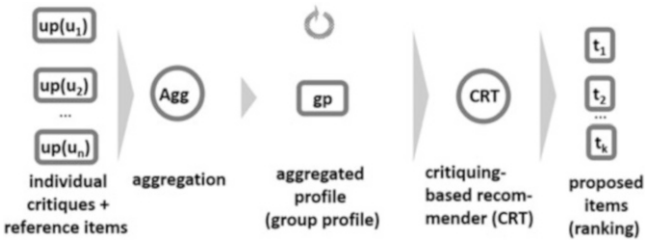


Fig. 2.11 Critiquing-based recommendation for groups with *aggregated models*. Group preferences are constructed iteratively (conversational recommendation)

Table 2.23 Set of critiques (=group profile *gp*) defined by the group $G = \{u_1, u_2, u_3\}$

Group G	Group profile (defined by critiques) of G
$\{u_1, u_2, u_3\}$	Winter ∈ season, nature ∈ topics, eval > 4.5, citytours ∈ topics

of the aggregation of group member specific critiquing histories into a group profile (*gp*). In this scenario, the aggregation of individual critiques can lead to a situation where none of the items completely fulfills the defined critiques (see the example group profile in Table 2.23). As a consequence, we have to identify recommendations which support as many critiques as possible. In order to determine a ranking for the different items, Formula 2.9 can be applied where $utility(t, gp)$ denotes the utility of item t with regard to the critiques part of the group profile gp, and *weight* represents the weight of a critique. In our example, we assume equal weights, however, weights can also be used to reduce the impact of less up-to-date critiques.

$$utility(t, gp) = \Sigma_{crit \in critiques(gp)} consistent(t, crit) \times weight(crit) \tag{2.9}$$

Table 2.24 Group-specific utilities of new items determined on the basis of Formula 2.9

Item	Utility
t_1	0.25
t_2	0.25
t_3	0.5
t_4	0.75 √
t_5	0.25
t_6	0.5
t_7	0.75 √
t_8	0.25
t_9	0.5
t_{10}	0.5

The √ symbol indicates items with the highest utility values

Table 2.24 represents a list of items (determined on the basis of Formula 2.9) and corresponding utilities with regard to the critiques contained in the group profile *gp*. For example, $utility(t_1, gp) = 0 \times 0.25 + 0 \times 0.25 + 0 \times 0.25 + 1 \times 0.25 = 0.25$.

2.9 Hybrid Recommendation for Groups

As already mentioned, hybrid recommendation helps to compensate specific limitations of one recommendation approach with the strengths of another one [8, 16]. In Chap. 1, we already took a look at different basic hybrid recommendation approaches. We will now sketch hybridization in the context of group recommender systems [3, 14, 15].

Weighted The idea of weighted hybrid recommendation is to combine the results received from individual recommenders into a corresponding group recommendation. Table 2.25 shows a simple example of applying weighted hybridization in the context of group recommendation. A collaborative recommender for groups (CF) based on the *aggregated models* (AM) strategy and a content-based filtering recommender (CBF) for groups based on the *aggregated predictions* (AP) strategy return the item rankings shown in Table 2.25. The *Borda Count* (BRC) strategy (see Table 2.2) can now be applied to aggregate the corresponding scores.

Mixed Hybrid recommendation based on the mixed strategy combines the recommended items returned by the individual recommenders (see Table 2.26).

In our example, the rankings returned by two group recommenders are aggregated using the fairness (FAI) function where items are included in the final recommendation following the zipper principle, i.e., the item ranked highest by the CBF recommender is integrated first, then the item ranked highest by the CF recommender is integrated into the recommendation result, and so on.

Table 2.25 Recommendation results of two group recommenders (CF based on *aggregated models* (AM) and CBF based on *aggregated predictions* (AP)) as list of ranked items are aggregated on the basis of *Borda Count* (BRC)

	Recommender-specific evaluations (scores)		Aggregation
Item	CF ratings (AM,AVG)	CBF similarities (AP,LMS)	BRC
t_1	4.9 (8)	0.81 (9)	17 $\surd$
t_2	2.2 (1)	0.32 (1)	2
t_3	5.0 (9)	0.66 (7)	16
t_4	4.3 (7)	0.61 (6)	13
t_5	1.5 (0)	0.2 (0)	0
t_6	3.8 (3)	0.55 (5)	8
t_7	3.4 (2)	0.49 (4)	6
t_8	4.1 (4)	0.45 (3)	7
t_9	4.2 (5.5)	0.33 (2)	7.5
t_{10}	4.2 (5.5)	0.79 (8)	13.5

The $\surd$ symbol indicates the item with the best evaluation

Table 2.26 Recommendation results of two group recommenders (CF and CBF) as a list of ranked items aggregated on the basis of *Fairness* (FAI) that implements the zipper principle (alternate inclusion of best ranked items—the item ranked highest by CBF is integrated first)

	Recommender-specific rankings		Aggregation
Item	CF (AM,AVG)	CBF (AP,LMS)	FAI (ranking)
t_1	10	9	10
t_2	2	1	1 $\surd$
t_3	7	6	7
t_4	6	5	6
t_5	1	4	2
t_6	3	7	4
t_7	5	3	5
t_8	9	8	9
t_9	4	2	3
t_{10}	8	10	8

$\surd$ indicates the item with the highest ranking

2.10 Matrix Factorization for Groups

Up to now, we have discussed ways to apply the recommendation approaches of collaborative filtering, content-based filtering, constraint-based, critiquing-based, and hybrid recommendation in group contexts. *Matrix factorization* is a popular approach to collaborative filtering based recommendations [39]. The underlying idea is to explain ratings by characterizing items and users on the basis of a set of factors. The original user $\times$ item matrix is separated into two lower-dimensional ones that explain user item interactions on the basis of the mentioned factors (see Table 2.27).

Table 2.27 We factorize the rating matrix $\boldsymbol{R}$ containing known ratings for $n = 8$ users and $m = 8$ items into matrices $\boldsymbol{P}$ and $\boldsymbol{Q}$ such that $\boldsymbol{P}\boldsymbol{Q}^T$ closely approximates $\boldsymbol{R}$

(a) Rating matrix $\boldsymbol{R}$

	i1	i2	i3	i4	i5	i6	i7	i8
u1	5	1		2	2			2
u2	1	4	2		5	1		4
u3			3	5		1	2	4
u4	4		5	3		5	3	
u5	4	1		1		4	3	
u6	4		1	1		5		1
u7		2	2		2		4	1
u8	4		3			4	3	

(b) User factors $\boldsymbol{P}$

	$p_{u,1}$	$p_{u,2}$	$p_{u,3}$
u1	0.27984223	1.33194126	−0.25748666
u2	−0.13639896	−1.00299326	1.09421098
u3	0.29145967	−1.08249042	0.85824434
u4	1.29398513	0.94631031	0.77574863
u5	0.25258519	0.72222304	−1.20079938
u6	−1.26795635	1.16829146	−0.11806872
u7	−0.82074846	0.92881098	−0.20635514
u8	0.15691092	0.64969789	0.53725284

(l) Item factors $\boldsymbol{Q}$

	$q_{i,1}$	$q_{i,2}$	$q_{i,3}$
i1	0.23698436	1.38151089	−0.23953783
i2	−0.19159424	−1.01718827	0.59249957
i3	1.60559024	0.21567611	−0.26351522
i4	0.3814163	−0.94038814	0.9485212
i5	0.43533279	−0.3900103	1.52692825
i6	0.1258419	1.88675619	0.13922563
i7	−0.47012841	0.65365324	0.03900384
i8	0.98047664	−0.63261944	0.51537004

For illustration purposes, we minimize the sum of the squared errors of the approximation together with a simple squared L2-norm regularization term. We set $k = 3$ factors and regularization parameter to $\lambda = 0.02$. We initialize $\boldsymbol{P}$ and $\boldsymbol{Q}$ randomly and optimize with the gradient descent algorithm. Note that factorization brings similar users close to each other in the factor space (c.f. factors of users $u2$ and $u3$ in $\boldsymbol{P}$), whereas dissimilar users are projected further apart (c.f. factors of users $u1$ and $u2$ in $\boldsymbol{P}$)

In this context, each item t is associated with a vector q_t that describes to which extent t represents the factors. Furthermore, each user u is associated with a vector p_u that describes to which extent the factors are important for the user. Finally $\hat{r}_{ui} = q_i^T p_u$ represents an approximation of a user's u rating of t (r_{ui} denotes a user's real rating). More formally, we factorize the rating matrix $\boldsymbol{R} \in \mathbb{R}^{n \times m}$ containing known ratings for n users and m items into matrices $\boldsymbol{P} \in \mathbb{R}^{n \times k}$ and $\boldsymbol{Q} \in \mathbb{R}^{m \times k}$ such that $\boldsymbol{P}\boldsymbol{Q}^T$ closely approximates $\boldsymbol{R}$. In literature and practice, there are several possibilities to measure and minimize the approximation error of the factorization. A popular choice for the approximation error is the sum of the squared errors combined with a simple regularization term, e.g. $\sum_{r_{u,i} \neq \bullet} (r_{u,i} - \mu - \mathbf{p}_u^T \mathbf{q}_i)^2 + \lambda(||\mathbf{p}_u||^2 + ||\mathbf{q}_i||^2)$, where μ is the global rating average and $\bullet$ represents an unknown rating. Minimization of the error is typically computed with a variant of the gradient descent method.

An approach to the application of matrix factorization in the context of group recommendation scenarios is presented in [51]. The authors introduce two basic strategies denoted as *After Factorization* (AF) and *Before Factorization* (BF). When using AF (see Table 2.28), user-individual matrix factorization is performed in order to identify user-specific factors which are thereafter aggregated (e.g., by determining the average (AVG) of the user-individual factor values). When using BF (see Table 2.29), first user-individual item ratings are aggregated into a group profile, followed by a matrix factorization approach. These two basic variants follow the idea of *aggregated predictions* (AF) and *aggregated models* (BF).

Table 2.28 In the *After Factorization* (AF) approach the group of users is factorized by merging factors of users (e.g., by calculating averages) in a given group

(a) AF: group factors

	$p_{G,1}$	$p_{G,2}$	$p_{G,3}$
G	0.144968	−0.25118	0.56499

(b) AF: predicted ratings

	i1	i2	i3	i4	i5	i6	i7	i8
G	2.40	3.41	2.88	3.68	3.87	2.47	2.64	3.44

In our example, we group three users from $G = \{u1, u2, u3\}$. Note that users $u2$ and $u3$ are highly similar to each other but are highly dissimilar to user $u1$. Thus, we expect the group ratings to be biased towards the ratings of users $u2$ and $u3$ as group ratings for items $i1$ (lower because of a low rating from user $u2$) and $i2$ (higher because of a high rating of user $u2$) show

Table 2.29 In the *Before Factorization* (BF) approach a virtual group user is created from the rating matrix by, e.g., calculating the average ratings (*AVG*) for the users from a given group

(a) BF: group factors

	$p_{G,1}$	$p_{G,2}$	$p_{G,3}$
G	0.12873	−0.56466	−0.03111

(b) BF: predicted ratings

	i1	i2	i3	i4	i5	i6	i7	i8
G	2.11	3.38	2.94	3.40	3.08	1.80	2.42	3.32

In the next step, the group factors are calculated from the given factorization by calculating the (Ridge) regression coefficients on the ratings of the virtual user. Finally, the group factors allow us to predict group ratings. The intuition behind BF approach is that the virtual user is a better representation of the users group than a simple aggregation of users factors. In our example, BF predicts a significantly lower rating than AF for item $i6$ because there is much stronger evidence in the data for a low rating (two 1-star ratings)

Due to its simplicity, AF is efficiently calculated and provides a solid baseline for group recommendation approaches based on matrix factorization. However, in practice BF gives significantly better prediction results on larger datasets and for larger groups. For more details on matrix factorization based recommendation approaches, we refer to [39]. Approaches to apply matrix factorization in the context of group recommendation scenarios are discussed in [32, 51].

2.11 Conclusions and Research Issues

In this chapter, we have introduced different group recommendation techniques which are based on the recommendation approaches for individual users introduced in Chap. 1. We showed how related group recommendation scenarios can be designed for collaborative filtering, content-based filtering, constraint-based including utility-based recommendation, critiquing-based, and hybrid recommendation. In this context, we focused on a discussion of the two aggregation strategies: (1) *aggregated predictions* (*items*) and (2) *aggregated models*. In (1), recommendations are determined for individual group members and then aggregated. In (2), the preferences of group members are aggregated, and recommendations are then determined on the basis of information contained in the integrated group profile. An issue already solved in a couple of *person-2-person* recommendation environments is which algorithms can be used to find a person that fits another person with regard to a set of predefined criteria. An online dating application is reported, for example, in [61]. Another application is the identification of experts to support the answering of specific questions [48]. A related issue, especially relevant in the context of group decision making, is *group synthesis*, i.e., the identification of a group that is able to solve a specific problem or to make a decision. Initial work on group synthesis in the context of open innovation scenarios can be found in [7, 31]. A major criteria is to identify a group that is able to solve a given (decision) task, taking into account availability aspects such as engagement in other projects. This scenario can become even more complex if we want to configure a set of groups to solve a specific task. Consider the following university-based task: Given that there are 300 students registered in a software engineering course, divide the population into groups of 6, such that each group is best suited to complete a specific project. A related issue is the analysis of inter-group influences, for example, in which way *influential groups* influence *susceptible groups* [54]. Further research issues are related to the topics of *evaluating group recommenders*, *explaining group recommendations*, *taking into account group dynamics*, and *counteracting biases* that trigger suboptimal decisions. These issues will be discussed in the following chapters of this book.

References

1. K. Arrow, The difficulty in the concept of social welfare. J. Polit. Econ. **58**(4), 328–346 (1950)
2. L. Baltrunas, T. Makcinskas, F. Ricci, Group recommendations with rank aggregation and collaborative filtering, in *4th ACM Conference on Recommender Systems*, Barcelona, 2010, pp. 119–126
3. S. Berkovsky, J. Freyne, Group-based recipe recommendations: analysis of data aggregation strategies, in *4th ACM Conference on Recommender Systems*, Barcelona, 2010, pp. 111–118
4. S. Berkovsky, J. Freyne, M. Coombe, D. Bhandari, Recommender algorithms in activity motivating games, in *ACM Conference on Recommender Systems (RecSys'10)*, 2010, pp. 175–182
5. L. Boratto, S. Carta, The rating prediction task in a group recommender system that automatically detects groups: architectures, algorithms, and performance evaluation. J. Intell. Inf. Syst. **45**(2), 221–245 (2015)
6. L. Boratto, S. Carta, G. Fenu, Investigating the role of the rating prediction task in granularity-based group recommender systems and big data scenarios. Inf. Sci. **378**, 424–443 (2017)
7. M. Brocco, G. Groh, Team recommendation in open innovation networks, in *ACM Conference on Recommender Systems (RecSys'09)*, NY, 2009, pp. 365–368
8. R. Burke, Hybrid recommender systems: survey and experiments. User Model. User Adap. Inter. **12**(4), 331–370 (2002)
9. L. Chen, P. Pu, Critiquing-based recommenders: survey and emerging trends. User Model. User-Adap. Inter. **22**(1–2), 125–150 (2012)
10. L. Chen, G. Chen, F. Wang, Recommender systems based on user reviews: the state of the art. User Model. User Adap. Inter. **25**(2), 99–154 (2015)
11. Y. Chevaleyre, U. Endriss, J. Lang, N. Maudet, A short introduction to computational social choice, in *33rd Conference on Current Trends in Theory and Practice of Computer Science*, Harrachov, 2007, pp. 51–69
12. K. Christakopoulou, F. Radlinski, K. Hofmann, Towards conversational recommender systems, in *International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, San Francisco, CA, 2016, pp. 815–824
13. T. DePessemier, S. Dooms, L. Martens, An improved data aggregation strategy for group recommenders, in *3rd Workshop on Human Decision Making and Recommender Systems (Held in Conjunction with the 7th ACM Conference on Recommender Systems)*, Hong Kong, 2013, pp. 36–39
14. T. DePessemier, S. Dooms, L. Martens, Comparison of group recommendation algorithms. Multimedia Tools Appl. **72**(3), 2497–2541 (2014)
15. T. DePessemier, J. Dhondt, K. Vanhecke, L. Martens, TravelWithFriends: a hybrid group recommender system for travel destinations, in *9th ACM Conference on Recommender Systems, Workshop on Tourism Recommender Systems*, 2015, pp. 51–60
16. T. DePessemier, J. Dhondt, L. Martens, Hybrid group recommendations for a travel service. Multimedia Tools Appl. **76**(2), 2787–2811 (2017)
17. M. Ekstrand, J. Riedl, J. Konstan, Collaborative filtering recommender systems. Found. Trends Hum. Comput. Inter. **4**(2), 81–173 (2011)
18. A. Felfernig, R. Burke, Constraint-based recommender systems: technologies and research issues, in *ACM International Conference on Electronic Commerce (ICEC08)*, Innsbruck, 2008, pp. 17–26
19. A. Felfernig, G. Friedrich, D. Jannach, M. Stumptner, Consistency-based diagnosis of configuration knowledge bases. Artif. Intell. **152**(2), 213–234 (2004)
20. A. Felfernig, M. Schubert, G. Friedrich, M. Mandl, M. Mairitsch, E. Teppan, Plausible repairs for inconsistent requirements, in *21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, CA, 2009, pp. 791–796

21. A. Felfernig, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger, F. Reinfrank, Group decision support for requirements negotiation. Springer Lect. Notes Comput. Sci. **7138**, 105–116 (2011)
22. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Design Anal. Manuf. **26**(1), 53–62 (2012)
23. A. Felfernig, M. Schubert, S. Reiterer, Personalized diagnosis for over-constrained problems, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, Peking, 2013, pp. 1990–1996
24. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, Toward the next generation of recommender systems, in *Multimedia Services in Intelligent Environments: Recommendation Services* (Springer, Berlin, 2013), pp. 81–98
25. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)*, 2016, pp. 69–72
26. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, S. Polat-Erdeniz, An analysis of group recommendation heuristics for high- and low-involvement items, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*, Arras, 2017, pp. 335–344
27. H. Garcia-Molina, G. Koutrika, A. Parameswaran, Information seeking: convergence of search, recommendations, and advertising. Commun. ACM **54**(11), 121–130 (2011)
28. S. Ghazarian, M. Nematbakhsh, Enhancing memory-based collaborative filtering for group recommender systems. Expert Syst. Appl. **42**(7), 3801–3812 (2015)
29. J. Guo, L. Sun, W. Li, T. Yu, Applying uncertainty theory to group recommender systems taking account of experts preferences. Multimedia Tools Appl. 1–18 (2017). https://doi.org/10.1007/s11042-017-4922-4
30. F. Guzzi, F. Ricci, R. Burke, Interactive multi-party critiquing for group recommendation, in *5th ACM Conference on Recommender Systems*, Chicago, IL, 2011, pp. 265–268
31. S. Hong, C. Mao, Z. Yang, H. Lai, A new team recommendation model with applications in social network, in *18th IEEE International Conference on Computer Supported Cooperative Work in Design (CSCWD)*, NY, 2014, pp. 644–648
32. X. Hu, X. Meng, L. Wang, SVD-based group recommendation approaches: an experimental study of Moviepilot, in *ACM Recommender Systems 2011 Challenge on Context-aware Movie Recommendation*, 2011, pp. 23–28
33. A. Jameson, More than the sum of its members: challenges for group recommender systems, in *International Working Conference on Advanced Visual Interfaces*, 2004, pp. 48–54
34. A. Jameson, B. Smyth, Recommendation to groups, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (Springer, Berlin, 2007), pp. 596–627
35. A. Jameson, S. Baldes, T. Kleinbauer, Two methods for enhancing mutual awareness in a group recommender system, in *ACM International Working Conference on Advanced Visual Interfaces*, Gallipoli, 2004, pp. 447–449
36. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in *Recommender Systems Handbook*, 2nd edn., ed. by F. Ricci, L. Rokach, B. Shapira (Springer, Berlin, 2015), pp. 611–648
37. M. Kompan, M. Bielikova, Group recommendations: survey and perspectives. Comput. Inf. **33**(2), 446–476 (2014)
38. J. Konstan, B. Miller, D. Maltz, J. Herlocker, L. Gordon, J. Riedl, GroupLens: applying collaborative filtering to usenet news. Commun. ACM **40**(3), 77–87 (1997)
39. Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems. IEEE Comput. **42**(8), 30–37 (2009)
40. J. Levin, B. Nalebuff, An introduction to vote-counting schemes. J. Econ. Perspect. **9**(1), 3–26 (1995)
41. G. Linden, B. Smith, J. York, Amazon.com recommendations – item-to-item collaborative filtering. IEEE Int. Comput. **7**(1), 76–80 (2003)

42. T. Mahmood, F. Ricci, Improving recommender systems with adaptive conversational strategies, in *20th ACM Conference on Hypertext and Hypermedia*, Torino, 2009, pp. 73–82
43. J. Marquez, J. Ziegler, Preference elicitation and negotiation in a group recommender systems, in *Interact 2015*. Lecture Notes in Computer Science, vol. 9297 (Springer, Berlin, 2015), pp. 20–37
44. J. Masthoff, Group modeling: selecting a sequence of television items to suit a group of viewers. User Model. User Adap. Inter. **14**(1), 37–85 (2004)
45. J. Masthoff, Group recommender systems: combining individual models, in *Recommender Systems Handbook* (Springer, Berlin, 2011), pp. 677–702
46. J. Masthoff, Group recommender systems: aggregation, satisfaction and group attributes, in *Recommender Systems Handbook* (Springer, Berlin, 2015), pp. 743–776
47. K. McCarthy, L. McGinty, B. Smyth, M. Salamó, Social interaction in the CATS group recommender, in *Workshop on the Social Navigation and Community Based Adaptation Technologies*, 2006, pp. 743–776
48. D. McDonald, M. Ackerman, Expertise recommender: a flexible recommendation system and architecture, in *Conference on Computer Support Cooperative Work*, Philadelphia, PA, 2000, pp. 231–240
49. T. Nguyen, Conversational group recommender systems, in *International Conference on User Modelling, Adaptation and Personalization (UMAP'17)* (ACM, New York, 2017), pp. 331–334
50. T. Nguyen, F. Ricci, A chat-based group recommender system for tourism, in *Information and Communication Technology in Tourism*, ed. by R. Schegg, B. Stangl (Springer, Berlin, 2017), pp. 17–30
51. F. Ortega, A. Hernando, J. Bobadilla, J.H. Kang, Recommending items to group of users using matrix factorization based collaborative filtering. Inf. Sci. **345**(C), 313–324 (2016)
52. D. Pennock, E. Horvitz, C. Giles, Social choice theory and recommender systems: analysis of the axiomatic foundations of collaborative filtering, in *17th National Conference on Artificial Intelligence (AAAI)*, Austin, TX, 2000, pp. 729–734
53. L. Quijano-Sánchez, D. Bridge, B. Díaz-Agudo, J. Recio-García, A case-based solution to the cold-start problem in group recommenders, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, 2013, pp. 3042–3046
54. L. Recalde, A social framework for set recommendation in group recommender systems, in *European Conference on Information Retrieval* (Springer, Berlin, 2017), pp. 735–743
55. R. Reiter, A theory of diagnosis from first principles. AI J. **32**(1), 57–95 (1987)
56. D. Sacharidis, Group recommendations by learning rating behavior, in *International Conference on User Modelling, Adaptation and Personalization (UMAP'17)* (ACM, New York, 2017), pp. 174–182
57. C. Senot, D. Kostadinov, M. Bouzid, Gerome Picault, A. Aghasaryan, C. Bernier, Analysis of strategies for building group profiles, in *Conference on User Modeling, Adaptation, and Personalization (UMAP 2010)*, Big Island, HI. Lecture Notes in Computer Science, vol. 6075 (2010), pp. 40–51
58. C. Senot, D. Kostadinov, M. Bouzid, J. Picault, A. Aghasaryan, Evaluation of group profiling strategies, in *IJCAI 2011*, 2011, pp. 2728–2733
59. M. Stettinger, A. Felfernig, CHOICLA: intelligent decision support for groups of users in context of personnel decisions, in *ACM RecSys'2014 IntRS Workshop*, Foster City, CA, 2014, pp. 28–32
60. D. Winterfeldt, W. Edwards, *Decision Analysis and Behavioral Research* (Cambridge University Press, Cambridge, 1986)
61. W. Wobcke, A. Krzywicki, Y. Kim, X. Cai, M. Bain, P. Compton, A. Mahidadia, A deployed people-to-people recommender system in online dating. AI Mag. **36**(3), 5–18 (2015)
62. Z. Yu, X. Zhou, Y. Hao, J. Gu, TV program recommendation for multiple viewers based on user profile merging. User Model. User Adap. Inter. **16**(1), 63–82 (2006)

Chapter 3
Evaluating Group Recommender Systems

Christoph Trattner, Alan Said,
Ludovico Boratto, and Alexander Felfernig

Abstract In the previous chapters, we have learned how to design group recommender systems but did not explicitly discuss how to evaluate them. The evaluation techniques for group recommender systems are often the same or similar to those that are used for single user recommenders. We show how to apply these techniques on the basis of examples and introduce evaluation approaches that are specifically useful in group recommendation scenarios.

3.1 Introduction

Evaluating group recommenders is intrinsically related to evaluation techniques used for single user recommenders [10, 15]. There are two types of *evaluation protocols*: (1) *offline* and (2) *online evaluation*.

Offline evaluation is based on the idea of estimating the prediction quality of an algorithm using *datasets* that include user × item evaluations (ratings). These datasets are typically divided into *training* and *test sets* with a split of, for example, 80% training data and 20% test data. Such settings are used for the evaluation of a recommendation algorithm in the light of given *evaluation metrics* on the basis of repeated sampling and cross validation [4, 8]. Since datasets are typically derived from recommender systems for individual users [14], *datasets for groups* have to be *synthesized* to be applicable to the evaluation of group recommenders [2].

Online evaluation is based on the idea of using user study techniques to evaluate an algorithm, a user interface, or a whole system online [16]. Over the past few years, this approach has lagged behind offline evaluation, due to higher efforts and the lack of standardized evaluation frameworks [16, 19]. *Lab studies* (as one type of online evaluation) involve the recruitment of study participants who are then engaged in tasks based on two kinds of designs: (1) *within-subjects* study design (each subject is assigned to a *set of conditions*) or (2) *between-subjects*

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_3

study design (each subject is assigned to *exactly one condition*). Online evaluations in the form of lab studies in the context of group recommender systems have been conducted, for example, by Zapata et al. [30], De Pessemier et al. [7], Masthoff et al. [17], and Stettinger et al. [25]. As an alternative to lab studies, which are often quite costly, recent research in recommender systems started to use *crowd-sourcing platforms* as a source of user feedback [6, 28]. A major challenge in this context is to assure the understanding of the defined tasks and to include quality assurance mechanisms to avoid low-quality feedback [1]. Finally, *naturalistic studies* often employ some kind of *A/B testing* where the system is used *as is* without any interventions or predefined tasks. A/B in this context refers to different user populations unknowingly assigned to different system versions in which some condition has been changed. Naturalistic studies in the context of group recommender systems have been conducted, for example, in Sanchez et al. [20].

Independent of the type of online evaluation protocol used, quantitative post-hoc analysis is typically employed to identify differences between interfaces, algorithms, and systems. Apart from the standard evaluation metrics (as discussed in the following), metrics such as *number of clicks*, *time needed to complete a task*, and *dwell time* are also employed to measure system efficiency. Based on the exact evaluation protocol defined, a set of recommendation metrics can then be used to estimate the performance of the recommender system.

In the following, we *first* focus on *accuracy metrics* which compare recommendations determined by a recommender system with a predefined set of real-world user opinions (also known as *ground truth*).[1] Depending on the underlying goal, accuracy can be measured on the basis of: (1) *classification metrics* that evaluate to which extent a recommender is able to determine items of relevance (interest) for the user, (2) *error metrics* that evaluate how well a recommender predicts ratings, and (3) *ranking metrics* that help to evaluate how well a recommender predicts the importance ranking of items. *Second*, we discuss a couple of *group recommendation-related metrics* that go beyond accuracy measurement.

3.2 Classification Metrics

Arguably, the most common classification metrics used in recommender systems are *precision* and *recall*. These metrics are often applied in *offline* evaluation scenarios where recommendation algorithms are trained using a portion of the available data for learning purposes and are then evaluated by comparing predictions to a withheld part of the data ("holdouts" constituting the test set). In the following, we will briefly explain the usage of precision and recall metrics in group recommendation scenarios. Table 3.1 contains (1) *user rating data* (evaluations of items already consumed by the members of groups g_1, g_2, and g_3) where each group consists of

[1] For an in-depth discussion of evaluation metrics for single user recommenders, we refer to Gunawardana and Shani [24].

Table 3.1 *Ratings* $r(u_i, t_j)$ and *predictions* $\hat{r}(u_i, t_j)$ for items t_1 and t_2

Groups	Group members	Ratings $r(u_i, t_j)$			Predictions $\hat{r}(u_i, t_j)$		
		t_1	t_2	...	t_1	t_2	...
g_1	u_1	4.5	2.5	...	3.4	3.8	...
	u_2	3.5	4.5	...	3.7	4.4	...
	u_3	4.5	4.0	...	4.4	3.9	...
g_2	u_4	3.5	2.5	...	3.8	2.6	...
	u_5	4.0	4.5	...	3.7	4.4	...
	u_6	4.5	3.5	...	4.5	3.7	...
g_3	u_7	4.5	3.5	...	3.4	3.8	...
	u_8	3.5	2.5	...	3.7	4.4	...
	u_9	4.0	3.5	...	4.4	3.9	...
...	...	...	...	...	...	...	...

Table 3.2 Example *test set*: *group ratings* $r(g_i, t_j)$ and *group predictions* $\hat{r}(g_i, t_j)$ where $r(g,t) = \frac{\sum_{u \in g} r(u,t)}{|g|}$ and $\hat{r}(g,t) = \frac{\sum_{u \in g} \hat{r}(u,t)}{|g|}$

Groups	Ratings $r(g_i, t_j)$		Predictions $\hat{r}(g_i, t_j)$	
	t_1	t_2	t_1	t_2
g_1	4.2	3.7	3.8	4.0
g_2	4.0	3.5	4.0	3.6
g_3	4.0	3.2	3.8	4.0

For groups g_1, g_2, and g_3, the item ratings of t_1 and t_2 are considered as elements (holdouts) of the test set

three users, and (2) *predictions of item ratings* (for items t_1 and t_2). For simplicity, we assume that each group member provided a rating for each item consumed by her/him.

The user-individual ratings and predictions are aggregated into (1) a *group rating* $r(g_i, t_j)$ and (2) corresponding *group predictions* $\hat{r}(g_i, t_j)$ determined by an *aggregated predictions* based group recommender system (see Table 3.2).[2] In a typical group recommendation scenario, a random set of *group-level item ratings* is withheld and used as test set. In our example, we assume for simplicity that for groups g_1, g_2, and g_3, the ratings for item t_1 and t_2 have been selected as "holdouts". The rating predictions $\hat{r}(g_i, t_j)$ (assumed to be provided by a group recommender) are depicted in the two rightmost columns of Table 3.2.

On the basis of the entries in Tables 3.1 and 3.2, we will now sketch the application of the classification metrics *precision* and *recall*.

Precision is the fraction of the number of *relevant recommended items* (true positives) in relation to the *total number of recommended items*. *Recall* is the fraction of the number of *relevant recommended items* in relation to the *number of all relevant items*. Both metrics are commonly expressed at a certain level k where k is the length of the list of recommended items. For example, $precision@1 = 1$ indicates that one item was recommended and this item was deemed to be a relevant

[2] Predictions are determined using rating data not used as test cases.

Table 3.3 *Precision* and *recall* values in example scenario

Groups	Predicted	Relevant	precision@2	recall@2
g_1	2	2	1.0	1.0
g_2	2	1	0.5	1.0
g_3	2	1	0.5	1.0
overall	6	4	0.67	1.0

recommendation. Furthermore, $precision@2 = 0.5$ denotes a situation where in a list of two recommended items only one is deemed to be a relevant recommendation (true positive). The precision of a group recommender that recommends k items to a group g can be defined as follows where $predicted_k(g)$ denotes a list of k items recommended to group g and $relevant(g)$ represents all items relevant for g. Definitions of *precision* and *recall* are given in Formulae 3.1 and 3.2.

$$precision@k(g) = \frac{|predicted_k(g) \cap relevant(g)|}{k} \tag{3.1}$$

$$recall@k(g) = \frac{|predicted_k(g) \cap relevant(g)|}{|relevant(g)|} \tag{3.2}$$

The calculation of precision and recall is sketched in Table 3.3—it is based on the test dataset defined in Table 3.2. For the purpose of our example, we define an item to be relevant, if the corresponding group rating > 3.5 (the *relevance threshold*). For example, $precision@2(g_1) = 1$ since both items predicted to be relevant (rating $>$ 3.5) are deemed as relevant by group g_1. The overall precision and recall values for the *test set* (see Table 3.2) are determined by integrating the group-specific prediction and relevance counts, for example, six predicted items of four correctly predicted items result in an overall precision of 0.67 (see Table 3.3).

Precision and *recall* can be used in *aggregated predictions* as well as in *aggregated models* based group recommenders (see Chap. 2) since both evaluation metrics are applied to the recommendation result, i.e., are independent from the underlying aggregation approach. The same holds for *content-based*, *constraint-based*, and *critiquing-based recommendation*. In these scenarios, the group recommender determines an overall item evaluation (similarity between user and item in content-based filtering, user-specific item utility in constraint-based recommendation, and similarity between candidate and reference item in critiquing-based recommendation) which is then used for estimating item relevance. Consequently, a threshold similar to the one used in our example can be applied.

As will be discussed in Chap. 7, some group recommenders operate on item *packages* and *parameters* [11, 12]. Package recommendations can be evaluated similarly to single item recommendations—a difference in this regard is that package items get recommended at the same point of time whereas single items are recommended at different points of time. Recommendations of configurations [11] (see Chap. 7) consist of parameter settings related to requirements of a single user (or a group). In this context, *precision* can be defined as the share of *correctly*

Table 3.4 Relevant (indicated by √) and predicted (indicated by √) parameter values in a group-based configuration scenario

Group g	Relevant	Predicted
$par_1 = a$	√	√
$par_1 = b$	×	×
$par_1 = c$	×	×
$par_2 = 1$	√	×
$par_2 = 2$	×	√
$par_3 = a$	√	×
$par_3 = b$	×	√
$par_4 = u$	√	√
$par_4 = v$	√	×
$par_4 = w$	×	×

par_4 is assumed to be multi-valued, i.e., can have more than one value at a time

predicted parameter values compared to the *total number of predicted parameter values*. Furthermore, *recall* can be defined as the share of *correctly predicted parameter values* compared to the *total number of relevant parameter values* (see Formulae 3.3 and 3.4).

$$precision(g) = \frac{|predictedvals(g) \cap relevantvals(g)|}{|predictedvals(g)|} \tag{3.3}$$

$$recall(g) = \frac{|predictedvals(g) \cap relevantvals(g)|}{|relevantvals(g)|} \tag{3.4}$$

An example of the calculation of *precision* and *recall* for group g in a group-based configuration scenario is sketched in Table 3.4. In this example, $precision = \frac{2}{4} = 0.5$ (2 correct predictions out of 4) and $recall = \frac{2}{5} = 0.4$. Our assumption is that parameter values for parameters par_1 .. par_4 have been predicted by a (group) recommender system.

Remark Note that this approach to determine *precision* and *recall* can also be applied to evaluate the predictive quality of diagnosis algorithms [9] (see Chaps. 1 and 2). In this case, *precision* can be regarded as the share of correctly predicted diagnoses compared to the total number of predictions. Likewise, *recall* is the share of correctly predicted diagnoses compared to the total number of relevant ones.

3.3 Error Metrics

Error metrics can be used to measure the error made by a recommender system to predict a rating of an item. The underlying assumption is that the smaller the error, the better the evaluated algorithm. A basic means of measuring prediction errors

Table 3.5 Mean absolute error (MAE) values determined on the basis of the rating information included in Table 3.2

Groups	MAE
g_1	$\frac{0.4+0.3}{2} = 0.35$
g_2	$\frac{0.0+0.1}{2} = 0.05$
g_3	$\frac{0.2+0.8}{2} = 0.5$
overall(*AVG*)	0.3

Table 3.6 Determining MAE in configuration scenarios (measuring the distance between predicted parameter values and those regarded as relevant ones)

Group g_1	Relevant	Predicted	MAE
$par_1(1,2,3)$	1	3	$\|1-3\| = 2$
$par_2(1,2)$	2	2	$\|2-2\| = 0$
$par_3(1,2,3,4)$	2	3	$\|2-3\| = 1$
overall(*AVG*)	–	–	1.0

is *mean absolute error* (*MAE*) (see Formula 3.5). A detailed discussion of error metrics is given, for example, in Shani and Gunawardana [24]. In Formula 3.5, R_g denotes the set of ratings of group g contained in the test set (see Table 3.2).

$$MAE(g) = \frac{\Sigma_{r(g,t)\in R_g} |r(g,t) - \hat{r}(g,t)|}{|R_g|} \tag{3.5}$$

The determination of the *MAE* value for the rating predictions in the test set shown in Table 3.2 is depicted in Table 3.5. The overall *MAE* value for a test set can be determined by averaging group-specific *MAE* values.

Similar to *precision* and *recall*, *MAE* can be used in the context of *aggregated predictions* and *aggregated models* based group recommenders. Given a function that estimates user × item ratings, this metric can also be applied in *content-based*, *constraint-based*, and *critiquing-based recommendation*. Furthermore, MAE can be applied to scenarios such as *package recommendation* and *group-based configuration* (see Chap. 7). An example of determining *MAE* in *configuration scenarios* is given in Table 3.6. For simplicity, we assume that parameter values are numeric.[3]

Remark Note that the usage of these metrics in the context of recommendation scenarios has declined as other types of metrics such as classification-based methods started to dominate. Recommendation is often interpreted as ranking problem.

3.4 Ranking Metrics

Ranking-dependent metrics do not only take into account item relevance but also the item position in a recommendation list. An example of such a metric is *discounted cumulative gain* (*DCG*) which is based on the idea that items appearing

[3]For a discussion of the handling of symbolic parameter values, we refer to [26].

Table 3.7 Example application of *discounted cumulative gain* (*DCG*)

Groups	*relevance*		
	$pos_1 : t_1$	$pos_2 : t_2$	$DCG@k$
g_1	1	1	$\frac{1}{1} + \frac{1}{1.6} = 1.625$
g_2	1	0	$\frac{1}{1} + \frac{0}{1.6} = 1$
g_3	1	0	$\frac{1}{1} + \frac{0}{1.6} = 1$
Overall $DCG(AVG)$			1.21

lower in a recommendation result should be penalized by downgrading relevance values logarithmically (see Formula 3.6). In Formula 3.6, k denotes the number of recommended items and $relevance(t_i, g)$ returns 1 if item t_i (at position i) is relevant for group g, and 0 otherwise.

$$DCG@k(g) = \Sigma_{i=1..k} \frac{2^{relevance(t_i,g)} - 1}{log_2(1+i)} \tag{3.6}$$

An example of the application of discounted cumulative gain (*DCG*) is provided in Table 3.7. The relevance values are derived from the group rating values of Table 3.2 ($relevance = 1$ if $r(g,t) > 3.5$, 0 otherwise). The more relevant items are included at the beginning of a list of k recommended items, the higher the *DCG* value. Since *DCG* operates on lists of ranked items, it can be applied to *collaborative filtering* as well as *content-based*, *constraint-based*, and *critiquing-based recommendation*. The overall *DCG* value for a test set is based on averaging group-specific *DCG* values.

If the length of recommendation lists for groups vary, i.e., there is no fixed k that reflects the number of recommended items, *DCG* has to be normalized by setting *DCG* in relation to the *ideal discounted cumulative gain* (*iDCG*)—see Formula 3.7. The resulting value is used to determine the *normalized discounted cumulative gain* (*nDCG*)—see Formula 3.8.

$$iDCG@k = \Sigma_{i=1..k} \frac{1}{log_2(1+i)} \tag{3.7}$$

$$nDCG@k(g) = \frac{DCG@k(g)}{iDCG@k} \tag{3.8}$$

In line with the previously discussed evaluation metrics, *DCG* can be used in the context of *aggregated predictions* as well as in the context of *aggregated models*-based group recommendation. *DCG* can also be applied to *package recommendation* (see Chap. 7) by evaluating the predictive quality with regard to different item types, and by aggregating type-individual *DCG* values into an overall *DCG* value.

When evaluating *sequences of recommended items*, not only the position of the item (the later the worse) but also the position of the item compared to the position selected by the group, is of relevance. In this context, a simple approach is to compare items at individual positions in the sequence of recommended items

Table 3.8 Example application of *Kendall's* τ to sequence evaluation (RS = recommended sequence, CS = chosen sequence)

Position	1	2	3	4	5
RS	t_4	t_2	t_5	t_1	t_3
CS	t_1	t_2	t_5	t_3	t_4

with the sequence chosen by a group (k denotes the length of the sequence). Such an evaluation can be performed, for example, on the basis of Kendall's τ (see Formula 3.9).

$$\tau(g) = \frac{|concordantpairs| - |discordantpairs|}{k \times \frac{(k-1)}{2}} \tag{3.9}$$

An example of the application of Formula 3.9 is depicted in Table 3.8. A recommended item sequence (*RS*) gets compared with an item sequence finally chosen by a group (CS)—the ground truth. In this example, four concordances are against six discordances. A concordant pair is (t_1, t_3) since these items are mentioned in the same order in both sequences (*RS* and *CS*). In contrast, (t_2, t_4) is an example of a discordant pair since the order of mentioning differs (t_4 before t_2 in *RS* and t_2 before t_4 in *CS*). Consequently, $\tau = -0.2$ (on a scale of -1 .. $+1$).

3.5 Coverage and Serendipity

In the context of recommender systems, *coverage* can be considered from different points of view—see, for example, Ge et al. [13]. *User coverage* can be interpreted as the number of users for whom at least one recommendation can be determined. For group recommendation, we introduce the term *group coverage* (GC) which represents the share of groups for whom at least one group recommendation could be identified (see Formula 3.10). No recommendations for a group can be determined in situations where, for example, the aggregated item ratings are below a certain threshold (collaborative filtering), or where all the given group requirements do not allow a recommendation (which could be the case in constraint-based recommendation scenarios).

$$GC = \frac{|groupswithprediction|}{|groups|} \tag{3.10}$$

Catalog coverage (CC) serves the purpose of analyzing which items from a catalog get recommended to users (groups). It represents the share of items that

were recommended to users (groups) at least once, compared to the total number of items contained in the item catalog (see Formula 3.11).

$$CC = \frac{|recommendeditems|}{|catalogitems|} \tag{3.11}$$

Serendipity, in the context of recommender systems, is defined as something surprising or unexpected that a user might not have seen before. A corresponding measure of *serendipity* (SER) is proposed in Ge et al. [13] (see also Formula 3.12). In this context, $RS(g)$ denotes the (useful) recommendations for group g determined by a group recommender system and $PM(g)$ denotes the recommendations generated by a primitive prediction model (e.g., based on item popularity). The overall *SER* value can be derived by averaging group-specific *SER* values.

$$SER(g) = \frac{|RS(g) - PM(g)|}{|RS(g)|} \tag{3.12}$$

3.6 Consensus and Fairness

Consensus can be regarded as a measure that evaluates to which extent group members have established an agreement with regard to their item preferences—see, for example, [5, 23]. In collaborative filtering, consensus can be measured in terms of the pairwise distances between the item-t ratings $r(u_i, t)$ of the individual group members u_i (see Formula 3.13) where r_{max} (the maximum rating possible) is used as normalization factor.

$$consensus(g, t) = 1 - \frac{\Sigma_{(u_i,u_j)\in g(i\neq j)}|r(u_i, t) - r(u_j, t)|}{|g| \times (|g| - 1)/2 \times r_{max}} \tag{3.13}$$

If rating information is available, the same measure can be applied in *content-based*, *constraint-based*, and *critiquing-based recommendation*. In conversational recommendation, i.e., in constraint-based or critiquing-based recommendation, group members are engaged in an interactive process where they define and refine their preferences. In constraint-based recommendation, group members define their preferences as requirements whereas in critiquing-based recommendation critiques are used to represent preferences. In both cases, preferences can become inconsistent and have to be adapted so that a recommendation can be identified. In the context of conversational recommendation, we define consensus as the share of pairwise agreements (e.g., equal parameter value selections) between group members in relation to the total number of pairwise agreements and disagreements (conflicts) (see Formula 3.14).

$$consensus(g) = \frac{\#agreements(g)}{\#agreements(g) + \#conflicts(g)} \tag{3.14}$$

Table 3.9 Example: determining *consensus* in conversational recommendation

Parameters	Users			Agreements	Disagreements	Consensus
	u_1	u_2	u_3			
$par_1(1,2)$	1	1	1	3	0	–
$par_2(1,2,3)$	1	2	1	1	2	–
$par_3(a,b)$	a	a	b	1	2	–
overall	–	–	–	5	4	0.56

A simple example of evaluating the degree of consensus in conversational recommendation scenarios is shown in Table 3.9. In this example, we count the pairwise agreements and disagreements between $\{u_1, u_2, u_3\} = g$. The total number of disagreements is 4 and the total number of agreements is 5. Following Formula 3.14, the consensus level in this example is $\frac{5}{5+4} = 0.56$ (on a scale 0..1).

Since group recommender systems involve multiple stakeholders, they can give rise to *fairness* issues [3]. Burke [3] introduces the concept of *multi-sided fairness* where different stakeholder groups have different interests that should somehow be balanced. In such a context, fairness can be considered as *the extent of imbalance between group member specific utilities* [29]. If single items are recommended to a group by collaborative filtering, fairness can be specified, for example, on the basis of the share of item ratings above a relevance threshold *th* (see Formula 3.15).

$$fairness(g,t) = \frac{|\bigcup_{u \in g} : r(u,t) > th|}{|g|} \tag{3.15}$$

If we evaluate the t_1 ratings of group g_1 in Table 3.1 on the basis of Formula 3.15 assuming $th = 3.5$, the overall degree of fairness with regard to item t_1 is $\frac{2}{3} = 0.66$ (on a scale of 0..1). The fairness interpretation of Formula 3.15 primarily considers situations where single items are recommended to groups, i.e., this metric does not take into account situations where packages are recommended to groups [23]. Alternative definitions of *fairness* are the following.[4]

First, *m-proportionality* (see Formula 3.16) interprets fairness as the share of group members u_i with at least m items in the recommended (or selected) package for which u_i has a high preference [23]. In this context, g_p denotes the set of users for whom the m-proportionality condition holds.

$$fairness_{m-prop}(g) = \frac{|g_p|}{|g|} \tag{3.16}$$

[4] See also Chap. 6.

Table 3.10 Evaluating fairness based on m-proportionality (m-prop) where $m = 2$

	Item ratings				
g_1	t_1	t_2	t_3	Rating threshold > 3.5	m-prop
u_1	5.0	4.0	1.0	2	1
u_2	4.0	3.0	2.0	1	0
u_3	4.5	5.0	5.0	3	1
u_4	3.5	3.0	5.0	1	0
$fairness_{m-prop}$	–	–	–	–	$\frac{2}{4} = 0.5$

Table 3.11 Fairness based on m-envy-freeness (m-envy) where $m = 1$ and $x = 25\%$

	Item ratings			
g_1	t_1	t_2	t_3	m-envy
u_1	5.0	4.0	1.0	1
u_2	4.0	3.0	2.0	0
u_3	4.0	5.0	5.0	1
u_4	3.0	3.0	5.0	1
$fairness_{m-envy}$	–	–	–	$\frac{3}{4}$

An example ($m = 2$) of the calculation of a fairness estimate following the m-proportionality criteria is given in Table 3.10. The g_p value in this example is 2 since two users (u_1, u_3) each evaluated two items above the threshold rating of 3.5.

Second, *m-envy-freeness* (see Formula 3.17) interprets fairness as the share of group members u_i with at least m items for which u_i is in the top $x\%$ item ratings. If this condition does not hold, the user feels envy towards other group members. In this context, g_{ef} denotes the users for whom m-envy-freeness holds [23].

$$fairness_{m-envy}(g) = \frac{|g_{ef}|}{|g|} \tag{3.17}$$

An example of the calculation of a fairness estimate following the m-envy-freeness criteria ($m = 1$ and $x = 25\%$) is given in Table 3.11. In this example, the g_{ef} value is 3 since u_1, u_3, and u_4 are group members of group g_1 with at least one item each for which they are in the top 25% of the item ratings.

The same approach to evaluate the fairness of recommendations proposed by a group recommender system can be applied in the context of *content-based*, *constraint-based*, and *critiquing-based recommendation*.

3.7 Conclusions and Research Issues

With a specific focus on group recommender systems, we have provided an overview of evaluation techniques. We have learned that the evaluation of group recommender systems can often be accomplished by employing standard evaluation

approaches from single user recommender systems [24]. We want to emphasize that there are several other metrics that one might want to consider when evaluating group recommender systems. Examples thereof are *trust*, *privacy*, and *performance*. Importantly, the usefulness of some evaluation metrics also depends on the item domain. For example, food recommender systems are not only following the goals of accuracy but also other criteria such as healthiness [22, 27]. An open research issue in the context of group recommender systems but also single-user recommender systems are evaluation metrics for complex, for example, configurable items. In this chapter, we have provided a couple of examples of metrics for complex items, but a more in-depth analysis and provision of corresponding metrics is an issue for future research. Synthesis approaches to generate groups sound like a promising and "cheap" alternative to studies with real groups. Often, clustering approaches are applied to derive groups from single user datasets—see, for example, Baltrunas et al. [2]. Group synthesis can also be based on analyzing social networks where social ties can serve as an indicator for group membership [18]. However, it should be mentioned that such approaches are based on simulations and should not replace controlled lab-studies, crowd-sourcing studies, or naturalistic online tests. For an overview of datasets related to single user recommender systems and existing software frameworks that can serve as a basis for developing group recommender systems, we refer to Said et al. [21].

References

1. M. Allahbakhsh, B. Benatallah, A. Ignjatovic, H. Motahari-Nezhad, E. Bertino, S. Dustdar, Quality control in crowdsourcing systems: issues and directions. IEEE Int. Comput. **17**(12), 76–81 (2013)
2. L. Baltrunas, T. Makcinskas, F. Ricci, Group recommendations with rank aggregation and collaborative filtering, in *4th ACM Conference on Recommender Systems*, Barcelona, 2010, pp. 119–126
3. R. Burke, Multisided fairness for recommendation, in *2017 Workshop on Fairness, Accountability, and Transparency in Machine Learning (FAT/ML 2017)*, 2017, pp. 1–5
4. P. Campos, F. Díez, I. Cantador, Time-aware recommender systems: a comprehensive survey and analysis of existing evaluation protocols. User Model. User Adap. Inter. **24**(1–2), 67–119 (2014)
5. J. Castro, F. Quesada, I. Palomares, L. Martínez, A consensus-driven group recommender system. Intell. Syst. **30**(8), 887–906 (2015)
6. S. Chang, F. Harper, L. He, L. Terveen, CrowdLens: experimenting with crowd-powered recommendation and explanation, in *10th International AAAI Conference on Web and Social Media (ICWSM'16)* (AAAI, Menlo Park, 2016), pp. 52–61
7. T. DePessemier, J. Dhondt, L. Martens, Hybrid group recommendations for a travel service. Multimedia Tools Appl. **76**(2), 2787–2811 (2017)
8. W. Dubitzky, M. Granzow, D. Berrar, *Fundamentals of Data Mining in Genomics and Proteomics* (Springer, Berlin, 2007)
9. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Design Anal. Manufact. **26**(1), 53–62 (2012)
10. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems, in *Recommendation Systems in Software Engineering* (Springer, Berlin, 2013), pp. 15–37

11. A. Felfernig, L. Hotz, C. Bagley, J. Tiihonen, *Knowledge-Based Configuration: From Research to Business Cases*, 1st edn. (Elsevier/Morgan Kaufmann, Amsterdam/Waltham, 2014)
12. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)* (2016), pp. 69–72
13. M. Ge, C. Delgado-Battenfeld, D. Jannach, Beyond accuracy: evaluating recommender systems by coverage and serendipity, in *4th ACM Conference on Recommender Systems*, Barcelona, 2010, pp. 257–260
14. F. Harper, J. Konstan, The MovieLens Datasets: history and context. ACM Trans. Interactive Intell. Syst. **5**(4), 19 (2015)
15. J. Herlocker, J. Konstan, L. Terveen, J. Riedl, Evaluating collaborative filtering recommender systems. ACM Trans. Inf. Syst. **22**(1), 5–53 (2004)
16. B. Knijnenburg, M. Willemsen, Evaluating recommender systems with user experiments, in *Recommender Systems Handbook*, ed. by F. Ricci, L. Rokach, B. Shapira (Springer, Berlin, 2015), pp. 309–352
17. J. Masthoff, A. Gatt, In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model. User Adap. Inter. **16**(3–4), 281–319 (2006)
18. A. Mislove, B. Viswanath, K. Gummadi, P. Druschel, You are who you know: inferring user profiles in online social networks, in *3rd ACM Conference on Web Search and Data Mining (WSDM'10)*, New York, 2010, pp. 251–260
19. D. Parra, S. Sahebi, Recommender systems: sources of knowledge and evaluation metrics, in *Advanced Techniques in Web Intelligence-2: Web User Browsing Behaviour and Preference Analysis* (Springer, Berlin, 2013), pp. 149–175
20. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, HAPPY MOVIE: a group recommender application in Facebook, in *24th International Florida Artificial Intelligence Research Society Conference*, Palm Beach, FL, 2011, pp. 419–420
21. A. Said, D. Tikk, P. Cremonesi, Benchmarking – a methodology for ensuring the relative quality of recommendation systems in software engineering, in *Recommender Systems in Software Engineering* (Springer, Berlin, 2014), pp. 275–300
22. H. Schaefer, S. Hors-Fraile, R. Karumur, A. Valdez, A. Said, H. Torkamaan, H. Ulmer, C. Trattner, Towards health (aware) recommender systems, in *DH 2017* (2017)
23. D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, P. Tsaparas, Fairness in package-to-group recommendations, in *WWW'17* (ACM, New York, 2017), pp. 371–379
24. G. Shani, A. Gunawardana, Evaluating recommender systems, in *Recommender Systems Handbook*, ed. by F. Ricci, L. Rokach, B. Shapira, P. Kantor (Springer, Berlin, 2011), pp. 257–297
25. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, Counteracting anchoring effects in group decision making, in *23rd Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*, Dublin. Lecture Notes in Computer Science, vol. 9146 (2015), pp. 118–130
26. J. Tiihonen, A. Felfernig, Towards recommending configurable offerings. Int. J. Mass Custom. **3**(4), 389–406 (2010)
27. C. Trattner, D. Elsweiler, Investigating the healthiness of internet-sourced recipes: implications for mean planning and recommender systems, in *26th International Conference on the World Wide Web* (2017), pp. 489–498
28. T. Ulz, M. Schwarz, A. Felfernig, S. Haas, A. Shehadeh, S. Reiterer, M. Stettinger, Human computation for constraint-based recommenders. J. Intell. Inf. Syst. **49**(1), 37–57 (2017)
29. L. Xiao, Z. Min, Z. Yongfeng, G. Zhaoquan, L. Yiqun, M. Shaoping, Fairness-aware group recommendation with pareto-efficiency, in *ACM Conference on Recommender Systems (RecSys'17)* (ACM, New York, 2017), pp. 107–115
30. A. Zapata, V. Menéndez, M. Prieto, C. Romero, Evaluation and selection of group recommendation strategies for collaborative searching of learning objects. Int. J. Hum. Comput. Stud. **76**, 22–39 (2015)

Part II
Group Recommender User Interfaces

Part II of this book focuses on *group recommender user interfaces* and related topics. In Chap. 4, we provide an overview of existing systems that are based on group recommendation technologies. Thereafter, we introduce preferences as a means of guiding the interaction with group recommenders, and determining recommendations for groups (Chap. 5). In Chap. 6, we discuss aspects especially relevant in the context of explaining recommendations to groups.

Chapter 4
Group Recommender Applications

Alexander Felfernig, Müslüm Atas, Martin Stettinger, Thi Ngoc Trang Tran, and Stefan Reiterer

Abstract In this chapter, we present an overview of different group recommender applications. We organize this overview into the application domains of music, movies and TV programs, travel destinations and events, news and web pages, healthy living, software engineering, and domain-independent recommenders. Each application is analyzed with regard to the characteristics of group recommenders as introduced in Chap. 2.

4.1 Introduction

Table 4.1 depicts an overview of example group recommender applications. Compared to single user recommenders [8], group recommenders are a relatively young research field. There are less commercial applications, and research prototype implementations still dominate. In the following, we will discuss the systems included in Table 4.1. In this context, we take into account (as far as possible) the criteria introduced in Table 2.1.

4.2 Music Recommendation

ADAPTIVERADIO [4] is a content-based song recommendation environment for groups. A specialty of this environment is that specifically *negative preferences* of users are taken into account in song recommendations. The underlying idea is that it is often easier to figure out what a user does not like than discovering what a user likes. If there is a need to find solutions (recommendations) that satisfy all group members, such an approach appears to be beneficial [4]. *Consensus solutions* are items that have been implicitly or explicitly approved by every group member

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_4

Table 4.1 Overview of existing group recommender systems—extended version of the overview introduced by Jameson and Smyth [13] (*CF* = collaborative recommendation, *CON* = content-based recommendation, *UTIL* = utility-based recommendation, *CRIT* = critiquing-based recommendation; *P* = aggregated profile, *I* = aggregated items or ratings)

System name	Item domain	Users	Recommendation	References
ADAPTIVE RADIO	Songs	Groups interested in hearing songs	CON (P)	[4]
CATS	Skiing vacation	Groups of friends planning a skiing vacation	CRIT (I,P)	[20]
CHOICLA-(WEB)	Domain-independent	Groups interested in completing a decision task	CON, UTIL (P)	[33, 34] www.choicla.com www.choicla-web.com
DOODLE	Domain-independent	Groups interested in scheduling a meeting	UTIL (P)	[30] www.doodle.com
EVENTHELPR	Tourist destinations, meetings	Groups interested in visiting tourist destinations and jointly defining a meeting agenda	UTIL (I)	www.eventhelpr.com
FLYTRAP	Songs	Groups interested in hearing songs	CON (P)	[6]
G.A.I.N	News	News adapted to different groups of users in a public space	CON (P)	[25]
GROUPFUN	Songs	Users interested in listening to songs	CF (I)	[26, 27]
GROUPLINK	Events	Group members searching events for face-2-face interactions	CON (I)	[36]
GROUP MODELER	Museum items	Groups jointly visiting a museum	CON (P)	[15]
GROUP-STREAMER	Songs	Users interested in listening to songs	CF(I)	Google Playstore
HAPPY MOVIE	Movies	Groups interested in movie recommendations	CON (I)	[28, 29]
INTELLIREQ	Requirements negotiation	Stakeholders interested in prioritizing software requirements	CON, UTIL (I)	[23] www.intellireq.org
IN-VEHIC. MM.-REC.	MM items, e.g., songs	Passengers interested in hearing music	CON (P)	[38]
I-SPY	Webpages	Company employees	CF (P)	[32]
INTRIGUE	Sightseeing destinations & Itineraries	Tourist groups interested in sightseeing destinations	UTIL (P)	[1]
JXGROUP-RECOM-MENDER	Songs and movies	Groups interested in watching movies and hearing songs	CF (I)	[5]

(continued)

Table 4.1 (continued)

System name	Item domain	Users	Recommendation	References
LET'S BROWSE	Web pages	Users interested in joint browsing	CON (P)	[16]
MUSICFX	Songs (genres)	Groups interested in hearing songs	CON (P)	[19]
NETFLIX GROUP REC.	Movies	Groups interested in watching movies	CF (I)	[3]
PLANIT-POKER	Effort estimates	Groups interested in effort estimation	CON (I)	www.planit-poker.com
POCKET RESTAU-RANT FINDER	Restaurants	Groups planning for a joint dinner	UTIL (I)	[18]
POLYLENS	Movies	Groups interested in watching movies	CF (I)	[24]
TRAVEL DEC. FOR.	Hotels	Groups of friends planning a holiday trip	CON (P)	[12]
XEET	Sports events	Persons interested in participating in sports events	CON (P)	

(in terms of ratings). ADAPTIVERADIO is an environment that broadcasts songs to a group and allows the group members to give feedback on individual songs in terms of *dislikes*. For a specific group, those songs that are not disliked by one of the group members are recommendation candidates. That is, if some group members dislike a song, it is filtered out. A basic similarity metric that primarily considers songs from the same album as similar is used.

FLYTRAP [6] is an environment that designs soundtracks for groups. Radio frequency ID badges let the environment know when users are nearby. The recommendation approach is content-based combined with a voting (aggregation) schema that is followed by user-specific automated agents. The system exploits knowledge about user preferences (e.g., in terms of genres) and the relationship between different song evaluation dimensions represented in terms of MP3 meta-information (for example, how different artists influence each other or what types of transitions between songs users prefer). In FLYTRAP, user preference data is derived from information about individual song preferences by a FLYTRAP AGENT locally installed on a user's computer.

GROUPFUN [26, 27] is a group recommender application implemented as a FACEBOOK plugin that recommends playlists for specific events, for example, birthday parties. In GROUPFUN, the playlists of individual users can be aggregated and recommendations for a specific event are determined on the basis of an advanced aggregation function denoted as *probabilistic weighted sum*, where the probability of a song being played is derived from a song's global popularity (represented in terms of a score). The GROUPSTREAMER system originates from the same research group as GROUPFUN and is currently available as an app in the Google Playstore.

IN-VEHICLE MULTIMEDIA RECOMMENDER [38] is a system that recommends multimedia items to a group of passengers. User profiles are exchanged via devices used during a car trip. The system aggregates relevant features from individual user profiles into a central profile that is used for determining recommendations. In the context of music recommendations, features could be general topics such as music styles, but could also be names of performers. Features are assigned a corresponding weight which reflects the importance of a feature for the whole group. Those items (e.g., songs and movies) with the highest overall similarity to the group profile have the highest probability of being recommended. No social-choice-based aggregation functions (see, e.g., [17]) are used in this context.

JXGROUPRECOMMENDER [5] suggests music and movies to groups. Two basic group recommenders are proposed—one supports the group-based recommendation of songs (JMUSICGROUPRECOMMENDER), the other one supports the recommendation of movies (JMOVIESGROUPRECOMMENDER). Both recommenders are based on the idea of merging the recommendations predetermined for group individuals. The aggregation of individual recommendations is based on the aggregation strategies discussed in Chap. 2.

MUSICFX [19] is a music recommendation system to be applied in the context of music consumption in fitness centers—more specifically, the system was built to be applied in the Andersen Consulting Technology Park (ACTP), where the fitness center has about 600 members. User preference information in MUSICFX is collected when members fill out an enrollment form upon first joining the fitness center. Preferences are specified on a rating scale [–2/I love this music .. +2/I hate this music] with regard to musical genres such as alternative rock, country, dance, and hits. MUSICFX then operates on the genre-level, i.e., does not recommend specific songs. The higher the aggregated popularity of a specific music genre for a group, the higher the probability that a song related to this genre will be selected.

4.3 Recommendation of Movies and TV Programs

HAPPY MOVIE [29] is a FACEBOOK application that supports the recommendation of movies to groups. A user profile in HAPPY MOVIE is based on the dimensions *personality*, *individual user preferences*, and *trust*. Personality information is derived from feedback on a personality questionnaire. Individual users' preferences with regard to movies are collected by ratings users have to provide before applying the system. Finally, trust information is collected from each FACEBOOK user profile. The recommendation approach integrated in HAPPY MOVIE is a so-called *delegation-based method* where the recommendations of a user's friend represent recommendation candidates. Their relevance is increased or decreased depending on the personality of the friend and finally weighted depending on the level of trust. For the different recommendation candidates, different preference aggregation functions [17] can be used to determine a final recommendation.

POLYLENS [24] is a collaborative filtering based prototype system that recommends movies to groups based on the individual preferences of users. It is an extension to the freely available MOVIELENS recommender system. POLYLENS users are allowed to create groups, receive group invitations, and also receive group-specific movie recommendations. *Least Misery* is used as aggregation strategy [17] to determine recommendations relevant to a group.

NETFLIX GROUP RECOMMENDER [3] is a collaborative-filtering-based prototype system that supports the recommendation of movies to groups of users. Predicted ratings are aggregated into a corresponding group rating by determining an average rating. Standard deviation values help to indicate potential disagreements among group members regarding specific recommendation candidates.

4.4 Recommendation of Travel Destinations and Events

CATS (Collaborative Travel Advisory System) [21] is a critiquing-based recommender system that assists a group of friends trying to jointly plan a skiing vacation. Users can provide individual feedback (in terms of critiques) on recommendations determined on the basis of a group profile which has been aggregated out of the set of individual user preferences. The critiquing approach provided in CATS is incremental critiquing where—in contrast to unit critiquing approaches—all critiques of individual users are taken into account when a new recommendation is determined. In the case of inconsistent preferences, "older" critiques are deleted, i.e., the most recent ones are favored when it comes to maintaining consistency.

EVENTHELPR[1] is a publicly-available environment that supports groups in organizing "ad hoc" events. An example is *project meetings*, where partners are supported in terms of providing information about location, restaurants, hotels, and related events. In addition, meetings have an associated agenda that can be defined by the organizer of the meeting or interactively by the group. In this context, agenda items can be evaluated with regard to their importance and then ranked (utility-based) such that the most important agenda items receive a higher ranking. Further application scenarios of EVENTHELPR are *group travels*, *workshops and conferences*, *interactive courses*, *birthday parties*, *sport events*, and *Christmas parties* (Fig. 4.1).

GROUPLINK [36] is a prototype group recommendation environment that recommends events to promote group members' face-to-face interactions in non-work settings. The underlying idea is to determine collections of events where the overall utility of a collection is interpreted as the minimum number of interaction opportunities for individual members (*best-minimum-connected* strategy which is a specific type of *Least Misery*).

[1] www.eventhelpr.com.

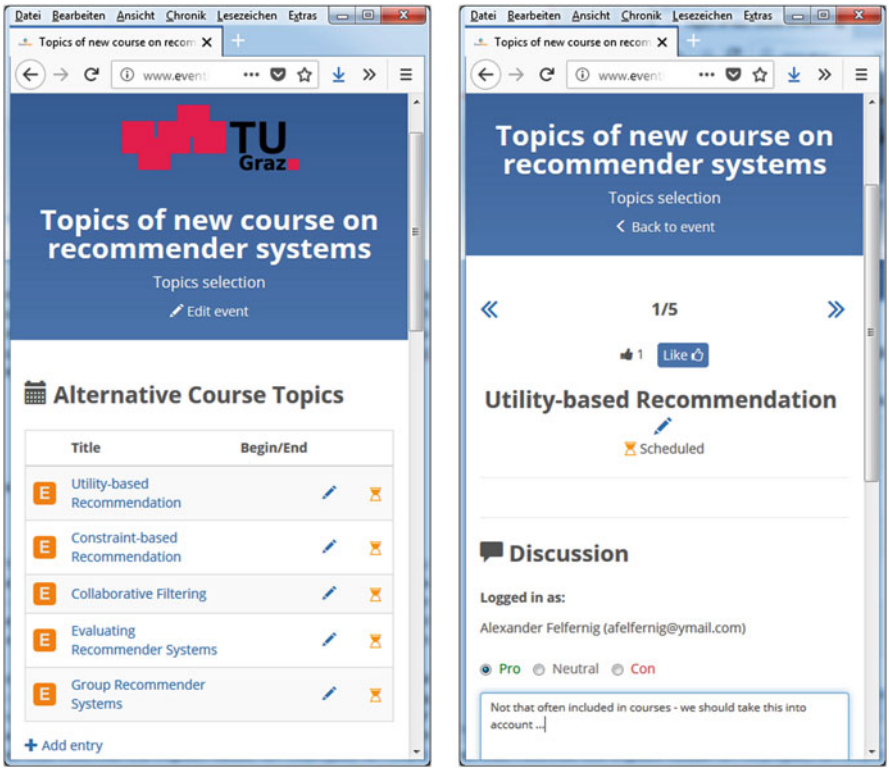


Fig. 4.1 EVENTHELPR: group-based decision making in "ad hoc" events

GROUP MODELER [15] represents a system architecture that supports the creation of group models from a set of individual user models. Different approaches to generate a group model are discussed in [15], where the authors mention museum visits as a typical example of the application of the group recommender.

INTRIGUE (INteractive TouRist Information GUidE) [1] is a prototype tourist information platform that provides personalized information about tourist attractions. The system recommends sightseeing destinations and itineraries that are selected depending on the preferences of the members of a tourist group (e.g., families with children or groups of elderly). The recommendation approach followed by INTRIGUE is to construct a group model and then to perform a utility analysis [37] of different items with regard to the preference criteria contained in the group model. The system provides recommendations for subgroups (e.g., one family) and also recommendations assumed to be relevant for the whole group. INTRIGUE generates explanations as to why certain items are proposed to the group. Importance of user preferences are the major criteria for generating explanations, for example, if a family has a strong preference regarding ancient Roman buildings, amphitheaters could be recommended and the corresponding recommendation would mention the family's preference for Roman culture.

TRAVEL DECISION FORUM [12] is a system that supports the cooperative specification of preferences regarding different dimensions of a tourist destination such as room facilities, hotel facilities, sports facilities, leisure activities, health facilities, and country. For example, in the context of health facilities, evaluation attributes could be the importance of having a whirlpool, a sauna, and a massage. The aggregation mechanisms (available are, e.g., average, median, and random choice) used to generate proposals for the group can be selected by the mediator of a decision task. Recommendations are explained in terms of showing the preferences of individual group members. Furthermore, individual group members can explain their satisfaction with regard to certain aspects of a recommendation and—as a response—can adapt their preferences or specify their preferences with regard to the utility of proposals. For example, group members can specify to which extent it is important for them that the preferences of specific other group members are satisfied. A group recommender application for tourist destinations is also introduced by Nguyen and Ricci [22] where user preferences are derived by analyzing group chats related to the range of available alternatives being discussed.

4.5 Recommendation of News and Web Pages

G.A.I.N. (Group Adapted Interaction for News) [25] is a research prototype that supports the recommendation of personalized news to user groups in public spaces (realized via wall displays and mobile displays). The system derives a group user model from individual models and generates a recommendation thereof, based on one of a selection of supported social choice functions [17].

I-SPY [32] is a collaborative search service which acts as a post-processing service for a search engine. It helps to re-rank results based on preferences learned from a user community with similar information needs. In I-SPY, search behaviors of similar users are grouped to identify search context and thus to help to improve the quality of search. In this context, groups are implicit and anonymous and the overall goal is not primarily to support a group decision, but more to exploit knowledge about the preferences of group members to improve the search quality for individual group members.

LET'S BROWSE [16] also follows the idea of collaborative browsing by providing an agent that supports a group of users in browsing by suggesting items (e.g., web sites) that could be of potential interest to the group. Individual websites are evaluated with regard to their match to the profiles of the currently active users. In other words, the underlying recommendation approach is a content-based one. The similarity metric used is related to a group profile which represents a linear combination of the individual user profiles. LET'S BROWSE also supports explanations: names of users are shown highlighted and the top terms (keywords) from the user profile are highlighted as are terms common to profiles of other users.

4.6 Group Recommenders for Healthy Living

POCKET RESTAURANT FINDER [18] is a system that recommends restaurants to groups on the basis of their culinary preferences and the location of the group members.[2] Group members fill out a profile that includes their preferences regarding restaurants as well as their willingness to travel and limits regarding the amount of money they want to spend. POCKET RESTAURANT FINDER is based on many of the ideas developed in the context of MUSICFX [19]. From individual user preferences, POCKET RESTAURANT FINDER derives a group preference for each restaurant. Further discussions on restaurant recommender systems for groups can be found, for example, in Hallström [11].

XEET is a group decision support environment primarily dedicated to achieving consensus regarding active participation in sports events. Sport events can be announced on different channels such as FACEBOOK or WHATSAPP and feedback from potential participants is immediately visible once it is available. The system provides a nice overview of the different user preferences and summarizes the current state of the preferences. This basic recommendation is given to the creator of the event. In this scenario, users are jointly agreeing on whether or not to participate in an event. The recommendation in this context is a yes/no decision (event should take place or not).

4.7 Group Recommenders in Software Engineering

Software engineering is a group-intensive task where stakeholders often have to make joint decisions [7], for example, regarding the requirements that should be implemented in the next software release or regarding the evaluation (on the meta-level) of a defined set of requirements. INTELLIREQ [23] is an environment for early requirements engineering, i.e., requirements engineering in the initial phases of a software project (see Fig. 4.2).

In INTELLIREQ, requirements can be specified on a textual level. Each user can evaluate a requirement with regard to different interest dimensions (risk, feasibility, cost, relevance, priority, and duration). If all stakeholders who are in charge of evaluating a requirement agree on the estimates for the meta-attributes, consensus is visualized by green traffic lights. If no consensus can be achieved, corresponding red or orange lights are shown which indicate that stakeholders have to perform another evaluation cycle. After having evaluated a requirement, each stakeholder is allowed to see the evaluations of the other stakeholders. Basic recommendation functionalities that support requirement evaluation are *Majority Voting* (MAJ) and *Average* (AVG) recommendation (see Fig. 4.2). The integrated traffic light

[2] An overview of the application of recommendation technologies in the healthy food domain can be found, for example, in Tran et al. [35].

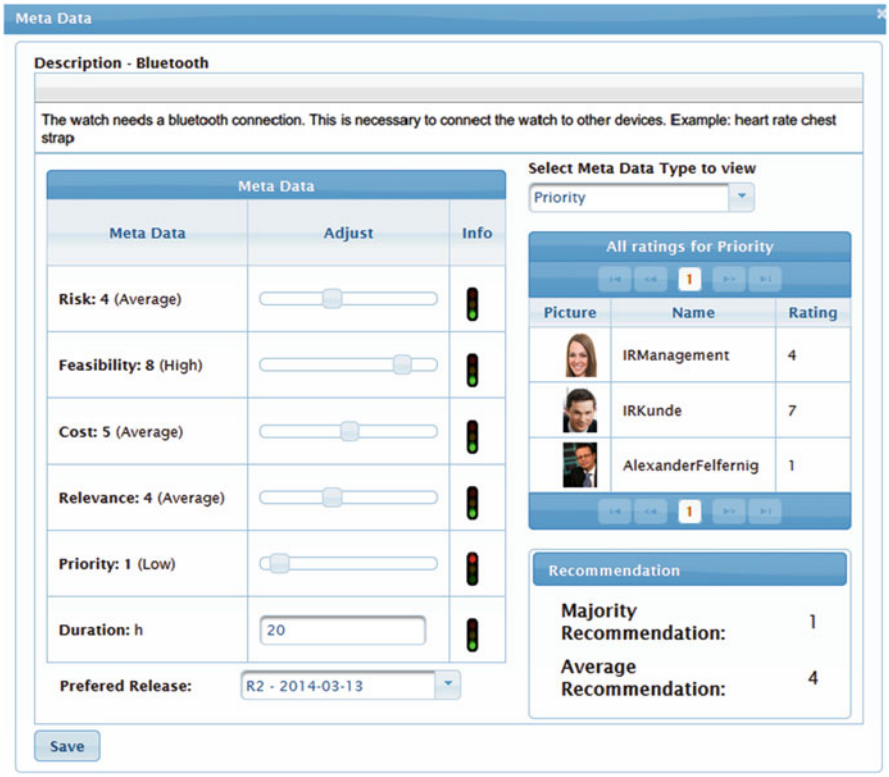


Fig. 4.2 INTELLIREQ: an environment for recommendation-enhanced software requirements engineering [23]

metaphor helps to signal need for completion, i.e., to better engage stakeholders in requirements engineering and thus, to increase the quality of requirement models.

PLANITPOKER[3] is a tool that supports, for example, effort estimation processes related to software requirements. Players of a game are allowed to articulate their estimates, where possible effort values correspond to Fibonacci numbers. The game enforces repeated estimation iterations until consensus regarding the effort of a software requirement is achieved. Note that the tool is not restricted to application in requirements engineering, but is generally applicable in scenarios where groups of users are engaged in a kind of estimation task. PLANITPOKER does not have a dedicated recommendation component; the recommendation can be considered as the output of the process (Fig. 4.3).

[3] www.planitpoker.com.

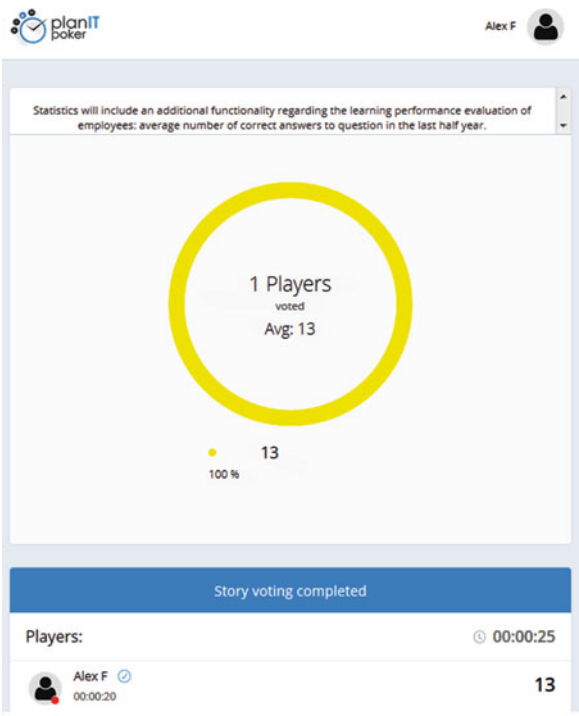


Fig. 4.3 PLANITPOKER: an environment for group-based effort estimation

4.8 Domain-Independent Group Recommenders

CHOICLA[4] [34] is a domain-independent commercially available system for the support of choice tasks that focuses on the ranking and selection of items, for example, deciding which restaurant to visit for a dinner, deciding on a set of requirements to be implemented in the next software release, deciding on a software system to purchase, or deciding the date of the next group meeting (see Fig. 4.4). Group recommender systems play a central role in the support of such tasks. In CHOICLA, *Average Voting* (AVG) and prospect theory [14] are used to determine group recommendations.

In scenarios where alternatives (items) are described in terms of different dimensions (see Fig. 4.4), CHOICLA supports a utility analysis approach for groups which is based on multi-attribute utility theory. When deciding, for example, on a specific accounting software to be purchased by a company, corresponding interest dimensions could be *coverage of needed functionalities*, *trust in the provider of the software*, *economy*, and *technological fitness*. Individual users rate alternatives

[4] www.choicla.com.

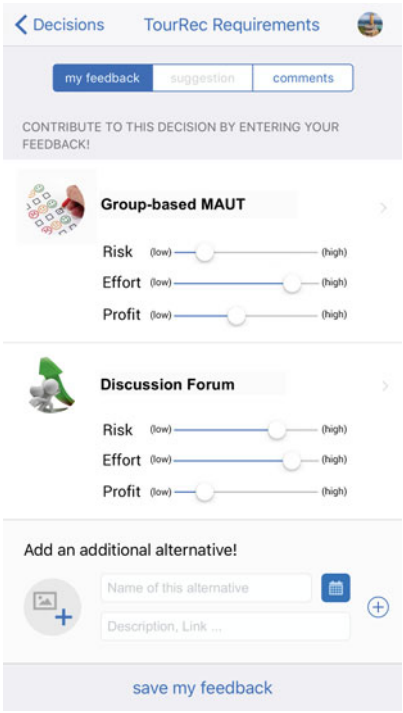


Fig. 4.4 CHOICLA: a domain-independent decision support environment

with regard to these dimensions and then *Average* (AVG) aggregation heuristics are used to aggregate individual user evaluations into one overall group evaluation with regard to a specific dimension of an item.

DOODLE[5] [30] (see Fig. 4.5) is a domain-independent commercially available system that supports different types of group decisions. The major focus of DOODLE is to support decisions regarding the dates of certain events (e.g., meetings in a company or Christmas parties). Individual preferences of group members are defined in terms of different support levels. For example, "yes", "yes, if needed", and "no" are typical answers used when scheduling a meeting. In DOODLE, the basic mechanism to integrate preferences is *Majority Voting* (MAJ), for example, dates that received the highest number of *yes* or *yes, if needed* answers can be considered as recommended alternative. However, DOODLE does not provide recommendations but limits itself to the visualization of the current status of the decision process.

CHOICLAWEB[6] (see Fig. 4.6) is a domain-independent commercially available decision support environment with the goal of supporting a broader range of choice tasks including *ranking and selection*, *configuration*, *release planning*, and

[5] www.doodle.com.

[6] www.choiclaweb.com.

Fig. 4.5 DOODLE: a domain-independent decision support environment

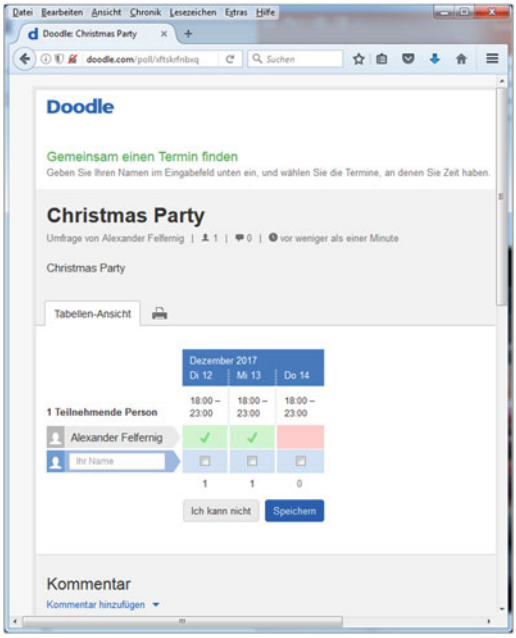
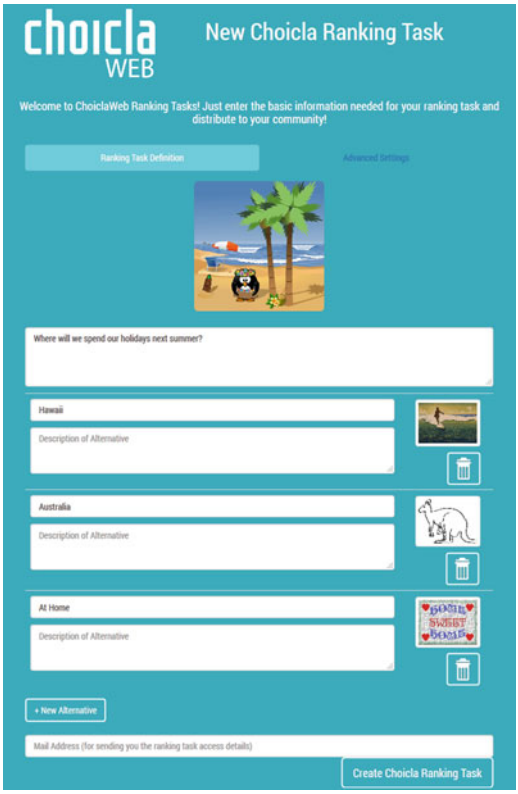


Fig. 4.6 CHOICLAWEB: a domain-independent decision support environment



sequencing. This basically covers the advanced choice scenarios discussed in Chap. 7. CHOICLAWEB also includes basic feedback mechanisms in terms of *polls*, *questionnaires*, and *elections*.

4.9 Conclusions and Research Issues

In this chapter, we gave an overview of different environments that include some kind of group recommendation functionality. Each environment has been analyzed with regard to specific characteristics of group recommender systems. In addition to domain-independent environments, the major application domains considered in this analysis were movies and TV programs, tourism destinations and events, news and web pages, healthy living, and software engineering. A major open issue in the context of group recommender research is the availability of publicly available datasets that provide a basis for the development and comparison of different group recommendation approaches. What already exists for single user recommendation domains, for example, in terms of the MovieLens dataset[7] should also be made available for group recommendation scenarios. Such datasets can serve as a driving force for new recommender-related research developments. There are also a couple of new application areas for group recommender systems. For example, the OPENREQ[8] research project focuses on the development of recommendation and decision technologies that support different kinds of requirements engineering processes [10]. A related issue is the scoping of product lines, i.e., to decide which features should be included in a new product line [31]. Another application domain for group recommendation technologies is sports, for example, recommendation technologies could be used to recommend training sessions for teams depending on the *current team configuration / team members currently participating in a training*. An example thereof is a tennis training session where players with different strengths and weaknesses are participating. In the context of the Internet of Things (IoT) [2], there are various application scenarios for group recommendation technologies [9]. For example, in in-store purchasing scenarios, product information and infomercials must be personalized for the users currently near the screen hardware. Similar scenarios exist in the context of public displays where information has to be adapted to the users in the surrounding area. A privacy-related challenge in this context is to identify the relevant user information in a manner that users would find acceptable.

[7] www.movielens.org.

[8] www.openreq.eu.

References

1. L. Ardissono, A. Goy, G. Petrone, M. Segnan, P. Torasso, INTRIGUE: personalized recommendation of tourist attractions for desktop and handset devices. Appl. Artif. Intell. **17**(8–9), 687–714 (2003). Special Issue on Artificial Intelligence for Cultural Heritage and Digital Libraries
2. L. Atzori, A. Iera, G. Morabito, The internet of things: a survey. Comput. Netw. **54**(15), 2787–2805 (2010)
3. S. Berry, S. Fazzio, Y. Zhou, B. Scott, L. Francisco-Revilla, Netflix recommendations for groups. Am. Soc. Inf. Sci. Technol. **47**(1), 1–3 (2010)
4. D. Chao, J. Balthorp, S. Forrest, Adaptive radio: achieving consensus using negative preferences, in *ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL, USA (2005), pp. 120–123
5. I. Christensen, S. Schiaffino, Entertainment recommender systems for group of users. Expert Syst. Appl. **38**(11), 14127–14135 (2011)
6. A. Crossen, J. Budzik, K. Hammond, FLYTRAP: intelligent group music recommendation, in *7th International Conference on Intelligent User Interfaces*, San Francisco, CA, USA (2002), pp. 184–185
7. A. Felfernig, W. Maalej, M. Mandl, M. Schubert, F. Ricci, Recommendation and decision technologies for requirements engineering, in *ICSE 2010 Workshop on Recommender Systems in Software Engineering (RSSE 2010)*, Cape Town, South Africa (2010), pp. 11–15
8. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems, in *Recommendation Systems in Software Engineering* (Springer, Berlin, 2013), pp. 15–37
9. A. Felfernig, S. Polat-Erdeniz, M. Jeran, A. Akcay, P. Azzoni, M. Maiero, C. Doukas, Recommendation technologies for IoT edge devices. Proc. Comput. Sci. **110**, 504–509 (2017)
10. A. Felfernig, M. Stettinger, A. Falkner, M. Atas, X. Franch, C. Palomares, OPENREQ: recommender systems in requirements engineering, in *RS-BDA'17*, Graz, Austria (2017), pp. 1–4
11. E. Hallström, *Group Recommender System for Restaurant Lunches*. KTH Computer Science and Communication (2013)
12. A. Jameson, More than the sum of its members: challenges for group recommender systems, in *International Working Conference on Advanced Visual Interfaces* (2004), pp. 48–54
13. A. Jameson, B. Smyth, Recommendation to groups, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (2007), pp. 596–627
14. D. Kahneman, A. Tversky, Prospect theory: an analysis of decision under risk. Econometrica **47**(2), 263–291 (1979)
15. J. Kay, W. Niu, Adapting information delivery to groups of people, in *1st International Workshop on New Technologies for Personalized Information Access*, Edinburgh (2005), pp. 34–43
16. H. Lieberman, N. Dyke, A. Vivacqua, Let's browse: a collaborative web browsing agent, in *4th International Conference on Intelligent User Interfaces*, Los Angeles, CA, USA (1999), pp. 65–68
17. J. Masthoff, Group recommender systems: combining individual models, in *Recommender Systems Handbook* (Springer, London, 2011), pp. 677–702
18. J. McCarthy, Pocket restaurant finder: a situated recommender system for groups, in *Workshop on Mobile Ad-Hoc Communication*, Minneapolis, MN, USA (2002), pp. 1–10
19. J. McCarthy, T. Anagnost, MusicFX: an arbiter of group preferences for computer supported collaborative workouts, in *Conference on Computer Support Cooperative Work*, Seattle, WA, USA (1998), pp. 363–372
20. K. McCarthy, L. McGinty, B. Smyth, M. Salamó, Social interaction in the CATS group recommender, in *Workshop on the Social Navigation and Community Based Adaptation Technologies* (2006)

21. K. McCarthy, M. Salamó, L. Coyle, L. McGinty, B. Smyth, P. Nixon, Group recommender systems: a critiquing-based approach, in *11th International Conference on Intelligent User Interfaces (IUI 2006)* (ACM, New York, 2006), pp. 267–269
22. T. Nguyen, F. Ricci, A chat-based group recommender system for tourism, in *Information and Communication Technologies in Tourism*, ed. by R. Schegg, B. Stangl (Springer, Cham, 2017), pp. 17–30
23. G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, W. Schanil, INTELLIREQ: intelligent techniques for software requirements engineering, in *Prestigious Applications of Intelligent Systems Conference (PAIS)* (2014), pp. 1161–1166
24. M. O'Connor, D. Cosley, J. Konstan, J. Riedl, PolyLens: a recommender system for groups of users, in *7th European Conference on Computer Supported Cooperative Work* (2001), pp. 199–218
25. S. Pizzutilo, B. DeCarolis, G. Cozzolongo, F. Ambruoso, Group modeling in a public space: methods, techniques, experiences, in *5th WSEAS International Conference on Applied Informatics and Communications*, Malta (2005), pp. 175–180
26. G. Popescu, P. Pu, Probabilistic game theoretic algorithms for group recommender systems, in *2nd Workshop on Music Recommendation and Discovery (WOMRAD 2011)*, Chicago, IL, USA (2011), pp. 7–12
27. G. Popescu, P. Pu, What's the best music you have?: designing music recommendation for group enjoyment in groupfun, in *CHI '12 Extended Abstracts on Human Factors in Computing Systems*, Austin, TX, USA (2012), pp. 1673–1678
28. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, Personality and social trust in group recommendations, in *22nd International Conference on Tools with Artificial Intelligence*, Arras, France (2010), pp. 121–126
29. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, HAPPY MOVIE: a group recommender application in facebook, in *24th International Florida Artificial Intelligence Research Society Conference*, Palm Beach, FL, USA (2011), pp. 419–420
30. K. Reinecke, M. Nguyen, A. Bernstein, M. Näf, K. Gajos, DOODLE around the world: online scheduling behavior reflects cultural differences in time perception and group decision-making, in *Computer Supported Cooperative Work (CSCW'13)*, San Antonio, TX, USA (2013), pp. 45–54
31. S. Shafiee, L. Hvam, M. Bonev, Scoping a product configuration project for engineer-to-order companies. J. Ind. Eng. Manag. **5**(4), 207–220 (2014)
32. B. Smyth, J. Freyne, M. Coyle, P. Briggs, E. Balfe, I-SPY - anonymous, community-based personalization by collaborative meta-search, in *Research and Development in Intelligent Systems XX* (2004), pp. 367–380
33. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, Counteracting anchoring effects in group decision making, in *23rd Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*. Lecture Notes in Computer Science, vol. 9146 (Dublin, Ireland, 2015), pp. 118–130
34. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, M. Jeran, Counteracting serial position effects in the CHOICLA Group decision support environment, in *20th ACM Conference on Intelligent User Interfaces (IUI2015)*, Atlanta, Georgia, USA (2015), pp. 148–157
35. T.N. Trang Tran, M. Atas, A. Felfernig, M. Stettinger, An overview of recommender systems in the healthy food domain. J. Intell. Inf. Syst. 1–26 (2017). https://doi.org/10.1007/s10844-017-0469-0
36. H. Wei, L. Yang, C. Hsieh, D. Estrin, GROUPLINK: group event recommendations using personal digital traces, in *19th ACM Conference on Computer Supported Cooperative Work (CSCW'16)*, San Francisco, CA, USA (2016), pp. 110–113
37. D. Winterfeldt, W. Edwards, *Decision Analysis and Behavioral Research* (Cambridge University Press, Cambridge, 1986)
38. Y. Zhiwen, Z. Xingshe, Z. Daquing, An adaptive in-vehicle multimedia recommender for group users, in *61st Vehicular Technology Conference*, Stockholm, Sweden (2005), pp. 1–5

Chapter 5
Handling Preferences

Alexander Felfernig and Martijn Willemsen

Abstract This chapter presents an overview of approaches related to the handling of preferences in (group) recommendation scenarios. We first introduce the concept of *preferences* and then discuss how preferences can be handled for different recommendation approaches. Furthermore, we sketch how to deal with inconsistencies such as contradicting preferences of individual users.

5.1 Introduction

Before making recommendations, it is necessary to know and understand the preferences of the users you are trying to serve [16]. Recommender systems create different types of preference models in order to discern the relevance of items. The term *preference* in recommender systems can be loosely characterized as *something that refers to the things in a user's head that determine how he/she will evaluate particular alternatives, and what choices he/she will make* [38, 40]. In this broad sense, preferences refer either to taste or to the utility of items (e.g., I like strawberry ice cream), or to the outcome of a decision process: I *prefer* strawberry over chocolate ice cream. In this latter sense, *preference* is by nature a relative statement. As discussed in De Gemmis et al. [16], a preference can also be regarded as an ordering relation between two or more items to describe which of a given set of alternatives best suits a user. Jameson et al. [38] differentiate between *general* and *specific preferences* where the former is related to evaluations on a categorical level[1] (e.g., *economy* of a car is more important than *sportiness*) and the latter to items or attributes (e.g., I prefer to see the movie *Transformers IV* over *Transformers V*).

[1] Level of interest dimensions [75].

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_5

Acquiring the preferences of users and interpreting these in a way that leads to items relevant for users is often a difficult task [16]. Traditional microeconomic models of human decision making assume that consumers are able to make optimal decisions [33, 52]. These models assume that human preferences are the result of a formal process of utility maximization where item utilities and attributes are fully known and remain stable over time. In many real-world settings, this assumption does not hold. For example, if a family wants to purchase a new car, an upper price limit could have been defined at the beginning of the decision process but then be revised in the face of new highly relevant features that were not considered beforehand. Preferences can change because our utilities for items or features change due to the context of the task [5], or simply because relevant features only come to mind over the course of the decision making process.

This evidence against the assumption of a given set of stable preferences led to alternative models of human decision making [56, 59, 65] and also coined the term *preference construction* [5, 44] which states that in many decision making situations, people construct their specific preferences for options while making the decision. In one way or another, most existing recommender systems take into account the fact that preferences are strongly influenced by user goals, personal experiences, information received from family and friends, and cognitive limitations [38]. Depending on the recommendation approach, specific aspects are taken more into account than others. For example, critiquing-based recommendation approaches take into account a user's limited knowledge about the item domain in terms of supporting the exploration of the search space on the basis of critiques; collaborative recommendation approaches simulate recommendations received from family and friends, but still assume that preferences for items, as reflected in their ratings, are stable, like traditional economic models do.

User feedback regarding specific preferences can be given in an explicit (the user is "actively" involved in the elicitation task) or implicit fashion (the user is not "actively" involved) [16, 59] (see Table 5.1). *Explicit feedback* is given, for example, by rating choice alternatives (*relevance feedback)* [16], critiquing the currently presented reference item [9, 64], ranking options via pairwise preferences [41] or choice-based preference elicitation [32], and in terms of explicit preferences with regard to item properties (specifically, in constraint-based recommendation scenarios) [37]. The advantage of explicit methods is an explicit link between the feedback given and the preference that is measured, but this comes at the disadvantage of requiring effort and the active involvement of users, which is not always practical in real life applications. Therefore, recommender systems often use *implicit feedback* that can be collected by observing a user's navigation and purchasing behavior. Implicit feedback is also given in terms of a user's eye movements when interacting with a recommender system [77], movements of users in public contexts [43], or a user's item purchases [15]. However, the link between the user's behavior and the specific preferences and goals of the user is only indirect. There are limits as to what can be inferred through observation [18].

In this chapter, we analyze preference elicitation support in different recommendation approaches (collaborative filtering, content-based filtering, constraint-based,

Table 5.1 Representation of user preferences (see also [58–60])

Recommendation approach	Explicitly formulated preferences	Implicitly formulated preferences
Collaborative filtering	Item ratings [19], pairwise preferences [41], choice-based preference elicitation [32]	Item reviews [10], user location data [61], time of item consumption [72]
Content-based filtering	Item ratings, categories and tags [57], excluded items [8]	Extracted keywords [57], eye movements [77], item reviews [10]
Constraint-based (incl. utility-based) recommendation	Attribute values [22], preferences between attribute values [7, 39, 71], attribute weights [21, 46], interest dimensions [22, 54]	Items selected for comparison, degree of domain knowledge derived from induced conflicts [23]
Critiquing-based recommendation	Critiques on item attributes [51], natural language based critiques [31]	Information from chats [55], eye movements [11]

and critiquing-based recommendation) [26] and also discuss specific aspects related to the group context. Furthermore, we point out ways to deal with inconsistencies in a given set of user preferences [25].

5.2 Collecting Preferences

Depending on the recommendation approach, preferences are observed / collected in different ways. An overview of the different types of preference representations used in recommendation scenarios is given in Table 5.1. Most *group recommender applications* apply preference elicitation approaches that are quite similar to approaches in single user recommender systems [2, 36]. Where appropriate, we will point out relevant differences.

Preferences in Collaborative Filtering The dominant approach to providing explicit preference feedback is to rate items [19, 76]. Implicit preferences are given in the form of item reviews, user location data, and point of time of item consumption [10, 61, 72]. In the context of collaborative-filtering-based group recommender systems, the individual assessments of items represent the (sometimes aggregated) preferences of individual group members. In this context, typically N-point response scales (e.g., 5-star rating scales) are used to represent user feedback. Different rating scales are used in collaborative filtering recommender systems, for example, the MovieLens recommender system [53] offers a 5-point scale (with half-star ratings) whereas the Jester *joke recommender system* provides a continuous rating scale between -10 and $+10$. Last.fm provides a binary rating scale and Netflix recently switched to a thumbs up/down rating, replacing its 5-star rating scale as A/B tests showed it increased explicit user feedback by 200%. This shows that

scale granularity reflects a tradeoff between cognitive effort [68] and amount of information acquired [42]. As ratings only provide an evaluation of solitary items [38], methods have been proposed that (1) take into account pairwise preferences that measure the relative preference between two items [41] or (2) elicit user preferences from list representations, adaptively changing the list to gradually discover the user's preference [32]. Though these alternative methods have not been applied directly to group recommendations, one can envision that asking a group to rank items rather than rate them might provide a more efficient and satisfactory way to discover a ranking that best fits the preferences of the entire group.

Preferences in Content-Based Recommendation Explicit preference feedback is provided in the form of item evaluations and the specification of meta-properties represented, for example, as categories or tags [57]. Implicit preferences are specified, for example, in terms of item reviews [10] and eye movement patterns (collected via eye-tracking) [77]. In the context of content-based group recommender systems, ratings and category preferences represent the preferences of individual group members. As pointed out, for example, in [8], it often makes sense to explicitly specify and represent negative preferences. Taking such information into account in the group recommendation algorithm helps to rule out items which group members consider unacceptable (e.g., in the context of music recommendation [8]).

Preferences in Constraint-Based Recommendation This type of recommender system is used in situations where items and recommendation knowledge is specified on a semantic level, for example, in terms of rules. In single-user as well as in group settings, preferences can be specified on the level of item attributes or user requirements that are related to item properties. In most of the cases, such preferences are represented in terms of specific types of rules [21]. Preferences between item attributes can also be specified on the basis of preference networks [7]. Attribute weights and interest dimensions are often used in the context of a utility-based analysis of recommendation candidates derived from a constraint-based recommendation process [22]. Preference collection in group-based recommendation settings resembles single-user settings, however, mechanisms are needed to resolve inconsistencies between the preferences of group members (see also Chap. 2).

Preferences in Critiquing-Based Recommendation Critiques are collected to derive user-individual recommendations. These can be aggregated afterwards to build a group model that is used for determining group recommendations [51]. Critiques can be specified directly on item attributes via conventional mechanisms such as *compound critiques* or *unit critiques* or on the basis of more advanced concepts such as *natural language based critiques* [31]. Such types of critiques can also be used in group recommendation. Natural language interfaces for group decision support have not been investigated up to now. Further information that can be used to understand preferences is provided in chat-based approaches [55].

5.3 Preference Handling Practices

Types of Preferences Ratings are influenced by the current context of the user [3, 6]. Some examples of contextual factors are (1) the time between item consumption and item evaluation (the longer the time, the more ratings regress towards the middle of the scale [6]) and (2) the type of rating scale used. Anchoring biases (see Chap. 8) can, for example, be reduced by applying binary or star-based rating scales (compared to numerical rating scales [1]). In general, adapted rating scales and preference collection user interfaces help to avoid rating biases, compared to post-hoc debiasing algorithms [1]. An analysis of anchoring effects based on rating interface is also presented in Cosley et al. [14]. The authors show that item evaluations by other users have an impact on the rating behavior of the current user (if made visible). The existence of the *positivity effect* in the recommendation context, i.e., pleasant items are processed and recalled from memory more effectively, is shown in [6]. In the context of group recommender systems, it has also been shown that multi-attribute utility-based rating scales can help to make ratings more stable in terms of a lower standard deviation of individual evaluations [38, 70]. In the context of critiquing-based recommender systems, combined preference feedback such as compound critiques and natural language based feedback helps to significantly reduce the number of critiquing cycles needed by a user to find a relevant item [31, 50]. In conversational recommendation scenarios [13], users specify their preferences in terms of preferred attribute values. In this context, not all attributes are of relevance for each user. For example, in a digital camera recommender, a user might be interested in specifying the desired camera type and resolution but not in specifying the supported video formats (reasons could include the irrelevance of video functionalities for his/her work, or a limited amount of technical domain knowledge). Approaches to recommending which questions/parameters to be shown to users are presented in [20, 21, 45]. Finally, in content-based recommendation, additional knowledge about user preferences collected, for example, in the form of eye-tracking data, can help to significantly improve the prediction quality of the recommendation algorithm [77].

Visibility of Preferences In the context of group decision making, we face the question of how to disclose the preferences of individual group members to other group members [36, 69]. Group members could be interested in seeing the preferences of other group members for different reasons. For example, if there are some experts in the group, non-experts engaged in the decision making process would like to follow the experts (effort-saving aspect [36]). Furthermore, what a single group member wants can depend directly on what other group members want. For example, if one group member likes to play tennis, his/her interest in having a hotel that offers a tennis court depends on the existence of other group members interested in playing tennis. If no other group members are interested in tennis, preferences regarding having a tennis court become moot. However, the other side of the coin is that knowing the preferences of other group members can lead to situations where

potentially decision-relevant knowledge is not made available to all group members due to a focus shift towards analyzing the preferences of other group members [69]. Furthermore, if some group members are able to communicate negative feedback to all group members, phenomena such as *emotional contagion* [49] can occur, i.e., other group members can be infected by negative moods. There is also the danger of *GroupThink* by which strongly coherent groups try to avoid conflicts and therefore agree on already established preferences. As a consequence, preference visibility should be postponed until individual group members have articulated their own preferences with regard to a set of items [69]. Following this approach, the *overall satisfaction* with the outcome of a group decision process can be increased and *anchoring effects* can be reduced, since group members focus more on item evaluation than on the analysis of the preferences of other group members [69]. Postponed preference visibility in collaborative preference specification processes also leads to an increased exchange of decision-relevant knowledge which helps to improve the overall *quality of a decision* [4]. An additional factor to increase the amount of content/knowledge exchange is *recommendation diversity*. In the extreme case, when recommendations reflect opinions that completely contradict the currently-defined preferences of group members, the amount of information exchanged between group members reaches its maximum [28]. How much diversity is accepted by a user (or a group), is still an open issue for future research.

Choice Overload The basic idea underlying the notion of choice overload is that the higher the number of decision alternatives (i.e., items shown by a collaborative and content-based recommender or parameters shown by a constraint-based or critiquing-based recommender), the higher the related effort to analyze alternatives, and the lower the probability that a decision is made (due to *choice overload*) [17, 34, 66]. Bollen et al. [6] analyzed the role of choice overload in the context of collaborative filtering based recommendation scenarios. They detected that larger result sets containing only attractive items do not necessarily lead to higher choice satisfaction compared to smaller item sets. In other words, the increasing attractiveness of result sets is counteracted by an increase in effort. The authors mention an optimal result set size of 5–7 but explicitly point out the need for further related research. A meta-analysis on choice overload [66] showed that choice overload is not omnipresent and that it mostly occurs when alternatives are very similar and users lack sufficient expertise to have stable and clear preferences. Later work by Willemsen et al. [74] showed that latent feature diversification can reduce choice difficulty and improve satisfaction. The diversification method reduced the similarity between items while controlling for their attractiveness, making small sets just as attractive and satisfactory as larger sets, with much less choice difficulty. For group decisions, choice overload could be tackled in creative ways, extending the diversification methods used for single users. One could imagine, for example, giving each group member a (diverse) subset of items out of which the best items should be identified. Afterwards, the group as a whole can decide which options to select from the conjunction of the best items from each of the subsets. In this way, resources of individual decision makers are combined, and larger sets of items can

be handled without much risk of choice overload. Consequently, reducing choice overload by using the joint resources of a group is an interesting new research direction.

In the context of constraint-based and critiquing-based recommender systems, similar studies are needed focusing on aspects such as result set size, but also on number of questions posed to the user and number of different repair alternatives shown in situations where no solution can be found by the recommender system. Mechanisms to reduce the number of questions are presented in [20, 21, 45] where questions are selected on the basis of collaborative recommendation algorithms [20, 21], or where information-gain based measures are used to predict the next relevant questions to be posed to users [45]. Groups often apply choice deferral more frequently than individuals [73]. As mentioned in White et al. [73], possible explanations thereof are (1) defending a choice deferral seems to be easier and easier to justify than the selection of an option. For example, in jury decision making there is often a tendency towards acquittal. (2) Groups are more risk-seeking than individuals (see Chap. 8), and choice deferral is often a riskier behavior. (3) Groups as a whole often have more reasons to defer a decision compared to individuals. Finally, we want to point out that the optimal size of a choice set can also differ, depending on item selection strategy. For example, users who emphasize finding the optimal solution (maximizers) would like to analyze *as many items as possible* whereas users interested in finding a satisfying solution as quickly as possible (satisficers) prefer smaller option sets [67].

5.4 Consistency Management

There exist situations where no solution / recommendation can be found for a given set of user requirements, especially in the context of constraint-based recommendation scenarios [21]. Given, for example, a set of user requirements (represented by a list of attribute/value pairs) which is *inconsistent* with the underlying product catalog (e.g., pre-defined item list), a user needs support to know which attribute values have to be adapted in order to be able to identify a solution [20]. In such scenarios, conflict detection and diagnosis techniques can help to automatically figure out minimal sets of requirements that have to be adapted in order to find a solution [24, 27, 63]. Whereas [24, 27] focus on the determination of personalized diagnoses for single users, [29] introduced an approach to take into account the principles of computational social choice [12] for diagnosing inconsistent user requirements in group-based recommendation and configuration settings (for example, diagnosis ranking is implemented on the basis of *least misery*). In group-based settings, inconsistencies do not only occur between user requirements and the underlying product catalog, but also between the requirements / preferences of different group members [29]. Similar inconsistencies can occur in critiquing-based recommendation scenarios. For example, if the complete critiquing history of a user (or a group [51]) is used to calculate recommendations, inconsistencies between

critiques have to be resolved. In most cases, such inconsistencies are resolved by simply omitting elder critiques and leaving the more recent ones in the active set. Diagnoses for inconsistent requirements can also be regarded as an explanation that can help users out of the *no solution could be found* dilemma [24]. Such explanations can help to make the identification of relevant items more efficient and can also help to increase the *trust* of a user and the degree of *domain knowledge*, which is extremely important in order to make high-quality decisions [23].

5.5 Conclusions and Research Issues

In this chapter, we focused on a short overview of existing approaches to support the handling of preferences. Preference handling mechanisms from single-user recommendation scenarios can often be applied in group-based settings, but more work is needed to investigate how preference elicitation procedures can be optimized for the group recommendation context. Furthermore, we summarized insights from user studies focusing on the acquisition of preferences and also on the management of inconsistent user requirements, i.e., requirements for which a recommender cannot find a solution. In this context, there are a couple of open research issues which will be discussed in the following.

There exist a couple of research contributions that introduce and discuss *aggregation mechanisms* that can be used to integrate individual user preferences. For example, in [47, 48] Masthoff introduces social-choice-based aggregation mechanisms (e.g., *Least Misery* (LMS)—see Chap. 2) that can be used to identify recommendations for a group. Although initial insights have already been provided in terms of which aggregation mechanisms are useful [48], there is no in-depth analysis of which aggregation strategies should be applied in which context. An analysis of the appropriateness of aggregation strategies depending on item type is presented in Felfernig et al. [30]. A related insight is that, for decisions related to high-involvement items, groups tend to apply *Least Misery*-style heuristics, whereas in low-involvement item domains, misery of individual users is accepted to a larger extent. Two examples of aggregation methods used in this context are *Average* (AVG) and *Most Pleasure* (MPL). An open issue in this context is how to integrate basic aggregation functions with knowledge of the personality and emotions of group members (see also [62]). New related insights will serve as a basis for context-dependent preference aggregation mechanisms that take into account the group context before deciding which aggregation and corresponding explanation method to apply.

Avoiding manipulations is an important aspect of assuring high-quality, fair group decision making. In order to achieve this goal, aggregation mechanisms have to be provided (in combination with corresponding recommender user interfaces) that help to avoid different kinds of manipulation efforts. Related work in the context of group recommendation has already been performed by Jameson et al. [35]. For example, median-based aggregation heuristics help to avoid an impact

of extremely high or low item evaluations. Further mechanisms can be included to limit the number of possible item evaluations per group member and to give feedback on the current status of the decision process on a meta-level. For example, in terms of statements such as *user X changed his/her preferences N times with regard to item A, the evaluations range from 1 to 4 stars*. A question that has to be answered in this context is to which extent we have to adapt user interfaces from single user recommendation scenarios to the group context [49]. For example, in which context should one provide information about the preferences of other group members or information about specific inconsistencies between the preferences of group members. Although user interfaces provide different mechanisms to handle user and group preferences, additional approaches have to be developed to improve the quality of the group decision making processes. For example, approaches that better *predict the preferences of the group*, *improve the quality of the decision outcome*, and *enable a more efficient process towards the achievement of group consensus*. User interfaces should also be capable of stimulating intended behavior, for example, stimulating information exchange between group members in order to make decision-relevant knowledge available to the whole group [28].

References

1. G. Adomavicius, J. Bockstedt, C. Shawn, J. Zhang, De-biasing user preference ratings in recommender systems, in *Joint Workshop on Interfaces and Human Decision Making for Recommender Systems*. CEUR Workshop Proceedings, vol. 1253 (2014), pp. 2–9
2. M. Agarwal, D. Reid, Predicting group choice: an experimental study using conjoint analysis, in *Academy of Marketing Science (AMS) Annual Conference* (1984), pp. 445–449
3. X. Amatriain, J. Pujol, N. Tintarev, N. Oliver, Rate it again: increasing recommendation accuracy by user re-rating, in *3rd ACM Conference on Recommender Systems*, New York, USA (2009), pp. 173–180
4. M. Atas, A. Felfernig, M. Stettinger, T.N. Trang Tran, Beyond item recommendation: using recommendations to stimulate knowledge sharing in group decisions, in *9th International Conference on Social Informatics (SocInfo 2017)*, Oxford, UK (2017), pp. 368–377
5. J. Bettman, M. Luce, J. Payne, Constructive consumer choice processes. J. Consum. Res. **25**(3), 187–217 (1998)
6. D. Bollen, M. Graus, M. Willemsen, Remembering the stars?: effect of time on preference retrieval from memory, in *6th CM Conference on Recommender Systems* (Dublin, Ireland, 2012), pp. 217–220
7. R. Brafman, F. Rossi, D. Salvagnin, K. Venable, T. Walsh, Finding the next solution in constraint- and preference-based knowledge representation formalisms, in *12th International Conference on the Principles of Knowledge Representation and Reasoning (KR 2010)*, Toronto, Ontario, Canada (2010), pp. 425–433
8. D. Chao, J. Balthorp, S. Forrest, Adaptive radio: achieving consensus using negative preferences, in *ACM SIGGROUP Conference on Supporting Group Work*, Sanibel Island, FL, USA (2005), pp. 120–123
9. L. Chen, P. Pu, Critiquing-based recommenders: survey and emerging trends. User Model. User-Adap. Inter. **22**(1–2), 125–150 (2012)
10. L. Chen, G. Chen, F. Wang, Recommender systems based on user reviews: the state of the art. User Model. User-Adap. Inter. **25**(2), 99–154 (2015)

11. L. Chen, F. Wang, W. Wu, Inferring users' critiquing feedback on recommendations from eye movements, in *International Conference on Case-Based Reasoning (ICCBR'16)*, Atlanta, GA, USA (2016), pp. 62–76
12. Y. Chevaleyre, U. Endriss, J. Lang, N. Maudet, A short introduction to computational social choice, in *33rd Conference on Current Trends in Theory and Practice of Computer Science*, Harrachov, Czech Republic (2007), pp. 51–69
13. K. Christakopoulou, F. Radlinski, K. Hofmann, Towards conversational recommender systems, in *International Conference on Knowledge Discovery and Data Mining (KDD 2016)*, San Francisco, CA, USA (2016), pp. 815–824
14. D. Cosley, S. Lam, I. Albert, J. Konstan, J. Riedl, Is seeing believing?: how recommender system interfaces affect users' opinions, in *CHI'03* (2003), pp. 585–592
15. A. Crossen, J. Budzik, K. Hammond, FLYTRAP: intelligent group music recommendation, in *7th International Conference on Intelligent User Interfaces*, San Francisco, CA, USA (2002), pp. 184–185
16. M. De Gemmis, L. Iaquinta, P. Lops, C. Musto, F. Narducci, G. Semeraro, Preference learning in recommender systems, in *ECML/PKDD-09 Workshop* (2009), pp. 41–55
17. K. Diehl, C. Poynor, Great expectations?! assortment size, expectations, and satisfaction. J. Mark. Res. **47**(2), 312–322 (2010)
18. M.D. Ekstrand, M. Willemsen, Behaviorism is not enough: better recommendations through listening to users, in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, Boston, MA, USA (ACM, New York, 2016), pp. 221–224
19. M. Ekstrand, J. Riedl, J. Konstan, Collaborative filtering recommender systems. Found. Trends Hum. Comput. Interact. **4**(2), 81–173 (2011)
20. A. Falkner, A. Felfernig, A. Haag, Recommendation technologies for configurable products. AI Mag. **32**(3), 99–108 (2011)
21. A. Felfernig, R. Burke, Constraint-based recommender systems: technologies and research issues, in *ACM International Conference on Electronic Commerce (ICEC08)*, Innsbruck, Austria (2008), pp. 17–26
22. A. Felfernig, G. Friedrich, D. Jannach, M. Zanker, An integrated environment for the development of knowledge-based recommender applications. Int. J. Electron. Commer. **11**(2), 11–34 (2006)
23. A. Felfernig, B. Gula, E. Teppan, Knowledge-based recommender technologies for marketing and sales. Int. J. Pattern Recognit. Artif. Intell. **21**(2), 1–22 (2006). Special Issue of Personalization Techniques for Recommender Systems and Intelligent User Interfaces
24. A. Felfernig, G. Friedrich, M. Schubert, M. Mandl, M. Mairitsch, E. Teppan, Plausible repairs for inconsistent requirements, in *IJCAI'09*, Pasadena, CA (2009), pp. 791–796
25. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Des. Anal. Manuf. **26**(1), 53–62 (2012)
26. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems. *Recommendation Systems in Software Engineering* (Springer, Berlin, 2013), pp. 15–37
27. A. Felfernig, M. Schubert, S. Reiterer, Personalized diagnosis for over-constrained problems, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, Peking, China (2013), pp. 1990–1996
28. A. Felfernig, M. Stettinger, G. Leitner, Fostering knowledge exchange using group recommendations, in *ACM RecSys'15 Workshop on Interfaces and Human Decision Making for Recommender Systems (IntRS'15)*, Vienna, Austria (2015), pp. 9–12
29. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)* (2016), pp. 69–72
30. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, S. Polat-Erdeniz, An analysis of group recommendation heuristics for high- and low-involvement items, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*, Arras, France (2017), pp. 335–344

31. P. Grasch, A. Felfernig, F. Reinfrank, ReComment: towards critiquing-based recommendation with speech interaction, in *7th ACM Conference on Recommender Systems* (ACM, New York, 2013), pp. 157–164
32. M. Graus, M. Willemsen, Improving the user experience during cold start through choice-based preference elicitation, in *Proceedings of the 9th ACM Conference on Recommender Systems*, RecSys '15, Vienna, Austria (ACM, New York, 2015), pp. 273–276
33. D. Grether, C. Plott, Economic theory of choice and the preference reversal phenomenon. Am. Econ. Rev. **69**(4), 623–638 (1979)
34. C. Huffman, B. Kahn, Variety for sale: mass customization or mass confusion? J. Retail. **74**(4), 491–513 (1998)
35. A. Jameson, More than the sum of its members: challenges for group recommender systems, in *International Working Conference on Advanced Visual Interfaces* (2004), pp. 48–54
36. A. Jameson, B. Smyth, Recommendation to groups, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (Springer, New York, 2007), pp. 596–627
37. A. Jameson, S. Baldes, T. Kleinbauer, Two methods for enhancing mutual awareness in a group recommender system, in *ACM International Working Conference on Advanced Visual Interfaces*, Gallipoli, Italy (2004), pp. 447–449
38. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in *Recommender Systems Handbook*, 2nd edn., ed. by F. Ricci, L. Rokach, B. Shapira. (Springer, Berlin, 2015), pp. 611–648
39. D. Jannach, M. Zanker, A. Felfernig, G. Friedrich, *Recommender Systems – An Introduction* (Cambridge University Press, Cambridge, 2010)
40. G. Jawaheer, P. Weller, P. Kostkova, Modeling user preferences in recommender systems: a classification framework for explicit and implicit user feedback. ACM Trans. Interact. Intell. Syst. **4**(2), 1–26 (2014)
41. S. Kalloori, F. Ricci, M. Tkalcic, Pairwise preferences based matrix factorization and nearest neighbor recommendation techniques, in *Proceedings of the 10th ACM Conference on Recommender Systems*, RecSys '16, New York, NY, USA (ACM, New York, 2016), pp. 143–146
42. D. Kluver, T. Nguyen, M. Ekstrand, S. Sen, J. Riedl, How many bits per rating?, in *Proceedings of the Sixth ACM Conference on Recommender Systems*, RecSys '12, New York, NY, USA (ACM, New York, 2012), pp. 99–106
43. E. Kurdyukova, S. Hammer, E. André, Personalization of content on public displays driven by the recognition of group context. Ambient Intell. **7683**, 272–287 (2012)
44. S. Lichtenstein, P. Slovic, *The Construction of Preference* (Cambridge University Press, Cambridge, 2006)
45. T. Mahmood, F. Ricci, Improving recommender systems with adaptive conversational strategies, in *20th ACM Conference on Hypertext and Hypermedia*, Torino, Italy (2009), pp. 73–82
46. J. Masthoff, Modeling the multiple people that are me, in *User Modeling 2003*. Lecture Notes in Artificial Intelligence, vol. 2702 (Springer, Berlin, 2003), pp. 258–262
47. J. Masthoff, Group recommender systems: combining individual models, in *Recommender Systems Handbook* (Springer, London, 2011), pp. 677–702
48. J. Masthoff, Group recommender systems: aggregation, satisfaction and group attributes, in *Recommender Systems Handbook* (Springer, London, 2015), pp. 743–776
49. J. Masthoff, A. Gatt, In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model. User-Adap. Inter. **16**(3–4), 281–319 (2006)
50. K. McCarthy, J. Reilly, L. McGinty, B. Smyth, On the dynamic generation of compound critiques in conversational recommender systems, in *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems* (Springer, Berlin, 2004), pp. 176–184
51. K. McCarthy, L. McGinty, B. Smyth, M. Salamó, Social interaction in the CATS group recommender, in *Workshop on the Social Navigation and Community based Adaptation Technologies* (2006)

52. D. McFadden, Rationality for economists. J. Risk Uncertain. **19**(1–3), 73–105 (1999)
53. B. Miller, I. Albert, S. Lam, J. Konstan, J. Riedl, MovieLens unplugged: experiences with a recommender system on four mobile devices, in *People and Computers XVII - Designing for Society*, ed. by E. O'Neill, P. Palanque, P. Johnson (Springer, London, 2004), pp. 263–279
54. J. Neidhardt, L. Seyfang, R. Schuster, H. Werthner, A picture-based approach to recommender systems. Inform. Technol. Tour. **15**(1), 49–69 (2015)
55. T. Nguyen, F. Ricci, A chat-based group recommender system for tourism, in *Information and Communication Technologies in Tourism*, ed. by R. Schegg, B. Stangl (Springer, Cham, 2017), pp. 17–30
56. J. Payne, J. Bettman, E. Johnson, *The Adaptive Decision Maker* (Cambridge University Press, Cambridge, 1993)
57. M. Pazzani, D. Billsus, Learning and revising user profiles: the identification of interesting web sites. Mach. Learn. **27**(3), 313–331 (1997)
58. B. Peintner, P. Viappiani, N. Yorke-Smith, Preferences in interactive systems: technical challenges and case studies. AI Mag. **29**(4), 13–24 (2008)
59. A. Pommeranz, J. Broekens, P. Wiggers, W. Brinkman, C. Jonker, Designing interfaces for explicit preference elicitation: a user-centered investigation of preference representation and elicitation process. User Model. User-Adap. Inter. **22**(4–5), 357–397 (2012)
60. P. Pu, L. Chen, User-involved preference elicitation for product search and recommender systems. AI Mag. **29**(4), 93–103 (2008)
61. Z. Qiao, P. Zhang, Y. Cao, C. Zhou, L. Guo, Improving collaborative filtering recommendation via location-based user-item subgroup. Proc. Comput. Sci. **29**, 400–409 (2014)
62. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, Social factors in group recommender systems. ACM Trans. Intell. Syst. Technol. **4**(1), 8:1–8:30 (2006)
63. R. Reiter, A theory of diagnosis from first principles. AI J. **32**(1), 57–95 (1987)
64. F. Ricci, Q. Nguyen, Acquiring and revising preferences in a critique-based mobile recommender systems. IEEE Intell. Syst. **22**(3), 22–29 (2007)
65. P. Sawyer, S. Viller, I. Sommerville, A behavioral model of choice. Q. J. Econ. **69**(1), 99–118 (1955)
66. B. Scheibehenne, R. Greifeneder, P. Todd, Can there ever be too many options? a meta-analytic review of choice overload. J. Consum. Res. **37**(3), 409–425 (2010)
67. B. Schwartz, A. Ward, J. Monterosso, S. Lyubomirsky, K. White, D. Lehman, Maximizing versus satisficing: happiness is a matter of choice. J. Pers. Soc. Psychol. **83**(5), 1178–1197 (2002)
68. E. Sparling, S. Sen, Rating: How difficult is it?, in *5th ACM Conference on Recommender Systems* (ACM, New York, 2011), pp. 149–156
69. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, Counteracting anchoring effects in group decision making, in *23rd Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*. Lecture Notes in Computer Science, vol. 9146 (Dublin, Ireland, 2015), pp. 118–130
70. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, M. Jeran, Counteracting serial position effects in the CHOICLA Group decision support environment, in *20th ACM Conference on Intelligent User Interfaces (IUI2015)*, Atlanta, Georgia, USA (2015), pp. 148–157
71. T. Ulz, M. Schwarz, A. Felfernig, S. Haas, A. Shehadeh, S. Reiterer, M. Stettinger, Human computation for constraint-based recommenders. J. Intell. Inf. Syst. **49**(1), 37–57 (2017)
72. S. Wei, N. Ye, Q. Zhang, Time-aware collaborative filtering for recommender systems, in *Chinese Conference on Pattern Recognition*, Beijing, China (2012), pp. 663–670
73. C. White, S. Hafenbrädl, U. Hoffrage, N. Reisen, J. Woike, Are groups more likely to defer choice than their members. Judgm. Decis. Mak. **6**(3), 239–251 (2011)
74. M. Willemsen, M. Graus, B. Knijnenburg, Understanding the role of latent feature diversification on choice difficulty and satisfaction. User Model. User-Adap. Inter. **26**(4), 347–389 (2016)

75. D. Winterfeldt, W. Edwards, *Decision Analysis and Behavioral Research* (Cambridge University Press, Cambridge, 1986)
76. H. Xie, J. Lui, Mathematical modeling of group product recommendation with partial information: How many ratings do we need? Perform. Eval. **77**, 72–95 (2014)
77. S. Xu, H. Jiang, F. Lau, Personalized online document, image and video recommendation via commodity eye-tracking, in *ACM Conference on Recommender Systems (RecSys'08)*, Lausanne, Switzerland (2008), pp. 83–90

Chapter 6
Explanations for Groups

Alexander Felfernig, Nava Tintarev,
Thi Ngoc Trang Tran, and Martin Stettinger

Abstract Explanations are used in recommender systems for various reasons. Users have to be supported in making (high-quality) decisions more quickly. Developers of recommender systems want to convince users to purchase specific items. Users should better understand how the recommender system works and why a specific item has been recommended. Users should also develop a more in-depth understanding of the item domain. Consequently, explanations are designed in order to achieve specific *goals* such as increasing the transparency of a recommendation or increasing a user's trust in the recommender system. In this chapter, we provide an overview of existing research related to explanations in recommender systems, and specifically discuss aspects relevant to group recommendation scenarios. In this context, we present different ways of explaining and visualizing recommendations determined on the basis of *aggregated predictions* and *aggregated models* strategies.

6.1 Introduction

Explanations have been recognized as an important means to help users to evaluate recommendations, and make better decisions, but also to deliver persuasive messages to the user [30, 62]. Empirical studies show that users appreciate explanations of recommendations [14, 30]. Explanations can be regarded as *a means to make something clear by giving a detailed description* [63]. In the recommender systems context, Friedrich and Zanker [26] define explanations as *information about recommendations* and as *means to support objectives defined by the designer of a recommender system*. Explanations can be seen from two basic viewpoints [5, 65]: (1) the *user's* (*group member's*) and (2) the *recommender provider's* point of view. *Users of recommender systems* are in the need of additional information to be able to develop a better understanding of the recommended items. *Developers of recommender systems* want to provide additional information to users for various

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_6

reasons, for example, to convince the user to purchase an item, to increase a user's item domain knowledge (educational aspect), and to increase a user's *trust in* and overall *satisfaction with* the recommender system. Another objective is to make users more *tolerant* with regard to recommendations provided by the system. This is especially important for new users/items, otherwise a recommendation may be perceived as inappropriate. Solely providing the core functionality of recommender systems, i.e., showing a list of relevant items to users, could evoke the impression of interacting with a *black box* with no transparency and no additional user-relevant information [30, 62]. Consequently, explanations are an important means to provide information related to recommendations, the recommendation process, and further objectives defined by the designer of a recommender system [13, 26, 38, 53, 67]. Visualizations of explanations can further improve the perceived quality of a recommender system [27, 65, 67]—where appropriate, examples of visualizations will be provided.

Explanations in Single User Recommender Systems

In single user recommender systems, various efforts have already been undertaken to categorize explanations with regard to *information sources used to generate explanations* and corresponding *goals of explanations* [26, 28, 48, 61, 62, 64]. A categorization of different information sources that can be used for the explanation of recommendations is given, for example, in Friedrich and Zanker [26] where *recommended items*, *alternative items*, and the *user model* are mentioned as three orthogonal *information categories*. Potential goals of explanations are discussed a.o. in Tintarev and Masthoff [62] and Jameson et al. [34]. Examples thereof are *efficiency* (reducing the time needed to complete a choice task), *persuasiveness* (exploiting explanations to change a user's choice behavior) [29], *effectiveness* (proactively helping the user to make higher-quality decisions), *transparency* (reasons as to why an item has been recommended, i.e., answering why-questions), *trust* (supporting a user in increasing her confidence in the recommender), *scrutability* (providing ways to make the user profile manageable), *satisfaction* (explanations focusing on aspects such as enjoyment and usability), and *credibility* (assessed likelihood that a recommendation is accurate). Bilgic and Mooney [5] offer a differentiation between explanations that focus on (1) *promotion*, i.e., convincing users to adopt recommendations, and (2) *satisfaction*, i.e., to help users make more accurate decisions.

Examples of verbal explanations for single user recommendations include phrases such as (1) "*users who purchased item x also purchased item y*", (2) "*since you liked the book x, we recommend book y from the same authors*", (3) "*since you prefer taking sports photos, we recommend camera y because it supports 10 pics/sec in full-frame resolution*", and (4) "*item y would be a good choice since it is similar to the already presented item x and has the requested higher frame rate (pics/sec)*". These example explanations are formulated based on the information collected

and provided by the underlying recommendation approaches, i.e., (1) collaborative filtering, (2) content-based filtering, (3) constraint-based recommendation, and (4) critiquing-based recommendation—see, for example, [12, 18, 28, 30]. These examples of explanations can be regarded as "basic", since further information could be included. For instance, information related to competitor items and previous user purchases: "*since you prefer taking sports photos, we recommend camera y because it supports 10 pics/sec in full-frame resolution. z would have been the other option but we propose y since you preferred purchasing from provider k in the past and y is only a little bit more expensive than its competitors*".

Another type of explanation is the following: "*no solution could be found—if you increase the maximum acceptable price or decrease the minimum acceptable resolution, a corresponding solution can be identified*". This explanation focuses on indicating options to find a way out of the "no solution could be found" dilemma which primarily occurs in the context of constraint-based recommendation scenarios [16]. Another example is "*item y outperforms item z in both, quality and price, whereas x outperforms z only in quality*". This explanation does not focus on one item but supports the *comparison* of different candidate items (in this case, *x* and *y*). Importantly, it is directly related to the concept of *asymmetric dominance* (*y* outperforms *z* two times whereas *x* does this only once) which is a *decision bias* discussed in Chap. 8. Explanations based on *item comparisons* are mostly supported in critiquing-based [12] and constraint-based recommendation [17] which are both based on semantic recommendation knowledge (see Chaps. 1 and 2). In critiquing-based recommendation, *compound critiques* point out the relationship between the current reference item and the corresponding candidate items [43]. An example of a compound critique in the domain of *digital cameras* is the following: *on the basis of the current reference item x, you can take a look at cameras with a [lower price] and a [higher resolution] or at cameras with a [higher price] and a [higher optical zoom]*. An analysis of comparison interfaces in single user constraint-based recommendation is presented in [17, 22].

Explanations in Group Recommender Systems

The aforementioned explanation approaches focus on single users, and so, do not have to consider certain aspects of group decision making. Explanations for groups can have *further goals* such as *fairness* (taking into account, as far as possible, the preferences of all group members), *consensus* (group members agree on the decision), and *optimality* (a group makes an optimal or nearly-optimal decision[1]). An important aspect in this context is that explanations show how the interests of individual group members are taken into account. This is not relevant in the

[1] In contrast to single-user decision making, the exchange of decision-relevant knowledge among group members has to be fostered [4].

context of single user recommender systems. Understanding the underlying process enables group members to evaluate the appropriateness of the way their preferences have to been taken into account by the group recommender system. Similar to explanations for single users, explanations for groups are shaped by the underlying recommendation algorithms. Explanations similar to those already mentioned can also be defined in a group context. For example, (1) "*groups that like item x also like item y*", (2) "*since the group likes the film x, we also recommend film y from the same director*", (3) "*since the maximum camera* price *accepted by group members is* 500 *(defined by Paul) and the* minimum accepted resolution *is* 18 mpix *(defined by Joe), we recommend y which supports* 20 mpix *at a* price *of* 459.", and (4) "*item x is a good choice since it supports a higher frame rate requested by all group members and is only a little bit more expensive*."

These examples show that the chosen preference aggregation approach (see Chap. 2) has an impact on the explanation style. While *aggregated predictions* include information about the individual preferences of group members (e.g., one group member specified the lowest maximum price of 500) and thus support explanation goals such as *fairness* and *consensus*, *aggregated models*-based approaches restrict explanations to the group level (e.g., groups that like *x* also like *y*). More advanced (hybrid) explanations [37] can also be formulated in group recommendation scenarios, for example, "*since all group members prefer sports photography, we recommend camera y rather than camera z. It is only a little bit more expensive but has a higher usability which is important for group member* Joe *who is a newbie in digital photography. Similar groups also preferred y*."

An example of an explanation in a situation where no solution could be found is: "*no* 23 mpix *camera with a* price *below* 250 *could be found. Therefore we recommend camera y with* 20 mpix *and a* price *of* 249 *since price is the most important criterion for all group members.*" Finally, the following example shows how to take into account a group's social reality, for example, in terms of "tactful" explanations [53]: "*Although your preference for item y is not very high, your close friend Peter thinks it is an excellent choice*." This example explanation is formulated on the level of *aggregated predictions* (see Chap. 2) and also takes into account social relationships among group members (e.g., neighborhoods in a social network). On the level of *aggregated models*, an explanation can be formulated as follows: "*A majority thinks that it is a good choice. Some group members think that it is an excellent choice.*" (assuming the existence of at least some aggregated categorization of preferences such as *number of likes*). Taking into account the individual preferences of group members helps to increase *mutual awareness* among group members, and thus counteracts the natural tendency to focus on one's own favorite alternatives [32]. An approach to explaining the *consequences of a given recommendation* is introduced by Jameson et al. [33], where *emotions* of individual group members with regard to a recommendation are visualized in terms of animated characters.

We want to emphasize that *explanations for groups* is a highly relevant research topic with a limited, but nevertheless direction-giving, number of research results [3, 11, 31, 32, 47]. In the following, we sketch ways in which explanations for single-

user recommendation scenarios can be adapted to groups. Following the idea of categorizing explanation types along the different recommendation approaches [63, 68], we discuss explanations for groups in the context of *collaborative-* and *content-based filtering*, as well as *constraint-* and *critiquing-based recommendation*.

6.2 Collaborative Filtering

A widely used example of explanations in collaborative filtering recommenders is "*users who purchased item x also purchased item y*." Such explanations can be generated, for example, on the basis of *association rule mining* which is often used as a model-based collaborative filtering approach [40]. Herlocker et al. [30] analyzed the role of explanations in collaborative filtering recommenders. They focused on the impact of different explanation styles on user acceptance of recommender systems. Explanations were mostly represented graphically. For example, a histogram of neighbors' ratings for the recommended item categorized ratings as "good", "neutral", or "bad". The outcome of their study was that rating histograms are the most compelling way to explain rating data. Furthermore, *simple graphs* were perceived as more compelling than more detailed explanations, i.e., *simplicity of explanations is a key factor*.

An orthogonal approach to propose explanations for collaborative-filtering-based recommendations is presented by Chang et al. [10]. Following the idea of generating recommendations based on knowledge from the crowd (see, e.g., [66]), the authors introduce the idea of asking crowd workers to provide feedback on explanations. Quality assurance is an issue but crowd-sourced explanations were considered high-quality. The authors mention *longer explanation texts* and an *increased number of references to item genres* as examples of indicators of high-quality explanations. An example of a question for crowd-sourcing in group recommendation scenarios is the following: "*given this movie recommendation* (e.g., *Guardians of the Galaxy*), *which of the following are useful explanations for a group of middle-aged persons? Can be viewed by the whole family*; *Includes plenty of songs from the 70ies*; *Best movie we have ever seen*." This way, crowd knowledge can be exploited to better figure out which kinds of explanations are useful in which context and which ones might be particularly well-received by specific groups (in this case, middle-aged persons). A similar approach can be used to figure out relevant explanations in other recommendation approaches, i.e., *which tags to use for an explanation*? (content-based filtering), *which requirements to relax*? (constraint-based recommendation), and *which critiques to propose to the user*? (critiquing-based recommendation).

As mentioned by Bilgic and Mooney [5], a goal of the explanations introduced in Herlocker et al. [30] is to promote items but not to provide more insights as to why the items have been recommended, i.e., not to provide satisfaction-oriented explanations that might help users to make more accurate decisions. There are different ways to move the explanation focus towards more informative explanations. As proposed in [5] (for single user recommenders), a collaborative-

filtering-based explanation can be extended by providing information on items that had a major influence on the determination of the proposed recommendation. Removing the *most influential items* (already rated by group members) from the set of rated items triggers the most significant difference in terms of recommended item ratings. Similar approaches can be used to determine the most influencing items in other recommender types [5, 59].

Collaborative Filtering Explanations for Groups

An example of basic explanations in group-based collaborative filtering is included in PolyLens, where the predicted rating for each group member and for the group as a whole is shown [49]. Some simple examples of how to provide explanations in the context of group-based collaborative filtering scenarios are provided in Tables 6.1 and 6.2. Both examples represent variants of the explanation approaches introduced by Herlocker et al. [30]. Table 6.1 depicts an example of an explanation that is based on the preferences (ratings) of the nearest neighbors ($NN = \bigcup\{n_{ij}\}$) of the group members u_i (for simplicity, we assume the availability of a complete set of rating data). For each recommended item t_i, the corresponding frequency distribution of the ratings of the nearest neighbors of individual group members is shown. Note that *NN* can represent users who are in the intersection of users who rated this item ($\{n_{11}, n_{12}, \ldots\} \cap \ldots \cap \{n_{m1}, n_{mk}, \ldots\}$). Alternatively, *NN* can represent the users in the union of nearest neighbors ($\{n_{11}, n_{12}, \ldots\} \cup \ldots \cup \{n_{m1}, n_{mk}, \ldots\}$). A related explanation can be "*users similar to members of this group rated item t as follows*."

Table 6.2 depicts an example of an explanation that is based on the preferences of neighborhood groups gp_j of the current group *gp*. We assume that ratings are only available in an aggregated fashion (ratings of individual users are not available, e.g., for privacy reasons). In this context, the frequency distribution of the ratings of the nearest neighbor groups is shown for each item t_i. An explanation can contain the following text: "*groups similar to the current group rated item t as follows*."

Table 6.1 Collaborative filtering explanations for *aggregated predictions*, i.e., explanations based on information about the preferences (ratings) of nearest neighbors (n_{ij}) of individual group members u_i

	Ratings of nearest neighbors $n_{ij} \in NN$						Explanation		
	u_1		u_2		u_3				
Rec. item t_i	nn_{11}	nn_{12}	nn_{21}	nn_{22}	nn_{31}	nn_{32}	Bad [0–2]	Neutral [>2–3.5]	Good [>3.5–5]
t_1	4.2	4.9	4.3	3.5	3.2	4.8	0	2	4
t_2	3.5	2.2	2.7	3.2	2.9	3.6	0	5	1
t_3	3.8	3.1	3.7	2.8	3.4	2.6	0	4	2
t_4	4.3	4.9	4.4	4.5	4.0	4.0	0	0	6
t_5	3.7	3.9	3.2	3.5	3.6	2.9	0	3	3

Table 6.2 Collaborative filtering explanations for *aggregated models*, i.e., explanations are based on the aggregated preferences of individual group members

	Ratings of NN groups (gp_j)				Explanation		
Rec. item	gp_1	gp_2	gp_3	gp_4	Bad [0–2]	Neutral [>2–3.5]	Good [>3.5–5]
t_1	4.2	4.9	4.3	3.5	0	1	3
t_2	1.2	2.9	3.1	1.8	2	2	0
t_3	3.5	3.8	2.9	3.3	0	3	1
t_4	4.9	4.8	4.1	4.4	0	0	4
t_5	3.7	3.3	2.4	3.9	0	2	2

In the given examples, explanations refer to ratings but do not take into account aggregation functions that were used (see Chap. 2). Ntoutsi et al. [47] present an approach to explain the aggregation functions in aggregated-prediction-based collaborative filtering. For example, the application of *Least Misery* (LMS) triggers explanations of type "*item y has a group score of 2.9 due to the (lowest) rating determined for user a*." A more 'group-oriented' explanation is "*item y is recommended because it avoids misery within the group*." When using *Most Pleasure* (MPL), the corresponding explanation would be "*item y has a group score of 4.8 due to the (highest) rating determined for user b*." Finally, when using *Average* (AVG), explanations of type "*item y is most similar to the ratings of users a, b, and c*" are provided. Similar explanations can be generated for content-, constraint-, and critiquing-based recommendations. Although initial approaches have already been proposed, different ways to explain group recommendations depending on the used aggregation function(s) are an issue for future research.

Visualization of Collaborative Filtering Explanations for Groups

There are different ways to visualize a recommendation determined using collaborative filtering [30]. The frequency distributions introduced and evaluated by Herlocker et al. [30] can also be applied in the context of group recommendation scenarios. An example thereof is given in Fig. 6.1, where the explanation information contained in Table 6.1 is represented graphically. Figure 6.2 depicts a similar example where an item-specific evaluation of nearest (most similar) groups is shown in terms of a frequency distribution. Alternatively, *spider diagrams* can be applied to visualize the preferences of nearest neighbors. An example is depicted in Fig. 6.3. This type of representation is based on the idea of consensus-based approaches to visualize the current status of a group decision process [41, 50].

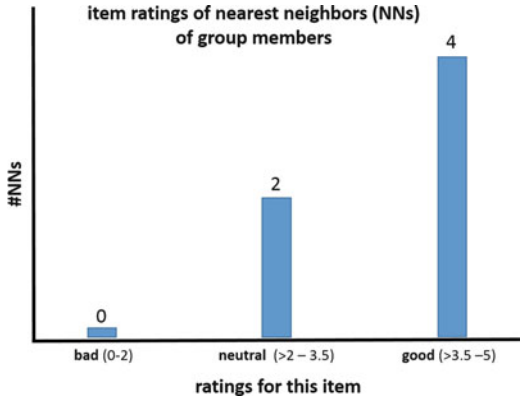


Fig. 6.1 Graphical representation of the explanation data contained in Table 6.1

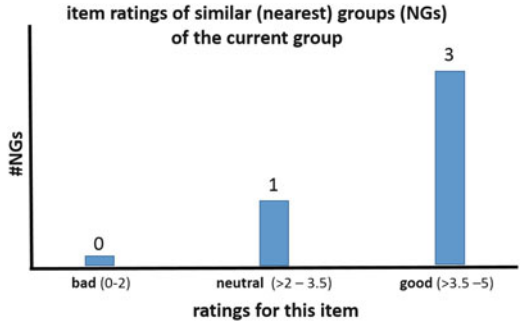


Fig. 6.2 Graphical representation of the explanation data contained in Table 6.2

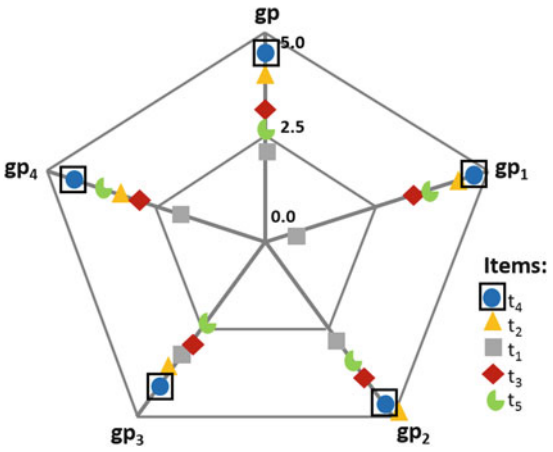


Fig. 6.3 Spider diagram for explaining aggregated models based collaborative filtering recommendations: ratings of nearest neighbor groups $gp_1, ..., gp_4$ of gp for the recommended item t_4. This representation is a variant of consensus-based interfaces discussed in [41]

6.3 Content-Based Filtering

The basis for determining recommendations in content-based filtering is the similarity between item descriptions and keywords (categories) stored in a user profile. Since the importance of keywords can differ among group members, it is important

to identify those which are relevant for all group members [39]. Explanations are based on the analysis of item-related content. Examples of verbal explanations in content-based filtering are given in [5]. The authors show that keyword-style explanations can increase both the perceived trustworthiness and the transparency of recommendations. Such explanations primarily represent occurrence statistics of keywords in item descriptions (see also [14]). Gedikli et al. [28] compare different approaches to representing explanations in content-based filtering scenarios, and show that tag-cloud-based graphical representations outperform verbal approaches.

Content-Based Filtering Explanations for Groups

A simple example of content-based filtering explanations for groups is depicted in Table 6.3.

Item categories cat_j have a user-specific weight (derived, for example, from the category weights of individual user profiles where user u_i is a member of group G). To determine the *explanation relevance* of individual categories, these weights are combined with item-individual weights (see Formula 6.1).

$$explanation\text{-}relevance(cat_j, t_k) = \frac{\Sigma_{u_i \in G} userweight(u_i, cat_j) \times itemweight(t_k, cat_j)}{|G|} \tag{6.1}$$

The higher the explanation-relevance of a category, the higher the category will be ranked in a list shown to the group (members). A verbal explanation related to item t_1 (Table 6.3) can be of the form "*item* t_1 *is recommended since each group member is interested in category* cat_2." If the preference information of individual group members is not available (e.g., for privacy reasons), this explanation would be formulated as "*item* t_1 *is recommended since the group as a whole is interested in category* cat_2." Also, more than one category can be used in such an explanation. As mentioned, category- or keyword-based explanations can also be extended with information about the most influential items [5]. This can be achieved by determining those items that trigger the most significant change in item rating predictions (if not taken into account by the recommendation algorithm).

Table 6.3 Content-based filtering explanations for *aggregated predictions*

	Userweights			Itemweights				Explanation-relevance			
Category	u_1	u_2	u_3	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
cat_1	0.05	0.1	0.15	0.1	0.1	0.2	0.3	0.01	0.01	0.02	0.03
cat_2	0.3	0.4	0.5	0.7	0.2	0.2	0.0	0.28√	0.08√	0.08	0.0
cat_3	0.15	0.25	0.2	0.1	0.4	0.2	0.3	0.02	0.08√	0.04	0.06√
cat_4	0.4	0.3	0.2	0.1	0.2	0.3	0.1	0.03	0.06	0.09√	0.03

The most explanation-relevant categories for an item t_k are marked with √

An approach to explaining recommendations on the basis of tags is presented in Vig et al. [68]. *Tagsplanations* (explanations based on user community tags) are introduced to explain recommendations. In this context, *tag relevance* is defined as the Pearson Correlation (see Chap. 1) between item ratings and corresponding tag preference values. *Tag preference* is the relationship between the number of times a specific tag has been applied to an item compared to the total number of tags applied to the item (weighted with corresponding item ratings). In a study with MovieLens [44] users, the authors show that both tag relevance and tag preference help to achieve the explanation goals of *justification* (why has an item been recommended) and *effectiveness* (better decisions are made). Similar to the example shown in Table 6.3, explanation-relevance (in this case tag relevance) is used to order a list of explanatory tags [68].

An *opinion mining* approach to generating explanations is introduced by Muhammad et al. [45]. In the context of opinion mining, features are extracted from item reviews [15] and then associated with corresponding sentiment scores. Features and corresponding sentiments are then used to generate explanations related to the *pros* and *cons* of specific items. Features are sorted into *pro* or *con* according to whether their values are above or below a predetermined threshold. If we assume, for example, a threshold of 0.4, all item features with an explanation relevance ≥ 0.4 are regarded as pros, the others are regarded as cons. Formula 6.2 represents an approach to determine the explanation-relevance of a specific feature f_i where *sentiment* represents a group preference with regard to a specific feature and *item-sentiment* represents the support of the feature by the item t_j.

$$\textit{explanation-relevance}(f_i) = \textit{sentiment}(f_i) \times \textit{item-sentiment}(t_j, f_i) \tag{6.2}$$

Opinion mining approaches to explanations can also be extended to groups. An example of applying Formula 6.2 in the context of group recommender systems is given in Table 6.4.

This example sketches the generation of explanations in *aggregated models* scenarios. When determining explanations in the context of *aggregated predictions*, explanation relevance could be determined for each individual user and then aggregated using an aggregation function such as *Average* (*AVG*) to select explanations considered most relevant for the group.

Table 6.4 Opinion mining based explanations for *aggregated models*

Group profile (*gp*)		Item-sentiments				Explanation-relevance			
Feature	Sentiment	t_1	t_2	t_3	t_4	t_1	t_2	t_3	t_4
f_1	0.10	0.19	0.23	0.35	0.68	0.019	0.023	0.035	0.068
f_2	0.76	0.61	0.52	0.47	0.52	0.46	0.40	0.36	0.40
f_3	0.21	0.47	0.43	0.21	0.31	0.10	0.09	0.04	0.07
f_4	0.82	0.92	0.76	0.49	0.77	0.75 $\surd$	0.62 $\surd$	0.40 $\surd$	0.63 $\surd$

Features f_i with the highest explanation-relevance are marked with $\surd$

Fig. 6.4 Tag-cloud representation used to show the relevance of tags with regard to a specific item extended with preference information related to group members (*Isa*, *Joe*, and *Leo*)

Visualization of Content-Based Filtering Explanations for Groups

An alternative to list-based representations of explanations is mentioned, for example, in Gedikli et al. [28], where content-based explanations are visualized in the form of *tag-clouds*. An example of a tag-cloud-based explanation in the context of group recommendation is depicted in Fig. 6.4. The used tags are related to our working example from the travel domain (see Chap. 2). In this scenario, the tag-cloud represents an explanation based on the *aggregated preferences* of individual group members. For example, *Leo* and *Isa* like city tours. One can imagine other visual encodings in terms of shape, textures, and highlightings [35]. Tag relevance can be determined on the basis of a tag relevance estimator similar to Formula 6.1.

6.4 Constraint-Based Recommendation

Constraint-based recommender systems are built upon deep knowledge about items and their corresponding recommendation rules (constraints). This information serves as a basis for explaining item recommendations by analyzing reasoning steps that led to the derivation of solutions (items) [25]. Such explanations follow the tradition of AI-based expert systems [6, 26]. On the one hand, explanations are used to answer *how*-questions, i.e., questions related to the reasons behind a recommendation. A corresponding analysis is provided, for example, by Felfernig et al. [17]. *How* questions are answered in terms of showing the relationship between defined user requirements req_i and the recommended items. An example of such an explanation is "*item y is recommended, since you specified the upper price limit with* 500 *and you preferred light-weight cameras*" (for details see [17, 25]). Besides answering *how* questions, constraint-based recommenders help to answer *why* and *why not* questions. Explanations for the first type are used to provide insights to the

Table 6.5 Explanation relevance of requirements in constraint-based recommendation (*aggregated models*)

Requirement	Importance			Explanation relevance
	u_1	u_2	u_3	
req_1	0.2	0.3	0.4	0.3
req_2	0.5	0.4	0.1	0.33
req_3	0.3	0.3	0.5	0.37 $\surd$

The most relevant requirement is marked with $\surd$

user as to why certain questions have to be answered, whereas explanations for *why not* questions help a user to escape from the *no solution could be found* dilemma [20] (see also Chap. 2). Felfernig et al. [17] show that such explanations can help to increase a user's trust in the recommender application. Furthermore, explanations related to *why not* questions can increase the perception of item domain knowledge.

Explanations in Constraint-Based Recommendation for Groups

Formula 6.3 represents a simple example of an approach to determine the explanation-relevance of user requirements in constraint-based recommendation scenarios for groups. A related example is depicted in Table 6.5. The assumption is that all group members have already agreed on the set of requirements $\bigcup req_j$ and each group member has also specified his/her preference in terms of an importance value. An explanation that can be provided to a group in such a context is "*requirement* req_3 *is considered important by the whole group*."

$$explanation\text{-}relevance(req_j) = \frac{\Sigma_{u_i \in G} importance(req_j, u_i)}{|G|} \tag{6.3}$$

The example explanation shown in Table 6.5 does not take into account causal relationships between requirements and items [25]. For example, if a group agrees that the *price* of a camera has to be below 1000 and every camera fulfills this criteria, the price requirement does not filter out items from the itemset, so there is no causal relationship between a recommendation subset of a given itemset and the price requirement.

Combining Constraints and Utilities

Constraint-based recommendation is often *combined with* an additional mechanism that supports the ranking of candidate items (see Chap. 1). An example thereof is Multi-Attribute Utility Theory (MAUT) [69] that supports the evaluation of

Table 6.6 Explanation relevance of interest dimensions in utility-based recommendation (*aggregated predictions*)

	Importance			Contribution			Explanation relevance		
Dimension	u_1	u_2	u_3	t_1	t_2	t_3	t_1	t_2	t_3
dim_1	0.1	0.3	0.1	0.3	0.3	0.1	0.05	0.05	0.02
dim_2	0.6	0.5	0.3	0.3	0.5	0.6	0.14	0.23 $\surd$	0.28 $\surd$
dim_3	0.3	0.2	0.6	0.4	0.2	0.2	0.15 $\surd$	0.07	0.07

The most relevant dimension is marked with $\surd$

items in terms of a set of *interest dimensions* which can be interpreted as generic requirements. For example, in the digital camera domain, *output quality* is an interest dimension that is related to user requirements such as *resolution* and *sensor size*. Group members specify their preferences with regard to the importance of the interest dimensions dim_i. Furthermore, items t_j have different contributions with regard to these dimensions (see Table 6.6).

Similar to content-based filtering, the item-specific *explanation relevance* of individual interest dimensions can be determined on the basis of Formula 6.4 where *imp* represents the user-specific importance of an interest dimension dim_i and *con* the contribution of an item to dim_i.

$$explanation\text{-}relevance(dim_i, t_j) = \frac{\Sigma_{u_k \in G}(imp(u_k, dim_i) \times con(t_j, dim_i))}{|G|} \tag{6.4}$$

Following this approach, [7, 19, 59, 60] show how to apply utility-based approaches to the selection of evaluative arguments,[2] i.e., arguments with the highest relevance. In this context, arguments take over the role of the previously-mentioned interest dimensions. Such an approach is provided in the INTRIGUE system [3], where recommended travel destinations are explained to groups, and arguments are chosen depending on their utility for individual group members or subgroups.

An example of an argument (as an elementary component of an explanation) for a car recommended by a constraint-based recommender is *'very energy-efficient'*, where *energy-efficiency* can be regarded as an interest dimension. The contribution of an item to this interest dimension is high if, for example, the fuel consumption of a car is low. If a customer is interested in energy-efficient cars and a car is energy efficient, the corresponding argument will be included in the explanation (see the example in Table 6.6). An example explanation from another domain (financial services) is the following: "*financial service t_1 is recommended since all group members strongly prefer low-risk investments*." Examples of interest dimensions used in this context are *risk*, *availability*, and *profit*.

[2] In line with Jameson and Smyth [32], we interpret arguments as elementary parts of explanations.

Consensus in Group Decisions

Situations can occur where the preferences of individual group members become inconsistent [21, 23, 41]. In the context of group recommendation scenarios, consensus is defined in terms of disagreement between individual group members regarding item evaluations (ratings) [2].[3] To provide a basis for establishing consensus, such situations have to be explained and visualized [31, 41]. In this context, diagnosis methods (see Chaps. 1 and 2) can help to determine repair actions that propose changes to the current set of requirements (preferences) such that a recommendation can be identified. Such repairs are able to take into account the individual preferences of group members [23]. The potential of aggregation functions (see, e.g., Table 2.2) to foster consensus in group decision making is discussed in Salamo et al. [55]. Concepts to take into account consensus in group decision making are also presented in [2, 8, 9]. In scenarios such as software requirements engineering [46], there are often misconceptions regarding the evaluation/selection of a specific requirement. For example, there could be misconceptions regarding the assignment of a requirement to a software release. An explanation in such contexts indicates possible changes of requirements (assignments) that help to restore consistency (see Chap. 2). In group-based settings, such repair-related explanations help group members understand the constraints of other group members and decide in which way their own requirements should be adapted.

User-Generated Explanations

User-generated explanations are defined by a group member (typically, the creator of a decision task) to explain, for example, why a specific alternative has been selected. The impact of user-generated explanations in constraint-based group recommendation scenarios was analyzed by Stettinger et. al [58]. The creator of a decision task (prioritization decisions in the context of software requirements engineering) had to explain the decision outcome verbally. In groups where such explanations were provided, this contributed to an increased satisfaction with the final decision and an increased perceived degree of group decision support quality [58]. User-generated explanations are not limited to constraint-based recommendation. For example, crowd-sourcing based approaches are based on the similar idea of collecting explanations directly from users.

[3] See also Chap. 3.

Fairness Aspects in Groups

Fair recommendations in group settings can be characterized as *recommendations without favoritism or discrimination towards specific group members*. The perceived importance of fairness, depending on the underlying item domain, has been analyzed in [24]. An outcome of this study is that in high-involvement item domains (e.g., decisions regarding new cars, financial services, and apartments), the preferred preference aggregation strategies (see Chap. 2) differ from low-involvement item domains such as restaurants and movies. The latter are often the domains of repeated group decisions (e.g., the same group selects a restaurant for a dinner every 3 months). Groups tend to apply strategies such as *Least Misery* (LMS), in high involvement item domains, and to prefer *Average Voting* (AVG) in low-involvement item domains. When recommending packages, the task is to recommend a set of items in such a way that individual group members perceive the recommendation as fair [56]. One interpretation of fairness stated in Serbos et al. [56] is that there are at least m items included in the package that a group member likes (see Chap. 3).

An approach to take into account fairness in *repeated group decisions* is presented by Quijano-Sanchez et al. [52], where rating predictions are adapted to achieve fairness in future recommendation settings. This adaptation also depends on the personality of a group member. For example, a group member with a strong personality who was treated less favorably last time will be immediately compensated in the upcoming group decision (see Chap. 9). A similar interpretation of fairness is introduced in Stettinger et al. [57] where fairness is also defined in the context of repeated group decisions, i.e., decisions that repeatedly take place within the same or stable groups (groups with a low fluctuation). Fairness in this context is achieved by introducing functions that systematically adapt preference weights, i.e., group members whose preferences were disregarded recently receive higher preference weights in upcoming decisions. For example, in the context of repeated decisions (made by the same group) regarding a restaurant for a dinner, the preferences of some group members are more often taken into account than the preferences of others. In such scenarios, the preference weights of individual group members can be adapted [57] (see Formulae 6.5 and 6.6).

Formula 6.6 provides a *fairness* estimate per user u_i in terms of the share of the number of supported preferences in relation to the number of defined preferences. The lower the value, the less the preferences of a user (group member of group G) have been taken into account, and the lower the corresponding degree of fairness with regard to u_i. Formula 6.5 reflects an approach to increasing fairness in upcoming recommendation sessions. If the fairness (Formula 6.6) in previous sessions was lower than average, a corresponding upgrade of user-specific importance weights takes place for each dimension. For an example of adapted weights, see Table 6.7.

$$imp'(u_i, dim_j) = imp(u_i, dim_j) \times (1 + (\frac{\Sigma_{u \in G} fair(u)}{|G|} - fair(u_i))) \tag{6.5}$$

$$fair(u_i) = \frac{\#supportedpreferences(u_i)}{\#group\ decisions} \tag{6.6}$$

Table 6.7 An example of an adaptation of individual users' weights to take *fairness* into account

	Importance (imp)				Adapted importance (imp')		
User	dim_1	dim_2	dim_3	Fairness (fair)	dim_1	dim_2	dim_3
u_1	0.3	0.3	0.4	4/8 = 0.5	0.375	0.375	0.5
u_2	0.5	0.4	0.1	6/8 = 0.75	0.5	0.4	0.1
u_3	0.3	0.2	0.5	8/8 = 1.0	0.225	0.15	0.375

In this example, the importance (*imp*) weights of user u_1 have been increased, the weights of u_2 remain the same, and the weights of user u_3 have been decreased (the preferences of u_3 have been favored in previous decisions—a visualization is given in Fig. 6.6)

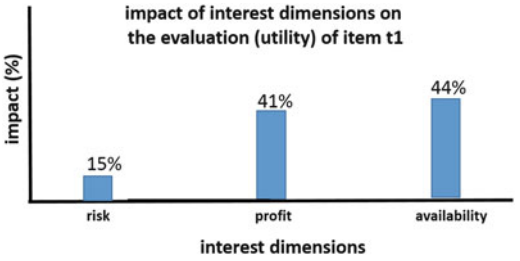


Fig. 6.5 Visualization of the importance of interest dimensions with regard to the overall item evaluation (the importance values are based on Table 6.6 where $dim_1 = risk$, $dim_2 = profit$, and $dim_3 = availability$)

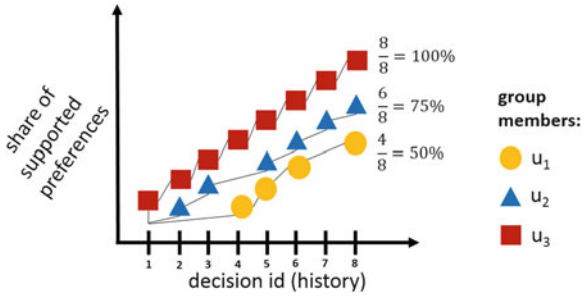


Fig. 6.6 Visualizing the *degree of fairness* (Formula 6.6) in repeated group decisions (e.g., decisions on restaurant visits). In this example, the visualization indicates that user u_1 was at a disadvantage in previous decisions

Visualization of Constraint-Based Explanations for Groups

An example of visualizing the importance of interest dimensions with regard to a final evaluation (utility) is given in Fig. 6.5. Examples of interest dimensions when evaluating, for example, financial services, are *risk*, *profit*, and *availability*.

If the degree of fairness of previous group decisions has to be made transparent to the group, for example, for explaining adaptations regarding the importance weights of individual group members, this can be achieved on the basis of a visualization as depicted in Fig. 6.6. An example of a related verbal explanation is the following: "*the interest dimensions favored by user u_1 have been given more consideration since u_1 was at a disadvantage in previous decisions*."

Table 6.8 Critiques of group members as a basis for generating explanations for item recommendations

Critiques of group members				Support(attribute,t_i)		
Attribute	crit(u_1)	crit(u_2)	crit(u_3)	t_1	t_2	t_3
Price	≤1.000	≤750	≤600	299 (1.0)	650 (0.66)	1.200 (0.0)
Res	≥20	≥18	≥25	24 (0.66)	25 (1.0)	30 (1.0)
Weight	≤1	≤2	≤1	1.5 (0.33)	3 (0.0)	2 (0.33)
Exchangeable lens	y	y	n	y (0.66)	y (0.66)	n (0.33)

Support is defined by the share of attribute-specific critiques supported by an item t_i

6.5 Critiquing-Based Recommendation

To assist users in constructing and refining preferences, critiquing-based recommender systems [12] determine recommendations based on the similarity between candidate and reference items. For example, in the domain of digital cameras, related explanations focus on item attributes such as *price*, *resolution*, and *optical zoom*. *System-generated critiques* (e.g., compound critiques [42]) help to explain the relationship between the currently shown reference item and candidate items. Such explanations have been found to help educate users and increase their *trust* in the underlying recommender system [51].

Critiquing-Based Explanations for Groups

User-defined critiques, i.e., critiques on the current reference item directly defined by the user, can be used for the generation of explanations for recommended items (see the example in Table 6.8).

In this context, $support(attribute, t_i)$ (see Formula 6.7) indicates how often an item supports a user critique on the *attribute*. For example, item t_1 supports a critique on *price* three times since all the critiques on *price* are consistent with the price of t_1, i.e., $support(price, t_1)$=1.0. However, $support(weight, t_1)$ is only 0.33 since the weight of t_1 is 1.5 which is inconsistent with two related critiques.

$$support(attribute, t_i) = \frac{\#supportedcritiques(attribute, t_i)}{\#critiques(attribute)} \tag{6.7}$$

On the verbal level, an explanation for item t_1 could be "*the price of camera* t_1 *(*299*) is clearly within the limits specified by the group members. As expected, it has an exchangeable lens. It has a resolution (24) that satisfies the requirements of* u_1 *and* u_2*, however,* u_3 *has to accept minor drawbacks. Furthermore, the weight of the camera (1.5) is significantly higher than expected by* u_1 *and* u_3."

Table 6.9 Summarization of the support-degree of user-specific critiques on item t_1

	Attributes(t_1)			
User	Price = 299	Resolution = 24	Weight = 1.5	Exchangeable lens = y
u_1	√	√	×	√
u_2	√	√	√	√
u_3	√	×	×	×

Such explanations can be provided if the preferences of group members are known. Otherwise, explanations have to be generated on the basis of *aggregated models*, where item properties are compared with the aggregated critiques defined in the group profile.

Visualization of Critiquing-Based Explanations for Groups

An example of visualizing the support of different attribute-specific critiques is given in Table 6.9. The √ symbol denotes the fact that the user critique on an attribute of item t_i is supported by t_i.

6.6 Conclusions and Research Issues

In this chapter, we provided an overview of explanations that help single users and groups to better understand item recommendations. As has been pointed out in pioneering work by Jameson and Smyth [32], explanations play a crucial role in group recommendation scenarios. We discussed possibilities of explaining recommendations in the context of the basic recommendation paradigms of collaborative filtering, content-based filtering, constraint-based, and critiquing-based recommendation, taking into account specific aspects of group recommendation scenarios. In order to support a more in-depth understanding of how explanations can be determined, we provided a couple of working examples of verbal explanations and corresponding visualizations.

Although extensively analyzed in the context of single-user recommendations (see, e.g., Tintarev [61]), the generation of explanations for groups entails a couple of open research issues. Specifically, aspects of group dynamics have to be analyzed with regard to their role in generating explanations. For example, *consensus*, *fairness*, and *privacy* are major aspects—the related research question is how to define explanations that best help to achieve these goals. Some initial approaches exist to explain the application of aggregation functions in group recommendation contexts (see, e.g., Ntoutsi et al. [47]), however, a more in-depth integration of social choice theories into the generation of explanations has to be performed. This is

also true on the algorithmic level, as in the context of group-based configuration (see Chap. 7). In this context, the integration of information about personality and emotion into explanations has to be analyzed (see also Chap. 9). Initial related work can be found, for example, in Quijano-Sanchez et al. [53] where social factors in groups are taken into account to generate *tactful explanations*, i.e., explanations that avoid, for example, damaging friendships.

Mechanisms that help to increase the quality of group decision making processes have to be investigated [36]. For example, explanations could also be used to trigger intended behavior in group decision making such as exchange of decision-relevant information among group members [4]. Finally, explaining hybrid recommendations [37] and recommendations generated by matrix factorization (MF) approaches [1, 54] are issues for future research. Summarizing, explanations for groups is a highly relevant research area with many open issues for future work.

References

1. B. Abdollahi, O. Nasraoui, Using explainability for constrained matrix factorization, in *11th ACM Conference on Recommender Systems*, Como, Italy (2017), pp. 79–83
2. S. Amer-Yahia, S. Roy, A. Chawla, G. Das, C. Yu, Group recommendation: semantics and efficiency, in *VLDB'09*, Lyon, France (2009), pp. 754–765
3. L. Ardissono, A. Goy, G. Petrone, M. Segnan, P. Torasso, INTRIGUE: personalized recommendation of tourist attractions for desktop and handset devices. Appl. Artif. Intell. Spec. Issue Artif. Intell. Cult. Heritage Digit. Libr. **17**(8–9), 687–714 (2003)
4. M. Atas, A. Felfernig, M. Stettinger, T.N. Trang Tran, Beyond item recommendation: using recommendations to stimulate knowledge sharing in group decisions, in *9th International Conference on Social Informatics (SocInfo 2017)*, Oxford, UK (2017), pp. 368–377
5. M. Bilgic, R. Mooney, Explaining recommendations: satisfaction vs. promotion, in *ACM IUI 2005 Workshop Beyond Personalization*, San Diego, CA, USA (2005), pp. 1–6
6. B. Buchanan, E. Shortliffe, *Rule-Based Expert Systems: The MYCIN Experiments of the Stanford Heuristic Programming Project* (Addison-Wesley, Boston, 1984)
7. G. Carenini, J. Moore, Generating and evaluating evaluative arguments. Artif. Intell. **170**(11), 925–952 (2006)
8. J. Castro, F. Quesada, I. Palomares, L. Martínez, A consensus-driven group recommender system. Intell. Syst. **30**(8), 887–906 (2015)
9. J. Castro, J. Lu, G. Zhang, Y. Dong, L. Martínez, Opinion dynamics-based group recommender systems. IEEE Trans. Syst. Man Cybern. Syst. **99**, 1–13 (2017).
10. S. Chang, F. Harper, L. He, L. Terveen, CrowdLens: experimenting with crowd-powered recommendation and explanation, in *10th International AAAI Conference on Web and Social Media (ICWSM'16)* (AAAI, Menlo Park, 2016), pp. 52–61
11. Y. Chen, Interface and interaction design for group and social recommender systems, in *ACM Conference on Recommender Systems (RecSys'11)*, Chicago, IL (2011), pp. 363–366
12. L. Chen, P. Pu, Critiquing-based recommenders: survey and emerging trends. User Model. User-Adap. Inter. **22**(1–2), 125–150 (2012)
13. L. Chen, F. Wang, Explaining recommendations based on feature sentiments in product reviews, in *ACM IUI 2017* (ACM, New York, 2017), pp. 17–28
14. H. Cramer, V. Evers, S. Ramlal, M. Van Someren, L. Rutledge, N. Stash, L. Aroyo, B. Wielinga, The effects of transparency on trust in and acceptance of a content-based art recommender. User Model. User-Adap. Inter. **18**(5), 455–496 (2008)

15. R. Dong, M. Schaal, M. OMahony, B. Smyth, Topic extraction from online reviews for classification and recommendation, in *23rd International Joint Conference on Artificial Intelligence (IJCAI 2013)* (AAAI, Menlo Park, 2013), pp. 1310–1316
16. A. Felfernig, R. Burke, Constraint-based recommender systems: technologies and research issues, in *ACM International Conference on Electronic Commerce (ICEC08)*, Innsbruck, Austria (2008), pp. 17–26
17. A. Felfernig, B. Gula, E. Teppan, Knowledge-based recommender technologies for marketing and sales. Spec. Issue Pers. Tech. Recomm. Syst. Intell. User Interfaces Int. J. Pattern Recognit. Artif. Intell. **21**(2), 1–22 (2006)
18. A. Felfernig, B. Gula, G. Leitner, M. Maier, R. Melcher, S. Schippel, E. Teppan, A dominance model for the calculation of decoy products in recommendation environments, in *AISB Symposium on Persuasive Technologies*, Aberdeen, Scotland (2008), pp. 43–50
19. A. Felfernig, B. Gula, G. Leitner, M. Maier, R. Melcher, E. Teppan, Persuasion in knowledge-based recommendation, in *3rd International Conference on Persuasive Technology*. Lecture Notes in Computer Science (Springer, Berlin, 2008), pp. 71–82
20. A. Felfernig, M. Schubert, G. Friedrich, M. Mandl, M. Mairitsch, E. Teppan, Plausible repairs for inconsistent requirements, in *21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, CA (2009), pp. 791–796
21. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Des. Anal. Manuf. **26**(1), 53–62 (2012)
22. A. Felfernig, L. Hotz, C. Bagley, J. Tiihonen, *Knowledge-Based Configuration: From Research to Business Cases*, 1st edn. (Elsevier/Morgan Kaufmann Publishers, Burlington, 2014)
23. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)* (2016), pp. 69–72
24. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, S. Polat-Erdeniz, An analysis of group recommendation heuristics for high- and low-involvement items, in *International Conference on Industrial, Engineering and Other Applications of Applied Intelligent Systems (IEA/AIE 2017)*, Arras, France (2017), pp. 335–344
25. G. Friedrich, Elimination of spurious explanations, in *16th European Conference on Artificial Intelligence (ECAI 2004)* (2004), pp. 813–817
26. G. Friedrich, M. Zanker, A taxonomy for generating explanations in recommender systems. AI Mag. **32**(3), 90–98 (2011)
27. E. Gansner, Y. Hu, S. Kobourov, C. Volinsky, Putting recommendations on the map: visualizing clusters and relations, in *ACM Conference on Recommender Systems*, New York, USA (2009), pp. 345–348
28. F. Gedikli, D. Jannach, M. Ge, How should I explain? a comparison of different explanation types for recommender systems. Hum. Comput. Stud. **72**(4), 367–382 (2014)
29. S. Gkika, G. Kekakos, The persuasive role of explanations in recommender systems, in *2nd International Workshop on Behavior Change Support Systems (BCSS 14)* (2014), pp. 59–68
30. J. Herlocker, J. Konstan, J. Riedl, Explaining collaborative filtering recommendations, in *ACM Conference on Computer Supported Cooperative Work* (ACM, New York, 2000), pp. 241–250
31. A. Jameson, More than the sum of its members: challenges for group recommender systems, in *International Working Conference on Advanced Visual Interfaces* (2004), pp. 48–54
32. A. Jameson, B. Smyth, Recommendation to groups, in *The Adaptive Web*, ed. by P. Brusilovsky, A. Kobsa, W. Nejdl. Lecture Notes in Computer Science, vol. 4321 (Springer, Berlin, 2007), pp. 596–627
33. A. Jameson, S. Baldes, T. Kleinbauer, Two methods for enhancing mutual awareness in a group recommender system, in *ACM International Working Conference on Advanced Visual Interfaces*, Gallipoli, Italy (2004), pp. 447–449
34. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in *Recommender Systems Handbook*, ed. by F. Ricci, L. Rokach, B. Shapira, 2nd edn. (Springer, Berlin, 2015), pp. 611–648
35. E. Knutov, P. DeBra, M. Pechenizkiy, AH 12 years later: a comprehensive survey of adaptive hypermedia methods and techniques. New Rev. Hypermed. Multimed. **15**(1), 5–38 (2009)

36. J. Konstan, J. Riedl, Recommender systems: from algorithms to user experience. User Model. User-Adap. Inter. **22**(1), 101–123 (2012)
37. P. Kouki, J. Schaffer, J. Pujara, J. O'Donovan, L. Getoor, User preferences for hybrid explanations, in *11th ACM Conference on Recommender Systems*, Como, Italy (2017), pp. 84–88
38. B. Lamche, U. Adigüzel, W. Wörndl, Interactive explanations in mobile shopping recommender systems, in *8th ACM Conference on Recommender Systems, Joint Workshop on Interfaces and Human Decision Making for Recommender Systems (IntRS'14)*, Foster City, Silicon Valley, California, USA (2014), pp. 14–21
39. H. Lieberman, N. Dyke, A. Vivacqua, Let's browse: a collaborative web browsing agent, in *4th International Conference on Intelligent User Interfaces*, Los Angeles, CA, USA (1999), pp. 65–68
40. W. Lin, S. Alvarez, C. Ruiz, Efficient adaptive-support association rule mining for recommender systems. Data Min. Knowl. Disc. **6**, 83–105 (2002)
41. N. Mahyar, W. Liu, S. Xiao, J. Browne, M. Yang, S. Dow, Consensus: visualizing points of disagreement for multi-criteria collaborative decision making, in *ACM Conference on Computer Supported Cooperative Work and Social Computing* (ACM, New York, 2017), pp. 17–20
42. K. McCarthy, J. Reilly, L. McGinty, B. Smyth, On the dynamic generation of compound critiques in conversational recommender systems, in *International Conference on Adaptive Hypermedia and Adaptive Web-Based Systems* (Springer, Berlin, 2004), pp. 176–184
43. K. McCarthy, J. Reilly, L. McGinty, B. Smyth, Thinking positively - explanatory feedback for conversational recommender systems, in *European Conference on Case-Based Reasoning (ECCBR-04) Explanation Workshop* (2004), pp. 1–10
44. B. Miller, I. Albert, S. Lam, J. Konstan, J. Riedl, MovieLens unplugged: experiences with a recommender system on four mobile devices, in *People and Computers XVII Designing for Society*, ed. by E. O'Neill, P. Palanque, P. Johnson (Springer, London, 2004), pp. 263–279
45. K. Muhammad, A. Lawlor, B. Smyth, A live-user study of opinionated explanations for recommender systems, in *21st International Conference on Intelligent User Interfaces (IUI 2016)* (ACM, New York, 2016), pp. 256–260
46. G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, W. Schanil, INTELLIREQ: intelligent techniques for software requirements engineering, in *Prestigious Applications of Intelligent Systems Conference (PAIS)* (2014), pp. 1161–1166
47. E. Ntoutsi, K. Stefanidis, K. Norvag, H. Kriegel, Fast group recommendations by applying user clustering, in *ER 2012*. Lecture Notes in Computer Science, vol. 7532 (Springer, Berlin, 2012), pp. 126–140
48. I. Nunes, D. Jannach, A systematic review and taxonomy of explanations in decision support and recommender systems. User Model. User-Adap. Inter. **27**, 393–444 (2017)
49. M. O'Connor, D. Cosley, J. Konstan, J. Riedl, PolyLens: a recommender system for groups of users, in *7th European Conference on Computer Supported Cooperative Work* (2001), pp. 199–218
50. I. Palomares, L. Martinez, F. Herrera, MENTOR: a graphical monitoring tool of preferences evolution in large-scale group decision making. Knowl.-Based Syst. **58**, 66–74 (2014)
51. P. Pu, L. Chen, Trust-inspiring explanation interfaces for recommender systems. Knowl.-Based Syst. **20**(6), 542–556 (2007)
52. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, Social factors in group recommender systems. ACM Trans. Intell. Syst. Technol. **4**(1), 8:1–8:30 (2006)
53. L. Quijano-Sanchez, C. Sauer, J. Recio-García, B. Díaz-Agudo, Make it personal: a social explanation system applied to group recommendations. Expert Syst. Appl. **76**, 36–48 (2017)
54. B. Rastegarpanah, M. Crovella, K. Gummadi, Exploring explanations for matrix factorization recommender systems, fatrec workshop, in *11th ACM Conference on Recommender Systems*, Como, Italy (2017)
55. M. Salamo, K. McCarthy, B. Smyth, Generating recommendations for consensus negotiation in group personalization services. Pers. Ubiquit. Comput. **16**(5), 597–610 (2012)

56. D. Serbos, S. Qi, N. Mamoulis, E. Pitoura, P. Tsaparas, Fairness in package-to-group recommendations, in *WWW'17* (ACM, New York, 2017), pp. 371–379
57. M. Stettinger, CHOICLA: towards domain-independent decision support for groups of users, in *8th ACM Conference on Recommender Systems*, Foster City, Silicon Valley, California, USA (2014), pp. 425–428
58. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, Counteracting anchoring effects in group decision making, in *23rd Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*. Lecture Notes in Computer Science, vol. 9146, Dublin, Ireland (2015), pp. 118–130
59. P. Symeonidis, A. Nanopoulos, Y. Manolopoulos, Providing justifications in recommender systems. IEEE Trans. Syst. Man Cybern. **38**, 1262–1272 (2008)
60. J. Teze, S. Gottifredi, A. Garcia, G. Simari, Improving argumentation-based recommender systems through context-adaptable selection criteria. J. Econ. Perspect. **42**(21), 8243–8258 (2015)
61. N. Tintarev, Explaining Recommendations, University of Aberdeen, 2009
62. N. Tintarev, J. Masthoff, Designing and evaluating explanations for recommender systems, in *Recommender Systems Handbook* (Springer, Boston, 2011), pp. 479–510
63. N. Tintarev, J. Masthoff, Evaluating the effectiveness of explanations for recommender systems. User Model. User-Adap. Inter. **22**(4–5), 399–439 (2012)
64. N. Tintarev, J. Masthoff, Explaining recommendations: design and evaluation, in *Recommender Systems Handbook*, ed. by F. Ricci, L. Rokach, B. Shapira, 2nd edn. (Springer, Boston, 2015), pp. 353–382
65. N. Tintarev, J. O'Donovan, A. Felfernig, Human interaction with artificial advice givers. ACM Trans. Interact. Intell. Syst. **6**(4), 1–10 (2016)
66. T. Ulz, M. Schwarz, A. Felfernig, S. Haas, A. Shehadeh, S. Reiterer, M. Stettinger, Human computation for constraint-based recommenders. J. Intell. Inf. Syst. **49**(1), 37–57 (2017)
67. K. Verbert, D. Parra, P. Brusilovsky, E. Duval, Visualizing recommendations to support exploration, transparency and controllability, in *International Conference on Intelligent User Interfaces (IUI'13)*, New York, NY, USA (2013), pp. 351–362
68. J. Vig, S. Sen, J. Riedl, Tagsplanations: explaining recommendations using tags, in *ACM IUI 2009*, Sanibel Island, FL,USA (ACM, New York, 2009), pp. 47–56
69. D. Winterfeldt, W. Edwards, *Decision Analysis and Behavioral Research* (Cambridge University Press, Cambridge, 1986)

Part III
Group Decision Processes

Part III of this book focuses on *group decision making processes* and related topics. In Chap. 7, we introduce and discuss group recommendation scenarios that differ in the way alternatives (items) are represented and recommendations are determined. Examples thereof are group-based configuration, packaging, and sequencing of items. Thereafter, we discuss different biases that can occur in the context of group decision making, and show ways to counteract these biases (Chap. 8). Finally, in Chap. 9 we show how aspects of personality, emotion, and group dynamics can be taken into account in order to improve the overall quality of group recommendations. This book is concluded with Chap. 10.

Chapter 7
Further Choice Scenarios

Alexander Felfernig, Müslüm Atas, Ralph Samer, Martin Stettinger, Thi Ngoc Trang Tran, and Stefan Reiterer

Abstract Until now, we have focused on group recommendation techniques for choice scenarios, related to explicitly-defined items. However, further choice scenarios exist that differ in the way alternatives are represented and recommendations are determined. We introduce a categorization of these scenarios and discuss knowledge representation and group recommendation aspects on the basis of examples.

7.1 Introduction

Until now, we have considered choice scenarios in which a group recommender selects items from a set of explicitly defined (enumerated) items. Examples thereof are the selection of a restaurant for a dinner and the selection of a holiday destination. In this chapter, we analyze scenarios that go beyond the ranking and selection of explicitly defined items (alternatives). We first characterize these scenarios with regard to the aspects of (1) the *inclusion of constraints* (constraints allow the definition of restrictions regarding the combination of choice alternatives) and (2) the *approach to define alternatives* (alternatives can be either represented *explicitly* or in terms of *parameters*). Thereafter, we discuss these scenarios in more detail on the basis of examples. There are hierarchical relationships between some scenarios: *release planning*, *triage*, *resource balancing*, and *sequencing* can be considered as subtypes of *configuration* differing in the type of variables and constraints used. We also differentiate between (1) *basic choice problems* (*ranking*, *packaging*, *parametrization*, *configuration*, *release planning*, *resource balancing*, *sequencing*, and *triage*) and (2) *methods for getting people's input* concerning choice problems (*voting*, *questionnaires*, and *parametrization*). The choice scenarios introduced in this chapter are the following (see Fig. 7.1).

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_7

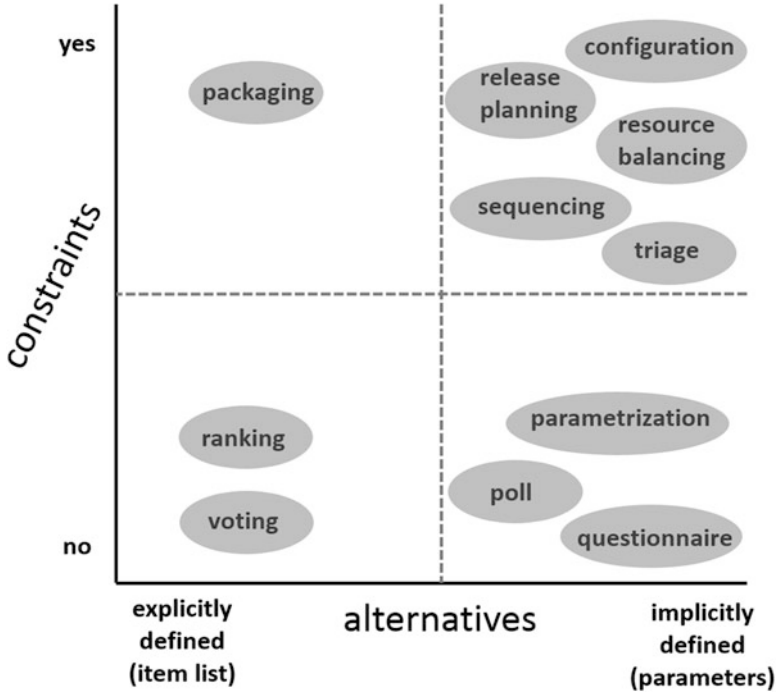


Fig. 7.1 Choice scenarios categorized with regard to (1) *constraint inclusion* and (2) the *representation of alternatives* (as parameters or items)

Ranking The choice scenarios discussed in the previous sections can be regarded as *ranking* since the overall goal is to derive a ranked list of items as a recommendation for a group. Ranking scenarios typically do not include constraints and choice alternatives are represented in the form of a list of explicitly defined items, for example, *restaurants* or *holiday destinations*.

Packaging Package recommendation goes beyond basic ranking [21, 22, 26]. The overall goal is to recommend combinations of items while taking into account constraints that restrict the way in which different items can be combined. For example, in *holiday trip planning*, a package recommendation problem is to find a set of destinations for the group that takes into account global constraints such as upper price limit and maximum total distance between the destinations, but also constraints related to individual items. For example, specific destinations should be excluded, or either one or the other should be visited but not both. Items in packaging problems are specified explicitly, for example, a list of museums and a list of restaurants. Another example of packaging is a group decision regarding the composition of a *Christmas party menu*. Decision alternatives are represented by lists of menu items where each item is associated with one of the categories *starter*, *main dish*, and *dessert*. Constraints can be specified, for example, according to the maximum number of menu items and the upper price limit of a menu.

Parametrization Parametrization decisions are related to detailed aspects of an item—related alternatives are represented as parameter values. In parametrization, no restrictions exist between the parameter values. In the context of group decision making, an example is the *parametrization of an already selected travel destination* or the *parametrization of intended properties of an already selected hotel*. Examples of parameters of a travel destination are number of days to be spent at the destination and time of the year. Parameters describing intended properties of hotels are the availability of a beauty farm, whirlpool, fitness studio, and massage service [14].

Configuration Configuration [2, 10, 24] is one of the most successful applications of Artificial Intelligence techniques. In terms of knowledge representation, configuration scenarios are similar to parametrization, i.e., decision alternatives are represented in terms of parameters. In contrast to parametrization, configuration tasks include a set of constraints that restrict the combination of individual parameter values. Examples thereof are the group-based configuration of *smarthome installations* and the group-based configuration of a *car* (e.g., a new company car) [11, 15]. Further examples of group-based configuration are *release planning* [6], *resource balancing*, *sequencing*, and *triage*. Because of their wide-spread application, these scenarios will be discussed in separate subsections.

Release Planning Both, in terms of knowledge representation and inclusion of constraints, a release planning task is a specific type of configuration task [19]. In Software Engineering, release planning refers to the task of assigning a set of requirements to one of a defined set of releases. This scenario is usually a group decision scenario, since stakeholder groups engaged in a software project have to make release-related decisions. An example of a related constraint is: *since the overall effort is too high, requirement x and requirement y must not be implemented in the same release*.

Triage Similar to release planning, triage can be considered a specific type of configuration task. Triage decisions can occur in domains such as medical decision making and Software Engineering. The overall goal of the underlying decision is to determine a tripartition[1] of a given set of alternatives. In *early requirements engineering* [19], triage can be applied to figure out (1) requirements that are essential for a company and must be implemented immediately, (2) requirements that can be implemented if the resources are available, and (3) unimportant requirements with no need for implementation in the near future. As opposed to this, the focus of release planning is to decide a.o. about the time of implementation. Constraints are similar to those occurring in the context of release planning. Further examples of triage-based decisions are *selection and assignment of students to open research projects of a research group* (students with high potential should be preferred, students with a low probability of successfully completing their tasks should be assigned to standard projects but not research projects, and all other

[1] We limit our discussions to scenarios with three partitions.

students should receive a research project position if possible), *funding decisions* (distribute the available budget between high-potential projects while taking into account an upper funding limit, do not fund low-potential projects, and fund "in-between" projects if additional money is available), *idea management* (focus on high-potential ideas, filter out low-potential ideas, and take into account ideas "in-between" if the needed resources are available), and *product line scoping* [23] (include the most relevant product features, features with potentials for new markets if possible, and filter out low-potential ones).

Resource Balancing The goal of resource balancing is to assign *consumers* to *resources* in such a way that a given set of constraints is satisfied. In this context, consumers and resources can represent humans as well as physical equipment or software. The assignment of resources to consumers can be represented in terms of parameters. Resource balancing often includes a set of constraints, for example, each student should be assigned exactly one paper and paper assignments should be equally distributed. Thus, resource balancing can also be interpreted as a specific kind of configuration task. In configuration scenarios, resource balancing is often included as a subtask, for example, to balance power supply and consumption [10].

Sequencing Sometimes, alternatives have to be arranged in a sequence. For example, when *planning a trip around the island of Iceland*, the sequence of venues (when to visit which destination) has to be clear from the outset since hotel reservations have to be arranged correspondingly. Items in sequencing tasks are often represented in terms of parameters. Constraints are related to user preferences (e.g., three waterfalls should not be visited directly one after another) and further restrictions (e.g., the distance between two destinations in a sequence should be below 100 km and the overall length of the round trip should be minimized).

Polls and Questionnaires Polls and questionnaires are basic means to better understand the opinions of a group or a community. Thus, both can be considered as basic decision support mechanisms. In poll scenarios, the group giving the feedback is in many cases not directly engaged in a decision making process. Polls are defined in terms of a question (parameter) and possible answers. No constraints are defined with regard to the choice alternatives. Questionnaires are a concept similar to polls with the difference that more than one question is typically posed and new questions are sometimes selected depending on answers that have already been provided.

Voting Compared to questionnaires and polls, voting has a strong decision aspect, since a group or a community decides on which alternative(s) should be chosen [16]. This takes place on the basis of a predefined process. The underlying options are represented in an explicit fashion, like presidency candidates or candidate soccer players for the "goal of the month". In voting, there are no constraints regarding the alternatives.[2]

[2]For a discussion of the potential impacts of voting strategies, we refer to [16].

Due to the high diversity of existing choice scenarios, we do not claim completeness. The scenarios presented must be seen as examples, i.e., different variants thereof exist. In the following, we will discuss knowledge representations of the choice scenarios shown in Fig. 7.1, and sketch approaches to include group recommendation techniques.

7.2 Ranking

In basic ranking scenarios [8], *choice alternatives are enumerated* and *no constraints* are applied to the alternatives. A group's task is to identify a ranking and then select *one item* (e.g., in the context of selecting a restaurant for dinner or a logo for a new product) or *a couple of items* (e.g., when selecting the n best conference papers or selecting the n best proposals submitted to a funding organization). Alternatives do not necessarily need to be specified completely before the decision process starts, for example, in *idea competitions* and *open innovation* scenarios, alternatives can be added during the decision process. A simple example of a ranking scenario is depicted in Table 7.1. Each item t_i received one ranking per group member. A score is associated with each rank, for example, rank 1 receives 3 points, rank 2 receives 2 points, etc. The item with the highest *Borda Count* (BRC) (see Chap. 2) scoring is recommended (in our case item t_4 which is indicated with $\surd$ in Table 7.1).[3]

Table 7.1 A basic group-based ranking scenario

Item	Ranking (score)			BRC	Ranking
	u_1	u_2	u_3		
t_1	4 (0)	4 (0)	4 (0)	0	4
t_2	2 (2)	3 (1)	2 (2)	5	2
t_3	3 (1)	2 (2)	3 (1)	4	3
t_4	1 (3)	1 (3)	1 (3)	9	1$\surd$

Group members u_i provide ranks for items $t_i \in I$ (alternatively, rankings can be derived by a recommender—see Chap. 2). Thereafter, an aggregation function such as *Borda Count* (BRC) can be used to derive a corresponding ranking for the group. The $\surd$ symbol indicates the recommended item

[3]The aggregation functions used in this and other scenarios are considered as convenient, however, other alternatives might exist.

Table 7.2 A group-based packaging scenario

	Item ranking (score)								
	Item type 1			Item type 2			Item type 3		
	t_{11}	t_{12}	t_{13}	t_{21}	t_{22}	t_{23}	t_{31}	t_{32}	t_{33}
u_1	1 (3)	2 (2)	3 (1)	2 (2)	1 (3)	3 (1)	1 (3)	2 (2)	3 (1)
u_2	2 (2)	3 (1)	1 (3)	1 (3)	2 (2)	3 (1)	1 (3)	2 (2)	3 (1)
u_3	1 (3)	2 (2)	3 (1)	1 (3)	2 (2)	3 (1)	3 (1)	2 (2)	1 (3)
BRC	8	5	5	8	7	3	7	6	5
Type-wise ranking	1$\surd$	2	2	1$\surd$	2	3	1$\surd$	2	3

Users provide ranks for items t_{ij} (jth item of type i). Thereafter, an aggregation function such as *Borda Count* (BRC) can be used for deriving a proposed package (in our case, $\{t_{11}, t_{21}, t_{31}\}$). The $\surd$ symbol indicates the recommended items part of the package
$c_1 : \forall i : \#proposeditems(type\ i) = 1$

7.3 Packaging

In a packaging scenario (see Table 7.2) [21, 22], each item t_{ij} is associated with a specific item type i. Choice *alternatives are explicitly defined* per item type and *constraints related to the alternatives have to be taken into account*. A group has to select items of different item types and compose these into a corresponding package. An example of a constraint that is defined in such a scenario is: *the number of selected items per item type must be exactly* 1 (see constraint c_1 in Table 7.2). Table 7.2 depicts an example of a group-based packaging scenario. Each item receives a ranking per group member and the item with the highest *Borda Count* (BRC) score within a specific item type i is the group recommendation for item type i. The recommended package in our example is $\{t_{11}, t_{21}, t_{31}\}$. In some scenarios, more than one item per item type is requested or less items than defined types are allowed to be included in a package recommendation. In more complex scenarios, constraints are also specified at the individual item level. An example of such a constraint is an incompatibility between the items t_{22} and t_{33}, i.e., these items must not be part of the same package. In the case of such constraints, solution search in packaging scenarios can be implemented on the basis of conjunctive (database) queries.

7.4 Parametrization

The alternatives are defined in terms of *parameters* and there are *no constraints* related to the alternatives. In such a scenario, a group's task is to select one value per parameter. An example of a group-based parametrization scenario is presented in Table 7.3. Each group member specifies his/her preferences with regard to the different parameters and then the values that were selected in the majority of the cases are considered as candidates for the group recommendation. The recommendation (parametrization) in our example is $\{par_1 = a, par_2 = 1, par_3 = 2\}$.

Table 7.3 Group-based parametrization

Parameter	Preferences u_1	u_2	u_3	MAJ
$par_1(a, b, c)$	a	a	c	a$\surd$
$par_2(1, 2, 3)$	1	1	1	1$\surd$
$par_3(1, 2)$	2	2	1	2$\surd$

Users define preferences with regard to the parameters par_i. Thereafter, an aggregation function such as *Majority Voting* (MAJ) can be used for recommending a parametrization (in our case, $\{par_1 = a, par_2 = 1, par_3 = 2\}$). The $\surd$ symbol indicates recommended parameter values

7.5 Configuration

In group-based configuration scenarios [11], the alternatives are defined by parameters and corresponding domain definitions. In most configuration scenarios, constraints restrict possible combinations of parameter values. Similar to parametrization scenarios, a group's task is to select one value per parameter such that the set of parameter value assignments is consistent with the defined constraints [10]. An abstract example of a group-based configuration scenario is shown in Table 7.4. Each group member specifies his/her preferences with regard to the values of the parameters $\{par_1, \ldots, par_4\}$. An example constraint is $c_1 : par_3 = u \rightarrow par_4 = 1$. Table 7.4 also depicts the solution candidates, i.e., complete sets of parameter assignments that take into account the defined constraints. These configurations include *trade-offs* in terms of neglecting some of the user preferences due to the fact that the union of all user preferences would be inconsistent [7]. In our example shown in Table 7.4, least misery (LMS) is applied to evaluate the configuration candidates (to determine a recommendation). *Misery* in this context is defined as the *number of times the preferences of an individual user are not taken into account* by a configuration. In contrast to rating-based approaches, the higher the value, the lower the quality of the corresponding configuration.

Solving Configuration Tasks Configuration tasks can be solved using constraint solvers [10, 25]. Thus, constraint solvers take over the role of determining candidate recommendations. These solvers generate solutions (candidate recommendations) consistent with the defined set of constraints. Due to the combinatorial explosion, it is often not possible to generate all possible solutions and then to filter out the best ones by using an aggregation function [3]. In order to deal with such situations, *search heuristics* that help to increase the probability of finding solutions that are optimal with regard to a selected aggregation function must be integrated into the constraint solver. A more lightweight integration of aggregation functions can be achieved with *majority voting* (MAJ). The votes of group members can be applied to derive preferences [1]. For example, for par_1 we can derive a preference ordering

Table 7.4 A group-based configuration scenario

	Preferences			Configuration (solution)					Misery			
Parameter	u_1	u_2	u_3	id	par_1	par_2	par_3	par_4	u_1	u_2	u_3	LMS
$par_1(a,b,c)$	a	a	c	1	a	1	u	1	1	1	1	$1\surd$
$par_2(1,2)$	1	1	1	2	c	1	u	1	2	2	0	2
$par_3(u,v)$	u	u	u	3	b	1	u	1	2	2	1	2
$par_4(1,2)$	2	2	1									

Users u_i specify their preferences in terms of parameter values. Constraints c_i specify the restrictions, a configuration must take into account. Thereafter, an aggregation function such as *Least Misery* (LMS) can be used for deriving a recommended configuration (in our case, $\{par_1 = a, par_2 = 1, par_3 = u, par_4 = 1\}$). The $\surd$ symbol indicates the configuration parameter values recommended to the group
$c_1 : par_3 = u \rightarrow par_4 = 1, c_2 : par_2 \neq 2, c_3 : par_3 \neq v$

$a \succ c \succ b$ indicating that a is preferred by a majority of group members (over c and b) and that c is preferred over b. Such preferences can be directly encoded as variable (domain) orderings into a constraint solver [20].[4]

7.6 Release Planning

Release planning is a configuration task [19] where the *alternatives* (possible assignments of requirements to releases) *are defined as parameters* and corresponding domain definitions. In most release planning scenarios, *constraints* restrict the possible assignments of requirements to releases. A group's task is to find one value per parameter (each requirement needs to be assigned to a release) in such a way that all assignments are consistent with the defined constraints. An example of a group-based release planning task is shown in Table 7.5.

Each group member specifies his/her preferences with regard to the assignment of requirements to releases. Example constraints are $c_1 : req_3 \leq req_4$, $c_2 : \forall_i : numreqrel_i \leq 2$ which denote the fact that (1) requirement req_3 must not be implemented after requirement req_4 and (2) no more than two requirements should be assigned to the same release. Similar to the aforementioned configuration scenario, the preferences of individual users are aggregated using *Least Misery* (LMS). In this context, LMS denotes the maximum number of times the preferences of an individual user are neglected by a release plan. For example, release plan 1 ignores the preferences of user u_3 *four times* which is the maximum for release plan 1. Both release plans 2 and 3 have the lowest LMS. Consequently, release plans 2 and 3 can be recommended. Techniques that can be used to determine individual release plans are the same as those discussed in the context of solving configuration tasks.

[4] For example, choco-solver.org.

Table 7.5 A group-based release planning scenario

	Preferences			Release plan					Misery			
Parameter	u_1	u_2	u_3	id	req_1	req_2	req_3	req_4	u_1	u_2	u_3	LMS
$req_1(1..2)$	1	1	2	1	1	1	2	2	1	0	4	4
$req_2(1..2)$	1	1	2	2	1	2	1	2	1	2	2	2$\surd$
$req_3(1..2)$	1	2	1	3	2	1	1	2	1	2	2	2$\surd$
$req_4(1..2)$	2	2	1	4	2	2	1	1	3	4	0	4

Users can specify their preferences in terms of assignments of requirements (req_i) to releases. Additionally, constraints c_i specify properties a release plan must take into account. Thereafter, an aggregation function such as *Least Misery* (LMS) can be used for deriving a proposed release plan (in our case, for example, release plan 2). The $\surd$ symbol indicates recommended release plans
$c_1 : req_3 \leq req_4$, $c_2 : \forall_i : numreqrel_i \leq 2$

Table 7.6 Group-based triage

	Preferences			Triage					Misery			
Parameter	u_1	u_2	u_3	id	req_1	req_2	req_3	req_4	u_1	u_2	u_3	LMS
$req_1(a, m, r)$	a	r	a	1	*a*	*m*	*a*	*m*	2	2	2	2$\surd$
$req_2(a, m, r)$	r	m	r	2	*a*	*r*	*a*	*r*	0	4	0	4
$req_3(a, m, r)$	a	r	a	3	*m*	*a*	*m*	*a*	4	4	4	4
$req_4(a, m, r)$	r	m	r	4	*r*	*a*	*r*	*a*	4	2	4	4

Users specify their preferences by categorizing requirements (req_i) into *a* (accept), *m* (maybe accept), and *r* (reject). Constraints c_1 and c_2 specify dependencies between requirements, c_3 specifies that two requirements have to be accepted (*a*). An aggregation function such as *Least Misery* (LMS) can be used for deriving a triage solution (in our case, triage 1). The $\surd$ symbol indicates the triage recommended to the group
$c_1 : req_1 = req_3$, $c_2 : req_2 = req_4$
$c_3 : a(req_1) + a(req_2) + a(req_3) + a(req_4) = 2$

7.7 Triage

Triage can be regarded as a configuration task. In the context of software requirements engineering, alternative requirements have to be assigned to one of the three triage categories: *accept* (a) = requirement must be implemented, *maybe accept* (m) = requirement can be implemented if resources are available, and *reject* (r) requirement will not be implemented (now). As in release planning, constraints can restrict the assignment of requirements to the three categories. Table 7.6 includes an example of a simple triage task.

A group's task is to assign one category to each requirement in such a way that all assignments are consistent with the defined constraints. In this example, the proposed triage follows the recommendation determined by *Least Misery* (LMS). Techniques that can be used to determine individual triage solutions are the same as those discussed in the context of solving configuration tasks.

Table 7.7 Group-based resource balancing

Parameter	Preference rating (r_iu_j)	Resource assignment (rating)							LMS
		id	r_1u_1	r_1u_2	r_1u_3	r_2u_1	r_2u_2	r_2u_3	
$r_1u_1(0,1)$	5	1	1 (5)	1 (5)	0	0	0	1 (2)	2
$r_1u_2(0,1)$	5	2	1 (5)	0	1 (4)	0	1 (1)	0	1
$r_1u_3(0,1)$	4	3	1 (5)	0	0	0	1 (1)	1 (2)	1
$r_2u_1(0,1)$	4	4	0	1 (5)	1 (4)	1 (4)	0	0	4$\surd$
$r_2u_2(0,1)$	1	5	0	1 (5)	0	1 (4)	0	1 (2)	2
$r_2u_3(0,1)$	2	6	0	0	1 (4)	1 (4)	1 (1)	0	1

Users specify their preferences with regard to resource assignments in terms of *ratings*. Constraints c_i specify properties a resource assignment must take into account. *Least misery* (LMS) denotes the lowest user-specific evaluation of a resource assignment. The $\surd$ symbol indicates the recommended assignment (in our case, assignment 4)

$c_1 : nr_1 = r_1u_1 + r_1u_2 + r_1u_3$

$c_2 : nr_2 = r_2u_1 + r_2u_2 + r_2u_3$

$c_3 : |nr_1 - nr_2| <= 1$

$c_4 : r_1u_1 + r_2u_1 = 1 \wedge r_1u_2 + r_2u_2 = 1 \wedge r_1u_3 + r_2u_3 = 1$

7.8 Resource Balancing

A resource balancing task is defined on the basis of *parameters* r_iu_j indicating the assignment of a consumer (user) u_j to a resource r_i ($r_iu_j = 1 \leftrightarrow$ consumer (user) j is assigned to resource i). In the example given in Table 7.7, each consumer (user) u_j provided a preference evaluation (on a scale 1..5) with regard to all potential assignments r_iu_j.[5] The outcome is a *resource assignment* that indicates which consumer is assigned to which resource(s). In our example, resource balancing is interpreted in such a way that *each resource should be assigned to nearly the same number of consumers* and *each consumer should be assigned to exactly one resource* (see constraints c_1–c_4 in Table 7.7; nr_i are parameters/variables representing the quantity of users assigned to resource i).

Choice scenarios similar to resource balancing in terms of the used knowledge representation are *task assignment* (e.g., a set of tasks has to be assigned to the members of a group) and *production scheduling* (e.g., a set of orders has to be assigned to machines taking into account the preferences of different customers).

7.9 Sequencing

Sequencing can be regarded as a configuration task where sequential numbers have to be assigned to items. As in configuration, constraints can restrict the assignment.

[5] In order to reduce evaluation efforts, a user could specify only preferred items and the system would assume negative evaluations for items a user did not evaluate.

Table 7.8 Group-based sequencing

	Preferences			Sequence				Misery			
Parameter	u_1	u_2	u_3	id	t_1	t_2	t_3	u_1	u_2	u_3	LMS
$t_1(1..3)$	1	1	1	1	1	2	3	2	0	2	2$\surd$
$t_2(1..3)$	3	2	3	2	1	3	2	0	2	0	2$\surd$
$t_3(1..3)$	2	3	2	3	2	1	3	3	2	3	3
				4	2	3	1	2	3	2	3
				5	3	1	2	2	3	2	3
				6	3	2	1	3	2	3	3

Users specify their preferences in terms of assignments of sequential numbers to items t_i. Additionally, constraints c_i specify properties a sequence must take into account. Here, $u_i t_j$ is a parameter representing a user's (u_i) assignment of item t_j to a specific sequence position. *Least misery* (LMS) denotes the number of times, a user preference is neglected by a sequence. Sequences $id = 1$ and $id = 2$ can be regarded as recommendation candidates
$c_1 : \forall u_i : u_i t_1 = x \rightarrow u_i t_2 \neq x \wedge u_i t_3 \neq x \ldots$

Table 7.9 A sequencing scenario where different sequences are explicitly defined, i.e., the choice task is "reduced" to a ranking scenario

Sequence				Evaluation			
id	t_1	t_2	t_3	u_1	u_2	u_3	AVG
1	1	2	3	5	4	3	4$\surd$
2	1	3	2	3	3	5	3.67
3	2	1	3	2	3	5	3.33
4	2	3	1	3	1	1	1.67

In this example, sequence 1 has the highest *Average* (AVG) value, i.e., it will be recommended first

Table 7.8 depicts an example of a sequencing task. A group's task is to assign one sequential number to each item in such a way that all assignments are consistent with the defined constraints (in our case c_1). If sequences have already been pre-defined, sequencing can also be implemented as a ranking task where users evaluate sequences and an aggregation function determines the recommendations. An example thereof is shown in Table 7.9.

Different aspects of sequencing have been investigated by Masthoff [17] in the context of selecting television items (e.g., news and commercials). In the scenarios investigated until now, the primary inputs for determining recommendations are the ratings provided by individual group members. However, as mentioned in [17], a group member's evaluation of an item does not only depend on his/her personal preferences, but also on the context in which the item is shown. The evaluation of an item also depends a.o. on a user's mood (see also Chap. 9). For example, in the context of TV commercials, it is often the case that viewers prefer to see sad commercials in the middle of sad TV programs humorous commercials are preferred in humorous programs. This indicates a need for *consistency*, i.e., users try to maintain a specific mood throughout a TV program [17]. Masthoff presents an in-depth analysis of different influence factors in group decision making in the context of sequencing. Particularly, different social choice functions are compared with

regard to their applicability in the domain of television item sequencing. Results of the presented studies show that group members try to avoid individual misery and care about fairness in group decision making. Interestingly, ratings are used in a non-linear way, i.e., differences between extreme values are considered higher compared to rating values near the average. For further related details, we refer to [17]. Due to the possibility of compensating for items that are perceived suboptimal with better ones, especially in the context of sequencing, it is usually possible to make sure that no one is miserable.

7.10 Polls and Questionnaires

A *poll* is a kind of sampling of opinions on a specific subject which is collected from a selected or a randomized group of persons. A *micro-poll* is a technical term for a short poll that is added, for example, to a website. Polls are used in situations where one is interested in the feedback of a group or a community with regard to a specific topic or question. Thus, polls are used to collect feedback which can be related to a decision, though the group asked is not necessarily affected by the result. Typical examples of such polls are "*how did you like the new version of our software?*" or '*which version of the software do you use, the Android or the iOS-based implementation?*". Users participating in polls can be allowed to select one or more alternatives. A poll on the selection of the employee of the year could allow only one voting per user whereas a poll related to the selection of the best performer of a casting show could allow more than one vote. Systems supporting polls do not include any type of group recommendation functionality, in terms of supporting users in their decision making process. The aggregation mechanism applied in the context of polls is used to summarize the feedback of users (*ADD*-based aggregation) in terms of relative percentages per alternative (e.g., number of persons who voted for a candidate) (Table 7.10). In contrast to polls, *questionnaires* often consist of a collection of questions where the answer type of the questions can be defined in a flexible fashion (e.g., free text answers, multiple-choice answers, and single-choice answers). In some cases, questionnaires are defined on the basis of decision trees that specify in which context a question should be posed.

Table 7.10 Evaluation scheme of polls and questionnaires—persons providing feedback often do not participate in the related decision making process

	u_1	u_2	u_3	Feedback (ADD)
q_1(1,2)	1	1	1	1 (100%)
q_2(1,2,3)	2	3	2	2 (67%) 3 (33%)
q_3(1,2)	1	1	2	1 (67%) 2 (33%)
q_4(1,2,3)	1	2	3	1 (33.3%) 2 (33.3%) 3 (33.3%)

Table 7.11 A voting process

	u_1	u_2	u_3	Result (ADD)
$a_1(0,1)$	1	0	1	2✓
$a_2(0,1)$	0	1	0	1
$a_3(0,1)$	0	0	0	0
$a_4(0,1)$	0	0	0	0

Each user is allowed to give only one vote—a decision is made on the basis of the ADD aggregation function

7.11 Voting

Voting has a structure that is similar to polls, however, there is a decision aspect in voting since a group or a community decides on which alternative should be chosen, i.e., there is a clear pragmatics of the decision outcome. Typical examples of the application of voting are the *player of the month* (e.g., in soccer), the *reporter of the year*, and the *president of a country*. In many cases, the goal of voting is to select one alternative (e.g., the president), however, there are also scenarios where more than one alternative is selected. For example, in the context of a best paper award: if majority voting is used for determining a best paper and there is a tie (depending on the process) multiple alternatives could be selected as best papers. In the context of elections, the determined ranking of the alternatives has clear pragmatics. For example, the identified person becomes the new president. Elections can be single shot or iterative and different tie-breaking rules can be applied (an example thereof can also be a new election). An example of a voting process is shown in Table 7.11.

7.12 Further Aspects of Choice Scenarios

Tie-Breaking Rules can help in situations where there is no clear winner but a decision has to be made. A tie-breaking *method* could be selected before the decision making process starts. This is used in situations where all group members agree on the method (or the method has to be accepted "per-se"). Elections are an example of a situation where a group (in this case, a community) has to decide, and the method is already pre-defined. Further related examples are voting procedures in (public) organizations and companies, for example, when selecting a new rector for a university, selecting a new pope, or selecting the new president of the labor union. Situations where groups try to determine the tie-breaking method ahead of time also occur in less business-related decision processes. For example, what is the impact (weight) of the expert jury compared to the opinion of the audience collected via SMS votes in a TV show (in a situation where the jury ranking combined with the ranking of the audience does not result in a clear winner). Similar situations occur when it comes to the selection of the best paper at a conference—example resolution strategies in this context can be a simple majority-rules vote or the average rating the paper received from the reviewers.

Further examples of tie-breaking rules are *toss a coin* (useful, for example, in the context of low-involvement items such as restaurants), *least misery* (useful in situations where two or more high-involvement alternatives have the same evaluation), *authority voting* (if a group did not agree on a specific decision rule and accepts the decision of a single authority), and *fairness* (in the context of repetitive decisions, users who were treated less favorably in previous decisions have priority). In many situations, a formalized and pre-defined rule for making a final decision does not exist, but the final decision is made on the basis of an internal discussion. In the "best paper" scenario this means that the members of the jury simply analyze all the given alternatives and articulate their preferences, for example, in terms of an initial ranking. Given that every jury member has defined his/her preferences, a discussion can be started with the overall goal of achieving consensus between the group members. Such group decision-making requires the inclusion of forums which allow the discussion and exchange of views regarding (dis)advantages of alternatives [18].

Multi-stage Processes Multi-stage choice is performed if the decision making task can be separated into multiple phases (e.g., first decide about the date of the holidays and then decide on the location and the hotel), or the process itself may consist of the phase of identifying a consideration set (a set of candidate items that could potentially be chosen) and then selecting items from the identified consideration set. Examples thereof are personnel selections where the relevant candidates are pre-selected and—on the basis of the consideration set—hiring interviews are conducted. Further related examples are *idea management* (e.g., the selection of a name for a new product or the selection of topics that should be chosen for the next project proposal), *strategic planning* (e.g., the definition and selection of new topics for professorships to be announced as open positions in the upcoming years).

Process Iterations Iterative decisions (in contrast to *single-shot* decisions) are typically made in the context of high-involvement items, i.e., items with a higher negative impact triggered by a suboptimal decision (compared to low-involvement items). In the context of such decisions, different types of conversational recommendation approaches, such as constraint-based recommendation and critiquing-based recommendation, are useful [4]. Decisions related to high-involvement items are typically made in an *iterative* fashion, i.e., before the decision is made, a couple of iterations in terms of evaluations and discussions are performed. Examples thereof are manifold. For instance, a family purchases a new car, a new CEO is hired for a company, a group of students selects a new shared apartment, or a new ERP system is purchased by a company. Gamification-based approaches are a special case of iterative decision making, for example, *Planning Poker* [13] is a consensus- and gamification-based approach to effort estimation (often used in requirements engineering [12]) where group members play cards. Each member holds a full deck of cards where each card represents a time effort ascending from, for example, 5 min to 1 month. After each group member has played a card (face-down), these cards are disclosed and the estimates of individual group members are discussed. After the discussion, each member plays another card until consensus is achieved. Examples of single-shot decisions are the selection of a restaurant and the selection of a movie to be watched on the weekend.

Degree of Participation Active participation is given if the persons providing preference feedback on the choice options are also engaged in the corresponding choice process (see also Chap. 2). This is the case with most of the aforementioned scenarios, i.e., decision makers are also engaged in the feedback process and provide their preferences with regard to the given set of alternatives. The exception to the rule are *polls* and *questionnaires*, where communities provide feedback to decision makers but often do not actively participate in the decision making process.

7.13 Conclusions and Research Issues

In this chapter, we discussed choice scenarios that go beyond those of previous chapters. We introduced a categorization of these scenarios along the dimensions of knowledge representation (items vs. parameters) and the inclusion of constraints. For a more in-depth understanding of these scenarios, we provided a couple of examples that show how to determine group recommendations. A couple of research issues also exist in this context. For example, the overall idea of group-based configuration is to engage user groups in configuration processes for complex products and services [11]. Examples of such scenarios are the group-based configuration of software release plans, the configuration of smart homes, and the configuration of holiday packages. In all of these scenarios, approaches are required that support solution search that takes into account the preferences of individual group members. A specific issue is how to guide heuristic search when confronted with the preferences of a group of users. Initial related work can be found, for example, in Polat-Erdeniz et al. [20]. Similar aspects play a role when supporting groups in achieving consensus in the case of contradicting preferences. The research issue to be solved is how to include social choice mechanisms into preference elicitation, and corresponding diagnosis and repair processes. Initial work on the inclusion of personalization into diagnosis processes is presented, for example, in [5, 9].

References

1. E. Alanazi, M. Mouhoub, B. Mohammed, A preference-aware interactive system for online shopping. Comput. Inform. Sci. **5**(6), 33–42 (2012)
2. M. Aldanondo, E. Vareilles, Configuration for mass customization: how to extend product configuration towards requirements and process configuration. J. Intell. Manuf. **19**(5), 521–535 (2008)
3. A. Falkner, A. Felfernig, A. Haag, Recommendation technologies for configurable products. AI Mag. **32**(3), 99–108 (2011)
4. A. Felfernig, R. Burke, Constraint-based recommender systems: technologies and research issues, in *ACM International Conference on Electronic Commerce (ICEC08)*, Innsbruck, Austria (2008), pp. 17–26

5. A. Felfernig, M. Schubert, G. Friedrich, M. Mandl, M. Mairitsch, E. Teppan, Plausible repairs for inconsistent requirements, in *21st International Joint Conference on Artificial Intelligence (IJCAI'09)*, Pasadena, CA (2009), pp. 791–796
6. A. Felfernig, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger, F. Reinfrank, Group decision support for requirements negotiation, in *UMAP 2011: Advances in User Modeling*. Lecture Notes in Computer Science, vol. 7138 (Springer, Berlin, 2011), pp. 105–116
7. A. Felfernig, M. Schubert, C. Zehentner, An efficient diagnosis algorithm for inconsistent constraint sets. Artif. Intell. Eng. Des. Anal. Manuf. **26**(1), 53–62 (2012)
8. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems, in *Recommendation Systems in Software Engineering* (Springer, Berlin, 2013), pp. 15–37
9. A. Felfernig, M. Schubert, S. Reiterer, Personalized diagnosis for over-constrained problems, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)*, Peking, China (2013), pp. 1990–1996
10. A. Felfernig, L. Hotz, C. Bagley, J. Tiihonen, *Knowledge-Based Configuration: From Research to Business Cases*, 1st edn. (Elsevier/Morgan Kaufmann Publishers, Burlington, 2014)
11. A. Felfernig, M. Atas, T.N. Trang Tran, M. Stettinger, Towards group-based configuration, in *International Workshop on Configuration 2016 (ConfWS'16)* (2016), pp. 69–72
12. A. Felfernig, M. Stettinger, A. Falkner, M. Atas, X. Franch, C. Palomares, OPENREQ: recommender systems in requirements engineering, in *RS-BDA17*, Graz, Austria (2017), pp. 1–4
13. N. Haugen, An empirical study of using planning poker for user story estimation, in *AGILE 2006* (2006), pp. 23–34
14. A. Jameson, S. Baldes, T. Kleinbauer, Two methods for enhancing mutual awareness in a group recommender system, in *ACM International Working Conference on Advanced Visual Interfaces*, Gallipoli, Italy (2004), pp. 447–449
15. G. Leitner, A. Fercher, A. Felfernig, K. Isak, S. Polat Erdeniz, A. Akcay, M. Jeran, Recommending and configuring smart home installations, in *International Workshop on Configuration 2016 (ConfWS'16)* (2016), pp. 17–22
16. J. Levin, B. Nalebuff, An introduction to vote-counting schemes. J. Econ. Perspect. **9**(1), 3–26 (1995)
17. J. Masthoff, Group modeling: selecting a sequence of television items to suit a group of viewers. User Model. User-Adap. Inter. **14**(1), 37–85 (2004)
18. T. Nguyen, F. Ricci, A chat-based group recommender system for tourism, in *Information and Communication Technologies in Tourism*, ed. by R. Schegg, B. Stangl (Springer, Cham, 2017), pp. 17–30
19. G. Ninaus, A. Felfernig, M. Stettinger, S. Reiterer, G. Leitner, L. Weninger, W. Schanil, INTELLIREQ: intelligent techniques for software requirements engineering, in *Prestigious Applications of Intelligent Systems Conference (PAIS)* (2014), pp. 1161–1166
20. S. Polat-Erdeniz, A. Felfernig, M. Atas, Cluster-specific heuristics for constraint solving, in *International Conference on Industrial, Engineering, Other Applications of Applied Intelligent Systems (IEA/AIE)*, Arras, France (2017), pp. 21–30
21. S. Qi, N. Mamoulis, E. Pitoura, P. Tsaparas, Recommending packages to groups, in *16th International Conference on Data Mining* (IEEE, Piscataway, 2016), pp. 449–458
22. S. Qi, N. Mamoulis, E. Pitoura, P. Tsaparas, Recommending packages with validity constraints to groups of users. Knowl. Inf. Syst. **54**, 1–30 (2017)
23. K. Schmid, Scoping software product lines, in *Software Product Lines – Experience and Research Directions* (Springer, Boston, 2000), pp. 513–532
24. M. Stumptner, An overview of knowledge-based configuration. AI Commun. **10**(2), 111–125 (1997)
25. E. Tsang, *Foundations of Constraint Satisfaction* (Academic Press, London, 1993)
26. M. Xie, L. Lakshmanan, P. Wood, Breaking out of the box of recommendations: from items to packages, in *4th ACM Conference on Recommender Systems*, Barcelona, Spain (2010), pp. 151–158

Chapter 8
Biases in Group Decisions

Alexander Felfernig, Müslüm Atas, Martin Stettinger, Thi Ngoc Trang Tran, and Gerhard Leitner

Abstract Decision biases can be interpreted as tendencies to think and act in specific ways that result in a systematic deviation of potentially rational and high-quality decisions. In this chapter, we provide an overview of *example decision biases* and show possibilities to counteract these. The overview includes (1) biases that exist in both single user and group decision making (decoy effects, serial position effects, framing, and anchoring) and (2) biases that especially occur in the context of group decision making (GroupThink, polarization, and emotional contagion).

8.1 Introduction

Research suggests that groups have the potential to outperform individuals in terms of decision quality [39, 47]. The collective memory of a group in many cases entails more *decision-relevant knowledge* than the memory of each individual group member. The same holds for *solution knowledge*: different group members are able to recall approaches to solve problems or take decisions from the past. However, groups often fail to achieve this goal [16]. One reason for explanation of this phenomenon is *decision biases*, which are defined as a *tendency to think and act in specific ways which results in deviations from rational and high-quality decisions* [3, 25, 37, 39]. Decision biases occur in single-person decisions as well as in group decisions. In this chapter, we summarize existing research related to decision biases in recommender systems (see, e.g., [22, 28]) and point out issues to be dealt with especially in the context of group decision making. For each of the mentioned biases, we first explain the basic underlying principle, provide examples, and then focus on specific aspects that have to be taken into account in group decision

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_8

scenarios. The inclusion of theories of human decision making into recommender applications is still a relatively young research field with a couple of open research issues [22].[1] The biases discussed in this chapter represent examples but in no way cover the complete set of biases investigated in psychological research [3, 25].

8.2 Decoy Effects

Within an item list, *decoy items* are alternatives inferior to all other items. Decoy items trigger a violation of the *regularity choice behavior* axiom which says: the inclusion of a completely inferior option can *not* change the probability that an existing option will be chosen [20, 27]. Superiority or inferiority of items is often measured by comparing item properties with regard to the distance to the corresponding optimal value. Although not attractive for the user, a decoy item can manipulate his/her selection behavior. If we assume that T is an item that should be pushed in terms of purchasing probability and C is a competitor item, the inclusion of a decoy item D can trigger the following situation: $P(T,\{T,C,D\}) > P(T,\{T,C\})$ where P(X,I) denotes the purchase probability of X given the *item set* I. Consequently, the regularity choice behavior axiom gets violated.

An example of a decoy effect is provided in Table 8.1: item D represents the *decoy item*, T represents a *target item* (item that should be pushed to increase purchase), and C represents a *competitor item*. In this case, users perceive an increased attractiveness of robot T due to the fact that it has a reliability that is similar to the optimum one provided by robot D. However, robot T has a significantly lower price which makes this option a compromise between optimal reliability and corresponding costs. This kind of effect is denoted as *compromise effect*. Further related effects are *asymmetric dominance* (the decoy item is outperformed by the target item in all dimensions) and *attraction effect* (the target item is only a little bit more expensive but completely outperforms the decoy item with regard to reliability). An overview of decoy effects, their role in recommendation scenarios, and how to counteract them is provided in [13, 28, 48, 49].

Felfernig et al. [15] show the existence of compromise effects in the financial service domain. Within the scope of a study that operated on a real-world financial service dataset, participants had to select items they would prefer to purchase given a

Table 8.1 Example of a *compromise effect*: item (robot) T is interpreted as a compromise since it has nearly the same reliability as D but a significantly lower price

Item (robot)	T	C	D
Price	3.000	1.500	5.000
Reliability	9	4.5	10

[1] See the ACM RecSys Workshop Series on *Human Decision Making and Recommender Systems*.

specific set of financial services. The reference set without decoy items consisted of *bonds*, *gold*, and *funds* whereas a decoy set consisted, for example, of *bonds*, *gold*, *funds*, and *shares*. In this setting, *shares* (the decoy item) make *funds* (the target item) a compromise alternative and thus help to increase the selection probability of *funds* (under the assumption that *shares* have a significantly higher risk compared to *funds* and often similar return rates). Note that decoy items do not only play a role when the goal is to push certain items from a list [28]. Decoy items can also help to reduce the *time needed to make a decision*, since they provide a good basis for resolving cognitive dilemmas, and they help to *increase the confidence in a decision* by providing an easy means to explain it [28].

An analysis of the existence of decoy effects in the context of *group decision making* is presented in [44]. The authors analyze the existence of decoy effects in employee selection among job applicants on the shortlist. The attributes used as a basis for comparison are *work sample score* and *promotability score*. The relevance of this analysis is even increased by the fact that only 26% of high-level employee selection decisions are made by a single person [43]. For decision scenarios with a low degree of interaction between different group members it seems to be clear that decoy effects already occur on the individual level and then are propagated to the group decision level. However, in the study of [44], study participants were sitting at the same table discussing alternative job applicants. The decoy effect was even increased in situations where study participants also had the defined role of being responsible for the chosen candidate (aspect of accountability). An explanation of this effect is that study participants had to think about *arguments and explanations* as to why they made a specific decision (proposal to choose a specific job candidate) more intensively. Decoy elements provide a basis for the construction of such explanations [28].

8.3 Serial Position Effects

Serial position effects (primacy/recency effects) can occur in different forms. First, if a recommendation list is presented to a user, items at the beginning and the end of this list are investigated more intensively—a related study is presented in Murphy et al. [36] where users were confronted with a list of weblinks. Second, serial position effects have a *cognitive dimension* in terms of the probability of being able to memorize items included in a list [38].

The impact of serial position effects on user selection behavior has been investigated in recommendation settings addressing single users (see Fig. 8.1): Felfernig et al. [12] report that item attributes shown to a user in a sequence have a higher probability of being recalled if they are mentioned at the beginning or the end of the sequence. This holds true for popular/well-known properties, and also for those that are less popular/less well-known. The item attributes recalled by a user also have an impact on his/her selection behavior, i.e., item attributes presented at the beginning and the end of a dialog are used as selection criteria with

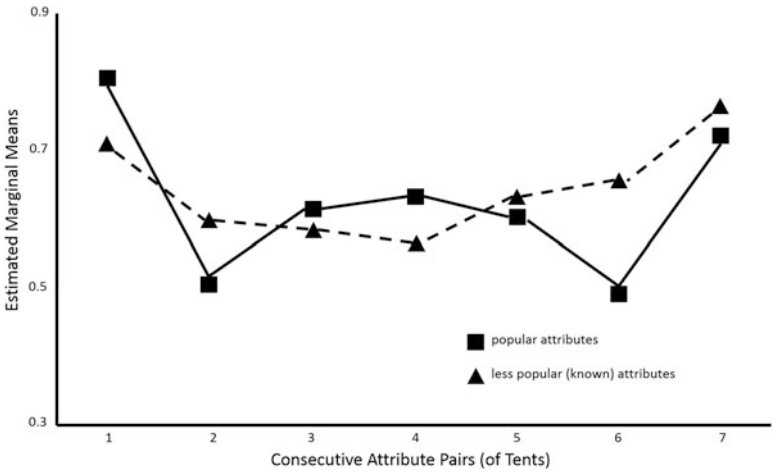


Fig. 8.1 Serial position effects when item attributes are presented in a sequence [12]. Item attributes presented at the beginning and the end of a list are recalled more often than those in the middle. This holds in situations where popular attributes are positioned at the beginning and the end of a list (solid line) but also in situations where less known/popular attributes were mentioned at the beginning and at the end of the list (dashed line)

a higher probability. A similar effect can be observed when analyzing argumentation sequences related to items: if positive arguments are positioned at the beginning and the end of an item evaluation, the evaluation of the item tends to be better [46].

The order of items in a list has also an impact on decision making in the context of *group decision scenarios*. Highhouse and Gallo [19] show that the order in which candidates are interviewed has an influence on which candidates are finally chosen. Specifically, recency effects were observed, i.e., job candidates interviewed at the end of the selection process had a higher probability of being selected. Stettinger et al. [46] present the CHOICLA group decision support environment that is based on social choice-based preference aggregation mechanisms for groups [29]. The environment supports different types of preference definition mechanisms which range *from* a star-based rating that can be used for simple items such as movies *to* items that can be evaluated using interest dimensions on the basis of multi-attribute utility theory (MAUT) [50]. The role of serial position effects in CHOICLA-based group decisions is discussed in [46] where the impact of the ordering of positive and negative arguments regarding an item is evaluated. Study participants were organized into groups of five to six persons who had to evaluate restaurants they would like to visit for a dinner. The *variation points* in the study were (1) the used *rating scales* (*5-star* vs. *MAUT* rating scale based on the interest dimensions *ambience*, *price*, *quality*, and *location* of the restaurant) and (2) *two different sequences of a set of arguments* in a restaurant review. In one review version, the positive arguments were positioned at the beginning and at the end of the evaluation (*positive salient* version), in the other version the negative arguments

were positioned at the beginning and at the end (*negative salient* version). One major insight of the study was that MAUT-based preference elicitation can counteract decision biases since there were no significant differences in the evaluation of the items in the positive and negative salient version. In the case of star ratings, the overall item evaluation in the negative salient version was significantly lower than in the positive salient version.

8.4 Framing

The way in which an alternative is presented to the user can influence a user's decision making behavior [24]. According to *prospect theory* [24], decision alternatives are evaluated with regard to potential losses and gains, where the impact of losses is evaluated higher than the impact of gains (user-specific asymmetric evaluation function). An example of framing is *price framing* [7]: two companies (x and y) sell wood pellets. Company x describes its product as *pellets for €24.50 per 100 kg with a €2.50 discount if the customer pays with cash* whereas company y provides the description *€22.0 per 100 kg, and charges a €2.50 surcharge if the customer uses a credit card*. Company x rewards buyers with a discount which would trigger an increased purchasing of items x, even though both offers are equivalent from the cost perspective. Framing effects can be reduced, for example, if explanations are required for a final decision [33]. These effects occur more often when decision heuristics are used, compared to situations where persons follow an analytic processing style to make a decision [31]. Framing effects also exist in *group decision scenarios* [9, 32, 40]. In gain situations, there is a tendency of more risk-awareness whereas in loss situations there is an increased risk-seeking tendency [9].

8.5 Anchoring

Anchoring represents a tendency to rely too heavily on the first information (the anchor) received within the scope of a decision process. Anchoring effects trigger decisions, which are influenced by a group member who first articulated his/her preferences [21, 45]. Related results in decision support scenarios are confirmed by social-psychological studies which show the relationship between decision quality and the visibility of individual user preferences [34]. It was shown that hidden preferences in early decision phases of a group can increase the amount of decision-relevant information exchanged by group members, and that a higher degree of information exchange correlates with a higher quality of related decision outcomes. Thus, early preference visibility triggers a *confirmation bias* where a group searches for information that confirms the initial views of group members and a *shared information bias* which reflects the situation where a group focuses on discussing information available to all group members but not on figuring out and sharing new

decision-relevant information. In group decision settings, there is also a tendency to not consider conflict-inducing information related to a preferred alternative if the group members providing this information are in the minority [26].

Anchoring effects have also been analyzed in the context of recommender systems. For example, in the context of collaborative filtering recommender systems, reference ratings of other users shown to the current user within the scope of an item evaluation (rating) process have an impact on the current user's ratings [1, 2, 10]. Manipulated higher average ratings shown to the current user trigger higher user ratings, manipulated lower ratings have the opposite effect. Furthermore, adapting the preference definition interface (e.g., from a 5-star rating scale to a binary one) can help to counteract such biases. For recommender user interfaces, this also means that item rating tasks should not include available rating information from other users [10].

Felfernig et al. [14] show the existence of anchoring effects in group-based software engineering scenarios. In this context, preference shifts were detected when software teams engaged in a university course on software requirements engineering had to make decisions regarding different *aspects of their software project* (e.g., *type of evaluation*, *presentation*, *programming language*, and *database technology*). Stettinger et al. [45] also analyze group decision scenarios in software engineering. In this context, they focus on requirements engineering where groups of software developers have to complete a *requirements prioritization task* in terms of deciding which requirements should be implemented in their software project. The existence of anchoring effects could be shown: the earlier the preferences of individual group members were shown to other users, the higher the probability of the occurrence of anchoring effects (see Fig. 8.2). The earlier user-individual preferences are disclosed, the lower the perceived quality of the decision outcome and of the perceived decision support.

Late preference disclosure increases discussion intensity and information exchange between group members, which has a positive impact on decision quality [8, 18]. Schulz-Hardt [42] point out that *overconfidence* within a group can be triggered by a shared information bias. Atas et al. [5] show the application of recommendation technologies in group decision scenarios to foster information exchange between group members. In the presented study, recommendations with different degrees of diversity were delivered to group members—the degree of information exchange between group members increased with an increased degree of recommendation diversity.

8.6 GroupThink

GroupThink [11, 23] occurs in situations where members of a cohesive group have a clear preference in terms of avoiding conflicts and maintaining unanimity, and are not primarily interested in analyzing existing decision alternatives [23]. In such situations, groups often fail to analyze relevant alternatives in detail, do

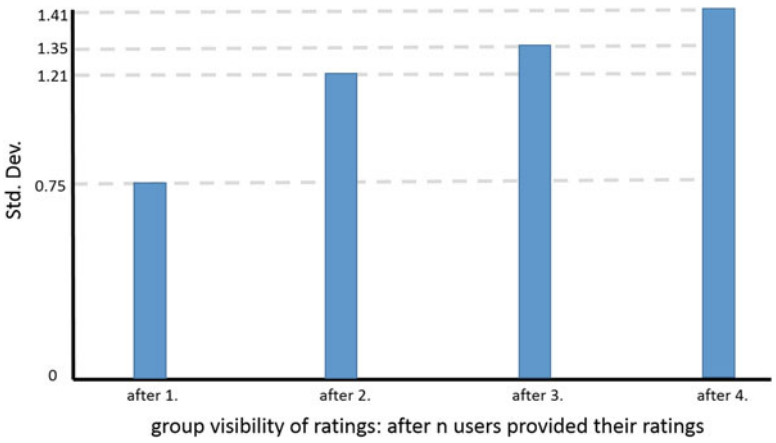


Fig. 8.2 Anchoring effect in requirements prioritization [45]. The earlier individual preferences are shown to other users (e.g., after 1 user has defined his/her preferences), the less ratings of users differ (measured in terms of standard deviation)

not adequately take risk into account, and do not focus on the exchange of additional decision-relevant information. GroupThink can be increased by encouraging conformity within a group [4] (when the majority of group members expresses an opinion different from an individual [30]), by an unwillingness to analyze existing alternatives, and by decision environments that do not tolerate *dissent*, a major ingredient and precondition for fostering information exchange between group members. Finally, GroupThink also increases the *confirmation effect*, i.e., the tendency to favor and recall information units in a way that confirms existing preferences [23, 39]. There are different ways to avoid GroupThink. Leaders should not articulate their opinion to other group members before discussing relevant alternatives in detail. Experts outside the group should be integrated in order to stimulate diverse opinions, related debates, and information exchange which are crucial for high-quality decision making. As already mentioned, an approach to exploit recommender systems functionality to stimulate information exchange in group decision processes is presented in [5].

8.7 Emotional Contagion

Emotional contagion describes the influence of the affective state of an individual on the affective state of other individuals within a group [6]. This effect can have a positive or a negative impact on overall satisfaction with a group decision [6]. The strength of the effect also depends on the item domain. For example, emotional contagion is more likely to happen in a music recommender system than in TV watching, since people are often more aware of others when not solely staring at

a screen [30]. An approach to counteract this effect in group decision scenarios is not to allow information exchange between group members in the very early phase of a group decision process. On the level of recommendation algorithms, emotional contagion and different personality aspects can be used to improve, for example, the prediction quality of the group recommender [30, 41] (see also Chap. 9).

8.8 Polarization

There is often a tendency in groups to shift towards more extreme decisions compared to the original positions/preferences of the individual group members [9]. For example, in group-based investment decisions it can be the case that—although individual group members prefer an average risk investment strategy—the final chosen risk level is higher than the preferred risk levels of individual group members. This tendency to shift towards more extreme decisions in the context of group decision making is denoted as *group polarization* [35]. Group decisions can be more risky if the original opinions of individual group members tend to be risky (*risky shift*). Vice-versa, there also exists a *cautious shift* if group individuals are supporting more conservative alternatives [17]. In the context of group investment decisions, Cheng and Chiou [9] show that group decisions appear to be more cautious in gain situations and more risky in loss situations. A reduction of such a polarization effect can be achieved by including dissent, which also helps to trigger discussions more related to potential negative impacts of a decision. Recommender systems aware of polarization can adapt, for example, the utility estimates of recommendations and provide corresponding explanations. To the best of our knowledge, such concepts have not been integrated into recommender systems up to now.

8.9 Conclusions and Research Issues

Although groups have the potential to perform better than individuals in solving decision tasks, suboptimal decisions are made due to different types of biases (e.g., decoy effects, serial position effects, framing, anchoring, GroupThink, emotional contagion, and polarization). Without claiming to have provided a complete discussion of possible biases in group decision making, we have emphasized biases that have been analyzed in single-user recommendation contexts and, to a lesser extent, in the context of group decision making. There exist a couple of research contributions related to the analysis of decision biases, especially with regard to their impact on the development of recommender applications. A major focus of existing work in the field is to show the existence of such biases in different item domains and recommendation contexts. However, it is even more important to develop approaches that help *counteract* these effects on different levels, such as

recommender algorithms and recommender user interfaces. Avoiding biases helps to increase decision quality; consequently, related research contributions have a potentially high impact on the quality of future group recommender systems [45].

References

1. G. Adomavicius, J. Bockstedt, S. Curley, J. Zhang, Recommender systems, consumer preferences, and anchoring effects, in *RecSys 2011 Workshop on Human Decision Making in Recommender Systems* (2011), pp. 35–42
2. T. Amoo, H. Friedman, Do numeric values influence subjects responses to rating scales? J. Int. Mark. Mark. Res. **26**, 41–46 (2001)
3. D. Arnott, A taxonomy of decision biases. Technical Report, Monash University, Caulfield East, VIC (1998), pp. 1–48
4. S. Asch, Effects of group pressure upon the modification and distortion of judgements, in *Groups, Leadership, and Men* (1951), Swarthmore College, pp. 177–190
5. M. Atas, A. Felfernig, M. Stettinger, T.N. Trang Tran, Beyond item recommendation: using recommendations to stimulate knowledge sharing in group decisions, in *9th International Conference on Social Informatics (SocInfo 2017)*, Oxford, 2017, pp. 368–377
6. S. Barsade, The ripple effect: emotional contagion and its influence on group behavior. Adm. Sci. Q. **47**(4), 644–675 (2002)
7. M. Bertini, L. Wathieu, The framing effect of price format. Working Paper, Harvard Business School (2006)
8. F. Brodbeck, R. Kerschreiter, A. Mojzisch, D. Frey, S. Schulz-Hardt, The dissemination of criticial, unshared information in decision making groups: the effects of pre-discussion dissent. Eur. J. Soc. Psychol. **32**(1), 35–56 (2002)
9. P. Cheng, W. Chiou, Framing effects in group investment decision making: role of group polarization. Psychol. Rep. **102**(1), 283–292 (2008)
10. D. Cosley, S. Lam, I. Albert, J. Konstan, J. Riedl, Is seeing believing? How recommender system interfaces affect users' opinions, in *Proceedings of CHI'03* (2003), pp. 585–592
11. J. Esser, Alive and well after 25 years: a review of groupthink research. Organ. Behav. Hum. Decis. Process. **73**(2–3), 116–141 (1998)
12. A. Felfernig, G. Friedrich, B. Gula, M. Hitz, T. Kruggel, R. Melcher, D. Riepan, S. Strauss, E. Teppan, O. Vitouch, Persuasive recommendation: serial position effects in knowledge-based recommender systems, in *2nd International Conference of Persuasive Technology (Persuasive 2007)*. Lecture Notes in Computer Science, vol. 4744 (Springer, Berlin, 2007), pp. 283–294
13. A. Felfernig, B. Gula, G. Leitner, M. Maier, R. Melcher, S. Schippel, E. Teppan, A dominance model for the calculation of decoy products in recommendation environments, in *AISB Symposium on Persuasive Technologies*, Aberdeen, 2008, pp. 43–50
14. A. Felfernig, C. Zehentner, G. Ninaus, H. Grabner, W. Maalej, D. Pagano, L. Weninger, F. Reinfrank, Group decision support for requirements negotiation, in *Advances in User Modeling*. Lecture Notes in Computer Science, vol. 7138 (Springer, Berlin, 2011), pp. 105–116
15. A. Felfernig, E. Teppan, K. Isak, Decoy effects in financial service E-sales systems, in *RecSys'11 Workshop on Human Decision Making in Recommender Systems (Decisions@RecSys'11)*, Chicago, IL, 2011, pp. 1–8
16. D. Forsyth, *Group Dynamics* (Thomson Higher Education, Belmont, 2006)
17. C. Fraser, C. Gouge, M. Billig, Risky shifts, cautious shifts, and group polarization. Eur. J. Soc. Psychol. **1**(1), 7–30 (1971)
18. T. Greitemeyer, S. Schulz-Hardt, Preference-consistent evaluation of information in the hidden profile paradigm: beyond group-level explanations for the dominance of shared information in group decisions. J. Pers. Soc. Psychol. **84**(2), 332–339 (2003)

19. S. Highhouse, A. Gallo, Order effects in personnel decision making. Hum. Perform. **10**, 31–46 (1997)
20. J. Huber, J. Payne, C. Puto, Adding asymmetrically dominated alternatives: violations of regularity and the similarity hypotheses. J. Consum. Res. **9**, 90–98 (1982)
21. K. Jacowith, D. Kahneman, Measures of anchoring in estimation tasks. Pers. Soc. Psychol. Bull. **21**(11), 1161–1166 (1995)
22. A. Jameson, M. Willemsen, A. Felfernig, M. de Gemmis, P. Lops, G. Semeraro, L. Chen, Human decision making and recommender systems, in *Recommender Systems Handbook*, 2nd edn., ed. by F. Ricci, L. Rokach, B. Shapira (Springer, Berlin, 2015), pp. 611–648
23. I. Janis, *Victims of Groupthink* (Houghton-Mifflin, Boston, 1972)
24. D. Kahneman, A. Tversky, Prospect theory: an analysis of decision under risk. Econometrica **47**(2), 263–291 (1979)
25. N. Kerr, G. Kramer, R. MacCoun, Bias in judgement: comparing individuals and groups. Psychol. Rev. **103**(4), 687–719 (1996)
26. J. Levine, E. Russo, Impact of anticipated interaction on information acquisition. Soc. Cogn. **13**(3), 293–317 (1995)
27. R. Luce, *Individual Choice Behavior* (Wiley, New York, 1959)
28. M. Mandl, A. Felfernig, E. Teppan, M. Schubert, Consumer decision making in knowledge-based recommendation. J. Intell. Inf. Syst. **37**(1), 1–22 (2010)
29. J. Masthoff, Group recommender systems: combining individual models, in *Recommender Systems Handbook* (Springer, New York, 2011), pp. 677–702
30. J. Masthoff, A. Gatt, In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model. User-Adap. Inter. **16**(3–4), 281–319 (2006)
31. T. McElroy, J. Seta, Framing effects: an analytic-holistic perspective. J. Exp. Soc. Psychol. **39**(6), 610–617 (2003)
32. K. Milch, E. Weber, K. Appelt, M. Handgraaf, D. Krantz, From individual preference construction to group decisions: framing effects and group processes. Organ. Behav. Hum. Decis. Process. **108**(2), 242–255 (2009)
33. P. Miller, N. Fagley, The effects of framing problem variations, and providing rationale on choice. Pers. Soc. Psychol. Bull. **17**(5), 517–522 (1991)
34. A. Mojzisch, S. Schulz-Hardt, Knowing others preferences degrades the quality of group decisions. J. Pers. Soc. Psychol. **98**(5), 794–808 (2010)
35. S. Moscovici, M. Zavalloni, The group as a polarizer of attitudes. J. Pers. Soc. Psychol. **12**(2), 125–135 (1969)
36. J. Murphy, C. Hofacker, R. Mizerski, Primacy and recency effects on clicking behavior. J. Comput.-Mediat. Commun. **11**(2), 522–535 (2006)
37. M. Neale, M. Bazerman, G. Northcraft, C. Alperson, Choice shift effects in group decisions: a decision bias perspective. Int. J. Small Group Res. **2**(1), 33–42 (1986)
38. E. Nipher, On the distribution of errors in numbers written from memory. Trans. Acad. Sci. St. Louis **3**, CCX–CCXI (1878)
39. J. Osmani, Heuristics and cognitive biases: can the group decision-making avoid them? Acad. J. Interdisciplinary Stud. **5**(3), 225–232 (2016)
40. P. Paese, M. Bieser, M. Tubbs, Framing effects and choice shifts in group decision making. Organ. Behav. Hum. Decis. Process. **56**(1), 149–165 (1993)
41. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, Social factors in group recommender systems. ACM Trans. Intell. Syst. Technol. **4**(1), 8:1–8:30 (2006)
42. S. Schulz-Hardt, D. Frey, C. Lüthgense, S. Moscovici, Biased information search in group decision making. J. Pers. Soc. Psychol. **78**(4), 655–669 (2000)
43. V. Sessa, R. Kaiser, J. Taylor, R. Campbell, Executive selection: a research report on what works and what doesn't. Center for Creative Leadership (1998)
44. J. Slaughter, J. Bagger, A. Li, Context effects on group-based employee selection decisions. Organ. Behav. Hum. Decis. Process. **100**(1), 47–59 (2006)
45. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, Counteracting anchoring effects in group decision making, in *23rd Conference on User Modeling, Adaptation, and Personalization (UMAP'15)*, Dublin, 2015. Lecture Notes in Computer Science, vol. 9146 (2015), pp. 118–130

46. M. Stettinger, A. Felfernig, G. Leitner, S. Reiterer, M. Jeran, Counteracting serial position effects in the CHOICLA group decision support environment, in *20th ACM Conference on Intelligent User Interfaces (IUI2015)*, Atlanta, GA, 2015, pp. 148–157
47. C. Sunstein, R. Hastie, *Wiser: Getting Beyond Groupthink to Make Groups Smarter* (Harvard Business Review Press, Boston, 2014)
48. E. Teppan, A. Felfernig, Minimization of decoy effects in recommender result sets. Web Intell. Agent Syst. **1**(4), 385–395 (2012)
49. G. Tomer, Implications of perceived utility on individual choice and preferences: a new framework for designing recommender system. J. Int. Technol. Inf. Manag. **24**(2), 55–64 (2015)
50. D. Winterfeldt, W. Edwards, *Decision Analysis and Behavioral Research* (Cambridge University Press, Cambridge, 1986)

Chapter 9
Personality, Emotions, and Group Dynamics

Marko Tkalčič, Amra Delić, and Alexander Felfernig

Abstract The methods and techniques introduced in the previous chapters provide a basic means to aggregate the preferences of individual group members and to determine recommendations suitable for the whole group. However, preference aggregation can go beyond the integration of the preferences of individual group members. In this chapter, we show how to take into account the aspects of *personality*, *emotions*, and *group dynamics* when determining item predictions for groups. We summarize research related to the integration of these aspects into recommender systems, and provide some selected examples.

9.1 Personality and Emotions

Research has already demonstrated that various properties of recommender systems (e.g., prediction quality) can be improved by taking into account the aspects of personality [9] and emotions [22, 38]. In this chapter, we show how these aspects can be considered in group recommendation scenarios. In contrast to single user recommenders [8], group dynamics [11], i.e., the way group members interact (e.g., in terms of communicating opinions) have to be taken into account [22, 30].

Personality

According to McCrae and John [24], personality reflects *individual differences in emotional, interpersonal, experiential, attitudinal, and motivational styles*. An overview of different models of personality especially in the context of offering personalized services is given by Matz et al. [23, 37]. The traditional approach to the *acquisition of personality information* are (obtrusive) questionnaires [17, 18] which should not be the first choice when following the objective of integrating personality

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_9

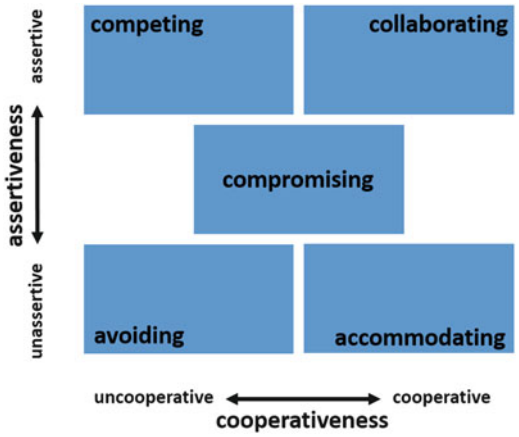


Fig. 9.1 *Thomas-Kilmann* (*TKI*) model of conflict resolution styles [18]

aspects into recommender systems. Such questionnaires are often employed in the context of user studies—see, for example, Quijano-Sanchez et al. [29]. As an alternative, there are a couple of methods for estimating personality in an unobtrusive fashion. For example, on the basis of social media features such as the number of *Twitter* followers and followees [28] (e.g., an above average number of followers and followees is correlated with *extraversion*), FACEBOOK likes [13, 19] (e.g., music from Leonard Cohen is correlated with *openness*), and color-based, low-level features from INSTAGRAM pictures [36] (e.g., *extroverted people* like a lot of green color). Related work is presented in Neidhardt et al. [26] who show how to elicit travel-related personality information in single user recommender systems. Users had to select pictures which were used to infer tourism-related personality factors such as *sun and chill-out*, *action and fun*, *nature and recreation*, etc. *Unobtrusive methods* come along with a trade-off in terms of lower algorithm accuracy, however, recent research has shown that using a combination of sensors and social media traces with advanced machine learning can yield acceptable predictive quality [10, 35]. Importantly, off-the-shelf solutions such as WATSON PERSONALITY INSIGHTS (e.g., personality prediction through written texts) are available.

In this chapter, we will use the *Thomas-Kilmann Conflict Style Model* (*TKI model*) [18] of personality as a basis for our working examples (see Fig. 9.1). In contrast to other models that primarily take into account characteristics of individual users, the advantage of this model is that it focuses on the interaction between group members. In this context, it serves as a basis for the provision of conflict resolution styles applicable in specific group settings. The TKI model differentiates between the two aspects of *cooperation* (*low .. high*) and *assertion* (*low .. high*). Combinations of these two aspects lead to different personality types which are *competing* (assertive and uncooperative, own concerns are pursued at the expense of other group members), *collaborating* (cooperative and assertive, the goal is to find a solution that satisfies the concerns of all group members),

compromising (moderate in cooperativeness and assertiveness, focus on finding trade-offs/solutions acceptable for all group members), *avoiding* (not assertive and not cooperative, no concerns are pursued), and *accommodating* (cooperative and not assertive, focus is on primarily satisfying concerns of other group members).

Emotions

Emotions can be regarded as *base affective occurrences that are usually triggered by a stimulus*, for example, if one wins a race (s)he usually gets happy. There are different models of emotion which will not be discussed in this chapter—for a related overview we refer to D'Errico and Poggi [7]. Typical dimensions covered by base models of emotions are *anger*, *disgust*, *fear*, *happiness*, *sadness*, and *surprise*. Due to their direct measurability, *valence* and *arousal* are often used to infer emotional categories [1, 25]. For example, *anger* is related to a high arousal and low valence. Similar to personality, emotions can also be measured on the basis of self-assessment questionnaires. Also, off-the-shelf tools (e.g., AFFECTIVA) support the automated detection of emotions from facial expressions, skin conductance, EEG (electroencephalography) signals, etc. A survey of existing techniques in automated emotion detection is given by Schuller [35]. Ho et al. [16] introduce a single user movie recommender system that takes into account the emotional states of users. Emotional states are determined with regard to colors that have to be chosen by users. Depending on this feedback, an emotional state can be determined and recommendations can be made based on items consumed by users in a similar emotional state (e.g., on the basis of collaborative filtering). Thus, emotions can be interpreted as a contextual dimension. Emotions in contextual recommender systems have also been analyzed by Zheng et al. [39]. The outcome of their study was that emotions can help to improve the predictive performance of recommendation algorithms.

9.2 Group Dynamics

Group dynamics account for processes and outcomes that occur in group settings [4, 11]. Social sciences research has shown that group decisions are not always rational and cannot always be deduced from (explained solely by) the preferences of individual group members. Consequently, supporting group decision processes on the basis of group recommendation technologies also requires knowledge of group dynamics. In the following, we discuss the aspects of *emotional contagion* and *conformity* which are the major influential aspects to be taken into account when analyzing group decision processes. We discuss these aspects in the light of existing research in group recommender systems.

Emotional Contagion Emotional contagion (see Chap. 8) reflects processes where the emotional state of one group member influences the emotional state of other group members [2, 15]. In this context, emotions can (1) be transferred "as-is" (e.g., the happiness of one group member makes other group members happy as well) or (2) trigger a *counter-contagion* (e.g., due to competitive situations among group members). In this case, the happiness of one person makes another person annoyed. Usually, this effect occurs automatically, unintentionally, and uncontrollably. Emotional contagion also occurs in online groups where it has been shown that, for example, FACEBOOK users confronted with positively formulated posts also generated more positive ones and vice-versa [20].

Emotional contagion has also been taken into account in the context of group recommendation scenarios [22]. If, for example, one group member is dissatisfied with a recommendation, it can be expected that her disappointment has a negative influence on the other group members. This in turn decreases the overall group satisfaction even though other group members would have enjoyed a given recommendation. In the work of Masthoff and Gatt [22], group recommendations are determined for item sequences (TV programs). In such a context, the satisfaction of an individual is not only a function of the currently-recommended item, but also a function of the items presented earlier. Recommendations of sequences to groups is outside the scope of this chapter—for an in-depth discussion of how to integrate the concept of emotional contagion on an algorithmic level, we refer to Masthoff and Gatt [22].

Conformity Conformity can be interpreted as a change in opinion, judgement, or action to match the opinions, judgements, or actions of other group members or to match the group's normative standards [11]. In the context of group recommender systems, conformity knowledge can be used to better predict the preparedness of individual group members to adapt their initial evaluations. In Masthoff and Gatt [22], a function to estimate the degree of conformity of a specific group member is based on factors such as *size of the subgroup with a different opinion*, *number of persons outside that subgroup*, and *difference between the individual's opinion and the opinion of the subgroup*. Berkovsky and Freyne [3] introduced a model of *influence* of specific group members that is based on rating counts. For example, the higher the share of ratings of one family member in relation to the number of ratings of all family members, the higher his/her influence. In this context, it is assumed that the lower the influence, the higher the preparedness of persons to adapt their evaluations (and the higher the conformity level). Quintarelli et al. [32] introduce a measure of influence that is based on the idea that the more often the individual preference of a group member appears as result in the final group choice, the higher the influence of this group member. Finally, Nguyen and Ricci [27] analyze three conformity types within the scope of an empirical study: (1) group members do not change their preferences (*independence*), (2) preferences of group members tend to become similar (*conversion*), and (3) preferences become more divergent (*anti-conformity*).

An approach to combine *personality* information with *conformity* in the context of group recommendations has been introduced by Quijano-Sanchez et al. [29]. The presented approach is able to estimate the extent of conformity susceptibility (on the basis of a trust measure) of a specific user and to take this information into account when generating group recommendations. The personality model used in [29] is based on the aforementioned *TKI model* [18]. In the following section, we provide an example of the group recommendation approach presented by Quijano-Sanchez et al. [29]. For an in-depth discussion of the integration of emotional contagion into group recommendation processes, we refer to Masthoff and Gatt [22].

9.3 Example: Taking into Account Personality and Conformity

In order to show how to integrate aspects of *group dynamics* into group recommendation processes, we give an example that is based on the approach presented in [29]. The ratings shown in Table 9.1 have to be considered as the user-specific item rating predictions determined by the underlying recommender system.

Quijano-Sanchez et al. [29] used a *TKI test* consisting of 30 questions [18] to categorize the personality of individual group members. Depending on the determined scores (categorized as *low* or *high*), corresponding *assertiveness* and *cooperativeness* values can be determined. For example, high estimates for competing and collaborating modes result in high assertiveness values (see Table 9.2).

User-specific assertiveness and cooperativeness evaluations can be represented as the sum of the five personality modes [29, 33]. After completion of the questionnaire, the degree of cooperativeness and assertiveness can be determined for each user. For the approach used to determine the high/low categories shown in Table 9.3, we refer to [29, 33]. Combining the information contained in Tables 9.2

Table 9.1 Example predictions of user × item ratings

User	t_1	t_2	t_3	t_4	t_5
u_1	2	4	5	1	3
u_2	3	2	3	4	5
u_3	1	3	5	2	1

Table 9.2 Coefficients for determining assertiveness and cooperativeness [33]

	Assertiveness		Cooperativeness	
TKI mode	High	Low	High	Low
Competing	0.375	−0.075	−0.15	0
Collaborating	0.375	−0.075	0.375	−0.075
Compromising	0	0	0	0
Avoiding	−0.375	0.075	−0.375	0.075
Accommodating	−0.15	0	0.375	−0.075

Table 9.3 Personality scores of example users u_i with regard to (*TKI* conflict resolution types [18])

User	Competing	Collaborating	Compromising	Avoiding	Accommodating
u_1	High	High	Low	Low	Low
u_2	High	Low	Low	Low	Low
u_3	Low	High	High	Low	High

Table 9.4 User-specific estimates of *assertiveness* and *cooperativeness* and corresponding *conflict mode weight* (*cmw*)—see Formula 9.1

User	Assertiveness	Cooperativeness	cmw(u_i)
u_1	$0.375 + 0.375 + 0$ $+0.075 + 0 = 0.825$	$-0.15 + 0.375 + 0 + 0.075$ $-0.075 = 0.225$	0.8
u_2	$0.375 - 0.075 + 0$ $+0.075 + 0 = 0.375$	$-0.15 - 0.075 + 0+$ $0.075 - 0.075 = -0.225$	0.8
u_3	$-0.075 + 0.375 + 0$ $+0.075 - 0.15 = 0.225$	$0 + 0.375 + 0 + 0.075$ $+0.375 = 0.825$	0.2

and 9.3 results in the estimates of *assertiveness* and *cooperativeness* depicted in Table 9.4. For example, group member u_1 is highly assertive whereas u_3 is highly cooperative.

The group recommendation approach then is based on the idea of encapsulating *assertiveness* (selfishness) and *cooperativeness* estimates of group members into the determination of rating predictions. The first step in this context is to determine the *conflict mode weight* (*cmw*) (see Formula 9.1) which represents the predominant behavior of a group member (on a scale -1 .. $+1$). The underlying assumption is that the higher the *cmw* value, the stronger the influence of that group member (higher assertiveness and lower cooperativeness). The *cmw* values determined for the group members in our example setting are depicted in Table 9.4.

$$cmw(u) = \frac{1 + assertiveness(u) - cooperativeness(u)}{2} \tag{9.1}$$

Personality-Enhanced Rating Prediction Using the *cmw* value, we are able to determine a personality-enhanced item rating prediction for each group member u ($p_{pers}(u, i)$). This rating serves as an input for determining the item rating prediction for the whole group ($g_{pers}(G, i)$)—see Formulae 9.2 and 9.3. In this context, $p(u_a, i)$ denotes the item-i rating predicted for user u_a determined by a recommendation algorithm. The underlying idea is that the original item ratings are adapted depending on the *cmw* value, i.e., users assumed to not be prepared to downgrade their ratings receive a corresponding positive adaptation.

$$g_{pers}(G, i) = \frac{\Sigma_{u \in G} p_{pers}(u, i)}{|G|} \tag{9.2}$$

$$p_{pers}(u_a, i) = \frac{\Sigma_{u \in G(u \neq u_a)} p(u_a, i) + (cmw(u_a) - cmw(u))}{|G| - 1} \tag{9.3}$$

Table 9.5 Example *personality-based rating predictions* (ratings in 0..5)

	p(u,i)					p_{pers}(u,i)				
User	t_1	t_2	t_3	t_4	t_5	t_1	t_2	t_3	t_4	t_5
u_1	2	4	5	1	3	2.3	4.3	5.0	1.3	3.3
u_2	3	2	3	4	5	3.3	2.3	3.3	4.3	5.0
u_3	1	3	5	2	1	0.4	2.4	4.4	1.4	0.4
AVG	2	3	4.3	2.3	3	2	3	4.2	2.3	2.9

5.0 is assumed to be the ceiling, i.e., a potential 5.3 rating is downgraded to 5.0. Predictions are determined on the basis of Formula 9.3

Table 9.6 Example trust relationships among group members $u_i \in G$

User	u_1	u_2	u_3
u_1	1.0	0.5	0.6
u_2	0.5	1.0	0.2
u_3	0.6	0.2	1.0

For simplicity we assume symmetry, i.e., $trust(u_i, u_j) = trust(u_j, u_i)$

Applying Formulae 9.2 and 9.3 results in the adapted item rating predictions depicted in Table 9.5. For example, group member u_3 has a low *cmw* value compared to group members u_1 and u_2 (see Table 9.4). As a consequence, the rating predictions for u_3 are downgraded whereas those of u_1 and u_2 get increased.

Influence-Based Rating Prediction The idea of influence-based rating prediction [31], i.e., rating prediction based on social influence, is to take into account both the *personality* of group members and *trust* relationships between group members (Table 9.6). In this context, trust between two users ($t(u_1, u_2)$) is defined as the weighted sum over a set of n factors f_i that are selected to act as indicators of trust relationships between group members (u_1 and u_2 in Formula 9.4).

$$t(u_1, u_2) = \Sigma_{i=1}^{n} w_i \times f_i(u_1, u_2) \tag{9.4}$$

Examples of such factors are *distance in a social network* (e.g., if two users are friends in a social network or have friends in common), *intensity of the relationship* (e.g., how often a user name appears on the wall of the other user), and *duration* (how long have two users known each other).[1]

On the basis of the identified trust level (Formula 9.4), influence-based rating prediction can be performed [31]. In this context, it is assumed that group members may adapt their ratings depending on the ratings of their friends. A rating prediction that integrates both, the level of *trust* and the *personality* of individual group members is defined by Formula 9.5. The positive or negative adaptation of a group member's u_a original rating is defined by the average positive or negative difference

[1] For a detailed discussion of these factors, we refer to [31].

Table 9.7 Example *influence-based rating predictions* (ratings in 0..5)

	p(u,item)					p_{pers}(u,item)				
User	t_1	t_2	t_3	t_4	t_5	t_1	t_2	t_3	t_4	t_5
u_1	2	4	5	1	3	1.99	3.84	4.9	1.21	2.98
u_2	3	2	3	4	5	2.91	2.12	3.14	3.81	4.82
u_3	1	3	5	2	1	1.40	3.16	4.84	1.92	1.8
AVG	2	3	4.3	2.3	3	2.1	3.04	4.29	2.31	3.2

Predictions are determined on the basis of Formula 9.5

between the rating of u_a on those of the other group members. This difference is weighted by (1) the level of trust between u_a and other group members ($t(u, u_a)$) and (2) the *cmw* factor representing a user's preparedness to adapt his/her rating [31].

$$p_{pers}(u_a, i) = p(u_a, i) + (1 - cmw(u_a)) \times \frac{\Sigma_{u \in G(u \neq u_a)} t(u, u_a) \times (p(u, i) - p(u_a, i))}{|G| - 1} \tag{9.5}$$

Applying Formulae 9.4 and 9.5 results in the rating predictions in Table 9.7.

As mentioned, different approaches exist to integrate the aspects of personality, emotion, and group dynamics into the determination of recommenders. In order to sketch how these aspects can be integrated on the algorithmic level, we demonstrated one possible approach [31] on the basis of a working example. Related open issues for future research will be discussed in the following.

9.4 Conclusions and Research Issues

Existing group recommendation techniques usually assume preference independence (the preferences of one group member do not have an impact on the preferences of the other group members) and thus do not take into account social interactions and relationships among the group members. It is assumed that $rating(user, item) = rating(user, item, group)$ which is not the case, i.e., group members are influenced in their evaluations by the composition of the group and the interaction between and social relationships among group members [12, 14, 22, 29]. Groups can significantly differ in terms of, for example, the *number of group members*, the *roles of persons within a group*, the *social dynamics within a group*, the *underlying goal of the group decision process*, the *status of group members*, the *age of the group members*, the *history of past group decisions and the related sentiments of group members*, and the *implicit decision policies defined within the group* [34]. These examples and many more have to be analyzed in more detail to better understand how to best support group decision making on the basis of recommendation technologies. A first approach to take into account the social dynamics of groups in the context of group recommendation is presented in [5],

where social networks are analyzed with regard to aspects such as *relationships between group members*, *social similarity*, and *social centrality*. Related contributions are also provided by Masthoff [21] who shows how to take into account the concept of emotional contagion, and Quijano-Sanchez et al. [29] who also show how to integrate *personality*-related information into group recommendation approaches. The role of group dynamics and decision making in recommender systems has also been analyzed in Delic et al. [6], where a user study is presented that focuses on measuring and observing the evolution of user preferences in travel decision making scenarios—more precisely, selecting a destination to visit. To some extent, not every group member is equally susceptible to emotional contagion and certain differences exist that depend on the personality of group members. A more in-depth investigation on how to best combine personality information with the concepts of emotional contagion is an important issue for future research. For example, a group recommender system could tailor recommendations more to those group members with a higher ability to transfer emotions to others.

References

1. M. Asgari, G. Kiss, J. van Santen, I. Shafran, X. Song, Automatic measurement of affective valence and arousal in speech, in *IEEE International Conference on Acoustic, Speech and Signal Processing (ICASSP)* (IEEE, New York, 2014), pp. 965–969
2. S. Barsade, The ripple effect: emotional contagion and its influence on group behavior. Adm. Sci. Q. **47**(4), 644–675 (2002)
3. S. Berkovsky, J. Freyne, Group-based recipe recommendations: analysis of data aggregation strategies, in *4th ACM Conference on Recommender Systems*, Barcelona, 2010, pp. 111–118
4. R. Brown, *Group Processes* (Blackwell Publishing, Malden, 2012)
5. I. Christensen, S. Schiaffino, Social influence in group recommender systems. Online Inf. Rev. **38**(4), 524–542 (2014)
6. A. Delic, J. Neidhardt, T. Nguyen, R. Ricci, Research methods for group recommender systems, in *RecTour 2016*, Boston, MA, 2016
7. F. D'Errico, I. Poggi, *Social Emotions. A Challenge for Sentiment Analysis and User Models*. HumanComputer Interaction Series (Springer, Berlin, 2016), pp. 13–34
8. A. Felfernig, M. Jeran, G. Ninaus, F. Reinfrank, S. Reiterer, M. Stettinger, Basic approaches in recommendation systems, in *Recommendation Systems in Software Engineering* (Springer, New York, 2013), pp. 15–37
9. I. Fernández-Tobías, M. Braunhofer, M. Elahi, F. Ricci, I. Cantador, Alleviating the new user problem in collaborative filtering by exploiting personality information. User Model. User-Adap. Inter. **26**(2–3), 221–255 (2016)
10. A. Finnerty, B. Lepri, F. Pianesi, Acquisition of personality, in *Emotions and Personality in Personalized Services: Models, Evaluation and Applications*, vol. 1181, ed. by M. Tkalčič, B. De Carolis, M. de Gemmis, A. Odić, A. Košir (Springer, Berlin, 2016), pp. 81–99
11. D. Forsyth, *Group Dynamics* (Thomson Higher Education, Belmont, 2006)
12. M. Gartrell, X. Xing, Q. Lv, A. Beach, R. Han, S. Mishra, K. Seada, Enhancing group recommendation by incorporating social relationship interactions, in *International Conference on Supporting Group Work (ACM GROUP)*, Sanibel, FL, 2010, pp. 97–106
13. J. Golbeck, C. Robles, K. Turner, Predicting personality with social media, in *Annual Conference on Human Factors in Computing Systems - CHI EA '11* (2011), pp. 253–262

14. J. Gorla, N. Lathia, S. Robertson, J. Wang, Probabilistic group recommendation via information matching, in *22nd WWW Conference*, Rio de Janeiro, 2013, pp. 495–504
15. E. Hatfield, J. Cacioppo, R. Rapson, Emotional contagion. Curr. Dir. Psychol. Sci. **2**(3), 96–99 (1993)
16. A. Ho, I. Menezes, Y. Tagmouti, E-MRS: emotion-based movie recommender system, in *IADIS e-Commerce Conference* (2006), pp. 1–8
17. O. John, S. Srivastava, The big five trait taxonomy: history, measurement, and theoretical perspectives, in *Handbook of Personality: Theory and Research*, ed. by L. Pervin, O. John, vol. 2 (Guilford Press, New York, 1999), pp. 102–138
18. R. Kilmann, K. Thomas, Developing a forced-choice measure of conflict-handling behavior: the 'MODE' instrument. Educ. Psychol. Meas. **37**(2), 309–325 (1977)
19. M. Kosinski, D. Stillwell, T. Graepel, Private traits and attributes are predictable from digital records of human behavior. Proc. Natl. Acad. Sci. U. S. A. **110**(15), 5802–5805 (2013)
20. A. Kramer, J. Guillory, J. Hancock, Experimental evidence of massive-scale emotional contagion through social networks. Proc. Natl. Acad. Sci. U. S. A. **111**(24), 8788–8790 (2014)
21. J. Masthoff, Group modeling: selecting a sequence of television items to suit a group of viewers. User Model. User-Adap. Inter. **14**(1), 37–85 (2004)
22. J. Masthoff, A. Gatt, In pursuit of satisfaction and the prevention of embarrassment: affective state in group recommender systems. User Model. User-Adap. Inter. **16**(3–4), 281–319 (2006)
23. S. Matz, Y. Chan, M. Kosinski, Models of personality, in *Emotions and Personality in Personalized Services: Models, Evaluation and Applications*, ed. by M. Tkalčič, B. De Carolis, M. de Gemmis, A. Odić, A. Košir (Springer, Berlin, 2016), pp. 35–54
24. R. McCrae, O. John, An introduction to the five-factor model and its applications. J. Pers. **60**(2), 175–215 (1992)
25. A. Mehrabian, Pleasure-arousal-dominance: a general framework for describing and measuring individual differences in temperament. Curr. Psychol. **14**(4), 261–292 (1996)
26. J. Neidhardt, L. Seyfang, R. Schuster, H. Werthner, A picture-based approach to recommender systems. Inf. Technol. Tour. **15**(1), 49–69 (2015)
27. T. Nguyen, F. Ricci, Combining long-term and discussion-generated preferences in group recommendations, in *25th ACM Conference on User Modeling Adaptation and Personalization*, Bratislava (ACM, New York, 2017), pp. 377–378
28. D. Quercia, M. Kosinski, D. Stillwell, J. Crowcroft, Our twitter profiles, our selves: predicting personality with twitter, in *Proceedings - 2011 IEEE International Conference on Privacy, Security, Risk and Trust and IEEE International Conference on Social Computing, PASSAT/SocialCom 2011* (IEEE, New York, 2011), pp. 180–185
29. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, G. Jiménez-Díaz, Social factors in group recommender systems. ACM Trans. Intell. Syst. Technol. **4**(1), 8:1–8:30 (2006)
30. L. Quijano-Sanchez, J. Recio-García, B. Díaz-Agudo, Personality and social trust in group recommendations, in *22nd International Conference on Tools with Artificial Intelligence*, Arras, 2010, pp. 121–126
31. L. Quijano-Sánchez, D. Bridge, B. Díaz-Agudo, J. Recio-García, A case-based solution to the cold-start problem in group recommenders, in *23rd International Conference on Artificial Intelligence (IJCAI 2013)* (2013), pp. 3042–3046
32. E. Quintarelli, E. Rabosio, L. Tanca, Recommending new items to ephemeral groups using contextual user influence, in *RecSys* (2016), pp. 285–292
33. J. Recio-Garcia, G. Jimenez-Diaz, A. Sanchez-Ruiz, B. Diaz-Agudo, Personality aware recommendations to groups, in *ACM Conference on Recommender Systems (RecSys'09)*, New York, 2009, pp. 325–328
34. B. Saha, L. Getoor, Group proximity measure for recommending groups in online social networks, in *SNA-KDD Workshop 2008*, Las Vegas, 2008, pp. 1–9
35. B. Schuller, Acquisition of affect, in *Emotions and Personality in Personalized Services: Models, Evaluation and Applications*, ed. by M. Tkalčič, B. De Carolis, M. de Gemmis, A. Odić, A. Košir (Springer International Publishing, Cham, 2016), pp. 57–80

36. M. Skowron, B. Ferwerda, M. Tkalčič, M. Schedl, Fusing social media cues: personality prediction from twitter and instagram, in *25th International Conference Companion on World Wide Web* (2016), pp. 107–108
37. M. Tkalčič, L. Chen, Personality and recommender systems, in *Recommender Systems Handbook* (Springer, New York, 2015), pp. 715–739
38. M. Tkalčič, U. Burnik, A. Košir, Using affective parameters in a content-based recommender system for images. User Model. User-Adap. Inter. **20**(4), 279–311 (2010)
39. Y. Zheng, R. Burke, B. Mobasher, The role of emotions in context-aware recommendation, in *ACM Conference on Recommender Systems (RecSys'13)*, Hong Kong, 2013, pp. 21–28

Chapter 10
Conclusions

Abstract In this chapter, we shortly summarize the contributions provided in this book.

The major focus of this book is to provide an integrated view of different aspects of group recommender systems. Since early group recommender systems were proposed in the 1990s, many different approaches and systems have been developed that altogether comprise a huge collection of group recommendation-related knowledge. With this book we want to provide a basic summary of the existing state of the art, in order to support persons new to the field as well as recommender systems researchers and practitioners. Following existing research approaches, we introduced two major preference aggregation strategies used for determining group recommendations which are (1) the aggregation of individual preferences into a group profile and (2) the aggregation of item recommendations (or item ratings) determined by single-user recommender algorithms. We showed how these aggregation approaches can be used in the context of collaborative filtering, content-based filtering, constraint-based (including utility-based), critiquing-based, and hybrid recommendation. Thereafter, we provided an overview of techniques useful for evaluating group recommender systems. In order to give insights into the user interfaces of existing systems, we provided an overview of example applications and discussed issues related to preference handling and explanations. Finally, we introduced different examples of group recommendation scenarios that go beyond the recommendation of single items. Examples include group-based configuration, packaging, sequencing, and release planning. Furthermore, we discussed decision biases that can have an impact on the quality of a group decision process. The book is concluded with an overview of concepts that help to make group recommender systems more group-aware. In this context, examples of how to take into account personality and emotion in group recommendation scenarios have been provided. Since group recommender systems can be regarded

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5_10

as relatively young field of research, there are many open research issues that have been discussed in the individual book chapters. Examples thereof are explanation approaches (especially focused on the explanation of recommendations for groups), dataset synthesis and evaluation approaches for group recommenders, counteracting biases in group decision making, and extending recommender systems research to more complex choice scenarios.

Index

A. Felfernig et al., *Group Recommender Systems*, SpringerBriefs in Electrical and Computer Engineering, https://doi.org/10.1007/978-3-319-75067-5