UNIVERSIDADE FEDERAL DE MINAS GERAIS
Instituto de Ciências Exatas
Programa de Pós-Graduação em Ciência da Computação

Reinaldo Silva Fortes

Enhancing the Multi-Objective Recommendation from three new perspectives: data characterization, risk-sensitiveness, and prioritization of the objectives

Belo Horizonte
2022

Reinaldo Silva Fortes

Enhancing the Multi-Objective Recommendation from three new perspectives: data characterization, risk-sensitiveness, and prioritization of the objectives

Versão Final

Tese apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal de Minas Gerais, como requisito parcial à obtenção do título de Doutor em Ciência da Computação.

Orientador: Marcos André Gonçalves
Coorientador: Anisio Mendes Lacerda

Belo Horizonte
2022

Reinaldo Silva Fortes

Enhancing the Multi-Objective Recommendation from three new perspectives: data characterization, risk-sensitiveness, and prioritization of the objectives

Final Version

Thesis presented to the Graduate Program in Computer Science of the Federal University of Minas Gerais in partial fulfillment of the requirements for the degree of Doctor in Computer Science.

Advisor: Marcos André Gonçalves
Co-Advisor: Anisio Mendes Lacerda

Belo Horizonte
2022

Fortes, Reinaldo Silva.

F738e Enhancing the Multi-Objective recommendation from three new perspectives: data characterization, risk-sensitiveness, and prioritization of the objectives [manuscrito] / Reinaldo Silva Fortes. – 2022.
125 f. il.

Orientador: Marcos André Gonçalves
Coorientador: Anisio Mendes Lacerda.
Tese (Doutorado) - Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Ciências da Computação.
Referências: f. 97-105.
.
1. Computação – Teses. 2. Sistemas de recomendação – Teses. 3. Otimização multi-objetivo – Teses. . I. Gonçalves, Marcos André II. Lacerda, Anisio Mendes. III. Universidade Federal de Minas Gerais, Instituto de Ciências Exatas, Departamento de Computação. IV.Título.

CDU 519.6*73(043)

Ficha Ficha catalográfica elaborada pela bibliotecária Belkiz Inez Rezende Costa CRB 6/1510 Universidade Federal de Minas Gerais - ICEx

UNIVERSIDADE FEDERAL DE MINAS GERAIS
INSTITUTO DE CIÊNCIAS EXATAS
DEPARTAMENTO DE CIÊNCIA DA COMPUTAÇÃO
PROGRAMA DE PÓS-GRADUAÇÃO EM CIÊNCIA DA COMPUTAÇÃO

FOLHA DE APROVAÇÃO

ENHANCING THE MULTI-OBJECTIVE RECOMMENDATION FROM THREE NEW PERSPECTIVES: DATA CHARACTERIZATION, RISK-SENSITIVENESS, AND PRIORITIZATION OF THE OBJECTIVES

REINALDO SILVA FORTES

Tese defendida e aprovada pela banca examinadora constituída pelos Senhores(a):

Prof. Marcos André Gonçalves - Orientador
Departamento de Ciência da Computação - UFMG

Prof. Anisio Mendes Lacerda - Coorientador
Departamento de Ciência da Computação - UFMG

Prof. Leandro Balby Marinho
Departamento de Sistemas e Computação - UFCG

Prof. Leonardo Chaves Dutra da Rocha
Departamento de Ciência da Computação - UFSJ

Prof. Ricardo Bastos Cavalcante Prudêncio
Centro de Informática - UFPE

Prof. Rodrygo Luis Teodoro Santos
Departamento de Ciência da Computação - UFMG

Belo Horizonte, 27 de maio de 2022.

Documento assinado eletronicamente por **Marcos Andre Goncalves**, **Membro de comissão**, em 10/06/2022, às 11:15, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Anisio Mendes Lacerda**, **Professor do Magistério Superior**, em 14/06/2022, às 09:13, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Rodrygo Luis Teodoro Santos**, **Professor do Magistério Superior**, em 14/06/2022, às 16:03, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Leandro Balby Marinho**, **Usuário Externo**, em 21/06/2022, às 15:29, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Leonardo Chaves Dutra da Rocha**, **Usuário Externo**, em 21/06/2022, às 17:38, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

Documento assinado eletronicamente por **Ricardo Bastos Cavalcante Prudêncio**, **Usuário Externo**, em 29/06/2022, às 06:36, conforme horário oficial de Brasília, com fundamento no art. 5º do Decreto nº 10.543, de 13 de novembro de 2020.

A autenticidade deste documento pode ser conferida no site https://sei.ufmg.br/sei/controlador_externo.php?acao=documento_conferir&id_orgao_acesso_externo=0, informando o código verificador **1512729** e o código CRC **74BFEDD1**.

This work is dedicated to my family, Silviane, Davi, and Luiza (wife and children); Neli and Bartolomeu (parents); Bianca, Bartira, and Marquinhos (sisters and brother), who greatly contributed to the conclusion of this stage of our life with their total understanding, affection, and support.

Acknowledgments

To the Federal University of Minas Gerais (UFMG) and the Computer Science Department (DCC), who through their magnificent faculty, directors, administrators, and many other employees, provided me with this opportunity and all the support needed throughout this journey. To the Federal University of Ouro Preto (UFOP) and the Computer Department (DECOM), which also provided me with this opportunity and all the support needed to complete this work. To my supervisor Marcos Gonçalves and co-supervisor Anisio Lacerda, for all the support, corrections, incentives, and guidance. To the collaborators' Alan Freitas, Carlos Bruckner, Daniel Sousa, Dayanne Coelho, Pedro Rodrigues, and Thierson Rosa, for their relevant contribution to obtaining the scientific results derived from this work. To the friendships built during the classes, studies, tests, and practical works, which provided us with good moments and much learning. To CAPES, CNPq, and FAPEMIG by the financial support for scientific publications. To the coordinators of the projects InWeb, MASWeb, and TerraLab, who provided important resources for the accomplishment of this work. And to everyone who somehow contributed to this work.

Resumo

Sistemas de Recomendação são ferramentas cujo principal objetivo é auxiliar os usuários a encontrar itens relevantes em meio a muitas opções. Entretanto, diferentes conceitos de "relevância" podem ser definidos, tornando a tarefa de recomendação ainda mais desafiadora se desejarmos boas recomendações sobre múltiplos conceitos de qualidade, *e.g.*, *acurácia*, *novidade* e *diversidade*. Neste cenário, a recomendação precisa utilizar mecanismos de otimização multi-objetivo. Apesar de encontrarmos trabalhos voltados para a este tipo de recomendação, a maioria deles possui limitações sobre alguns aspectos relevantes. Três aspectos, em especial, abrem margem para aprimorar a recomendação multi-objetivo sobre novas perspectivas com o uso de recursos adicionais: (a) ***meta-features***: características implícitas dos dados de entrada podem influenciar os algoritmos, *e.g.*, quantidade e distribuição dos *ratings* dos itens, portanto, o uso explícito de medidas estatísticas capazes de mensurar algumas destas características pode ser útil no processo de recomendação multi-objetivo; (b) **sensibilidade ao risco**: a otimização de múltiplos critérios pelas suas médias globais pode gerar resultados ruins em favorecimento de alguns resultados muito bons que, embora raros, sejam capazes de afetar positivamente as médias, portanto, a utilização explícita de métricas de sensibilidade ao risco pode ser útil no processo de otimização, reduzindo as recomendações ruins sem degradar as médias globais; (c) **priorização dos objetivos**: usuários possuem diferentes preferências em relação aos critérios de qualidade das recomendações, *e.g.*, enquanto alguns usuários não abrem mão de itens prediletos, outros podem ser mais tolerantes à descoberta de novos itens ou maior diversidade de itens, portanto, a utilização explícita de preferências dos usuários sobre os critérios de qualidade também pode ser útil para melhorar ainda mais as recomendações multi-objetivo. Sendo assim, neste trabalho investigamos a recomendação multi-objetivo sobre a ótica destas três novas perspectivas e definimos métodos de recomendação específicos. Extensos experimentos validaram esses métodos e respondem positivamente às nossas questões de pesquisa, e também nos permitiram começar a entender melhor a recomendação multiobjetivo sobre esses três aspectos, abrindo margem para relevantes trabalhos futuros.

Palavras-chave: Computação, Sistemas de Recomendação, Filtragem Híbrida, Filtragem Multiobjetivo

Abstract

Recommender Systems are tools whose main objective is to help users find relevant items among many options. However, different "relevance" concepts can be defined, making the recommendation task even more challenging if we want good recommendations on multiple quality concepts, *e.g.*, *accuracy*, *novelty*, and *diversity*. In this scenario, the recommendation needs to use multi-objective optimization mechanisms. Although we find works focused on this type of recommendation, most of them are limited in some relevant aspects. In particular, three aspects provide scope for improving the multi-objective recommendation on new perspectives with the use of additional resources: (a) ***meta-features***: implicit characteristics of input data can influence algorithms, *e.g.*, quantity and distribution of items' ratings, therefore, explicit use of statistical measures capable of measuring some of those characteristics can be helpful in the multi-objective recommendation; (b) **risk sensitivity**: the optimization by global averages of multiple criteria can generate bad results in exchange for some excellent results that, although rare, can positively affect these averages, therefore, explicit use of risk sensitivity metrics can be helpful in the optimization process, reducing harmful recommendations without degrading global averages; (c) **prioritization of objectives**: users have different preferences regarding the quality criteria of recommendations, *e.g.*, while some users do not give up favorite items, others may be more tolerant of discovering new items or a greater diversification of items, therefore, explicit use of users' preferences regarding the quality criteria can also be helpful to improve multi-objective recommendations further. Accordingly, in this work, we investigated the multi-objective recommendation from these three new perspectives and defined specific recommendation methods. Extensive experiments validated these methods, answered our research questions positively, and improved our knowledge concerning multi-objective recommendations on these three aspects, opening opportunities for relevant future work.

Keywords: Computer, Recommender Systems, Hybrid Filtering, Multi-Objective Filtering

List of Figures

List of Tables

Glossary

CB Content-Based Filtering. 16, 26, 37, 42, 43, 47, 92, 95, 106

CF Collaborative Filtering. 16, 26–29, 34–37, 42–44, 55, 92, 95, 108

DM Decision Making. 30, 47, 49, 51–53, 56, 65, 66, 68, 69, 73, 83

EILD Expected Intra-List Distance. 23, 24, 47, 49, 50, 52, 53, 65, 68, 69, 71, 83, 87, 96

EPD Expected Profile Distance. 23, 24, 47, 49, 52, 53, 65, 68, 71, 73, 83, 96

FWLS Feature-Weighted Linear Stacking. 11, 38–40, 44, 47, 49–51, 54, 56, 65, 66, 83, 85, 111

HR Hybrid Recommender. 44, 47, 49–51, 54, 56, 65–68, 70–73, 83, 85, 87, 88, 111

MO Multi-Objective. 10, 17–20, 29, 30, 34, 40, 42, 45, 47–56, 60, 63, 65–73, 75, 76, 78–83, 86–93, 96

MOF Multi-Objective Filtering. 10, 12, 17–22, 26, 31, 34, 40, 41, 45, 58, 60, 61, 64, 65, 75, 77, 78, 82, 92–94, 111, 112

NDCG Normalized Discounted Cumulative Gain. 23, 24, 47, 49, 50, 52, 59, 65–73, 83, 85, 96

NSGA-II Nondominated Sorting Genetic Algorithm II. 47, 48, 58, 63, 78, 81, 96, 112

PEH Pareto-Efficient Hybridization. 11, 31, 32, 34, 40, 41, 47, 56

RMSE Root Mean Squared Error. 22, 111

RS Recommender System. 16–19, 21, 23, 24, 26, 29, 34, 58–62, 76–78, 92, 94

SO Single-Objective. 48–53, 56, 65–68, 70, 72, 77, 78, 83, 86, 87

STREAM STacking Recommendation Engines with Additional Meta-features. 10, 11, 37–40, 44, 47, 49–51, 54, 56, 65, 66, 73, 83, 88, 111

WHF Weighted Hybrid Filtering. 11, 12, 16, 17, 20, 26, 29, 31, 34, 35, 37, 39, 40, 44, 47–53, 55, 56, 61, 64–67, 70–72, 78, 83, 85–88, 111, 112

*

Contents

Chapter 1

Introduction

Recommender Systems (**RSs**) [Aggarwal, 2016; Jannach et al., 2010; Ricci et al., 2011] has proved to be an essential asset for modern software applications due to information overload. Information overload is a problem faced by users of several computer applications due to a large amount of content available to them, which is very difficult to process without the system's assistance. In this context, it is challenging to find content relevant to the user's desires and needs, especially when they are not explicitly searching for something specific. Thus, RSs have emerged as a relevant personalization tool by helping users find what they really want or need, or even what may be of interest but is still unknown or hard to find.

The main RS approaches proposed in the literature include: (a) **Content-Based Filtering**; (b) **Collaborative Filtering**; and (c) **Hybrid Filtering**. Content-Based Filtering exploits attributes from items to recommend the most similar ones to user profiles. Collaborative Filtering exploits the idea that users who previously expressed similar tastes tend to continue expressing specific tastes in the future. The core principle of Hybrid Filtering is to combine different algorithms exploiting their strengths while avoiding their weaknesses [Burke, 2002, 2007].

One prominent hybridization method is the **Weighted Hybrid Filtering** (**WHF**) due to its potential to generate good results, despite its simplicity. This method computes a hybrid score by the linear combination of input features represented by the scores of the algorithms being combined. In particular, Bao et al. [2009] and Sill et al. [2009] have developed enhanced WHF methods based on Stacked Generalization and using additional features able to capture characteristics of the input data that can improve the combination of the algorithms. These additional features are called *meta-features* in this work and consist of statistical measures taken from the input data for the RS. Some simple examples may be listed: the amount of data (*e.g.*, number of ratings) and relationships among items (*e.g.*, number of items with high similarity with a particular item). However, these works only attempt to optimize the **accuracy** criterion, ignoring other important recommendation quality aspects, such as **novelty** and **diversity**, although several authors emphasize the importance of other quality aspects [Adomavicius and Tuzhilin, 2005; Gunawardana and Shani, 2009; Herlocker et al., 2004; McNee et al., 2006; Vargas and Castells, 2011].

To consider more than one quality aspect of the recommendations, authors typically model the recommendation task as a **Multi-Objective** (**MO**) optimization problem [Geng et al., 2015; Rodriguez et al., 2012; Wang et al., 2016; Zuo et al., 2015]. Therefore, another relevant recommendation approach is the **Multi-Objective Filtering** (**MOF**). However, due to conflicting factors between the multiple quality aspects and specificities of RSs, finding a model that can successfully produce useful recommendations becomes even more challenging. Therefore, addressing specific issues not yet explored in the MOF context becomes relevant, as discussed below.

Many MOF systems use re-ranking techniques to optimize multiple objectives. They first generate a list of the most relevant items (*accuracy*) and then reorder these lists to meet other quality criteria (*e.g.*, *novelty* and *diversity*). These strategies naturally tend towards accuracy. However, we claim that certain users may not want this trend, and systems must satisfy all users in any scenario, optimizing all objectives simultaneously. In particular, Ribeiro et al. [2012, 2014] proposed a strategy that aggregates the advantages of the two approaches, WHF and MOF, by combining results from various algorithms and optimizing all objectives simultaneously.

However, their works do not explicitly explore ***meta-features*** to improve the hybridization strategy, although *meta-features* help to improve results in single-objective hybrid recommendations, as observed in [Bao et al., 2009; Sill et al., 2009]. Nevertheless, no previous work explicitly uses this additional resource in multi-objective and hybrid recommendations. In Fortes et al. [2017], we produced an extensive multi-criteria analysis of *meta-featured* WHF, observing that, although these systems only optimize accuracy, *meta-features* were also valuable to improve results in other varied criteria. This issue motivates the **first research question** we aim to investigate: ***RQ1: "Does explicitly incorporating meta-features contribute to improving the results of multi-objective recommendation?"***. We investigate this question and aim to expand knowledge in this area by presenting a multi-objective recommendation strategy using an explicit set of *meta-features* in Chapter 3. Experimental results allow us to positively answer ***RQ1***, showing that in some scenarios, the explicit use of *meta-features* produces better recommendation results when considering multiple optimization criteria and provides better results for the MO search.

Even if the *meta-features* contribute to obtain better MO recommendations, another critical issue facing RSs is that the quality of recommendations may vary a lot among users in response to different recommendation requests. This issue can lead to a phenomenon where a given user receives good suggestions from the RS most of the time but sometimes receives such disappointing recommendations that they become suspicious about the actual effectiveness of the system. Knijnenburg et al. [2012] showed that users tend to remember the few failures of a Recommender System more quickly than the many successful results they have received from it. Hybridization can alleviate this issue by combining results from different algorithms, thus reducing variability. On the other hand, it can be worsened by conflicts between the multiple objectives to be optimized.

Therefore, we claim that, although hybridization and MO optimization have natural capabilities to reduce the variation of recommendation results, a method that uses other additional features directly related to this issue can promote even better results besides *meta-features*. This behavior has been known in the Information Retrieval literature related to Search Engines, which has paid attention to models that have the **risk** (*i.e.*, *non-negligible likelihood*) of producing poor effectiveness in some specific queries [Dinçer et al., 2016]. The main goal of the **risk-sensitive** task is to improve the overall effectiveness while minimizing poorer predictions when compared to baseline systems that do not take the risk into account [Dinçer et al., 2016; Sousa et al., 2016]. **Risk-sensitive measures** have been developed for search engines to maximize some overall measure of user satisfaction while avoiding the risk of incurring bad results for a few but essential queries [Dinçer et al., 2014; Wang et al., 2012].

Risk-sensitiveness has not yet been explicitly tackled in the context of RSs. We claim that recommender models can better homogenize users' satisfaction by considering sensitiveness to risk. Moreover, risk-sensitiveness has been historically considered only for accuracy-related measures – mainly in the search realm, where the concept has been coined. However, the risk concerning other aspects, such as *novelty* and *diversity*, even if not equally important, should also be considered for promoting user satisfaction. Thus, these issues motivate the **second research question** we aim to investigate: ***RQ2: "Does explicitly incorporating risk-sensitive measures contribute to improving the results of multi-objective recommendation?"***. We investigate this question and aim to expand knowledge in this area by presenting a multi-objective recommendation strategy explicitly using risk-sensitive measures in Chapter 4. Experimental results allow us to positively answer ***RQ2***, showing that in many scenarios, the explicit use of risk-sensitive measures can reduce loss and degradation of specific recommendations while still being able to maintain good overall recommendation results.

Even if *meta-features* are useful and it is possible to reduce risk sensitivity in MO recommendations, another essential issue involves the users' preferences concerning the optimized objectives. Ribeiro et al. [2012, 2014] argued that users might have different expectations regarding the relevance of each objective function. Thus, after searching for a set of Pareto solutions, they exploit a limited set of *ad hoc* weights for the objective functions to select the most promising solution to achieve the users' preferences concerning the optimized objectives.

However, we claim that users' preferences concerning objectives are essential in modern MOF systems and deserve more attention to enhance their personalization capability even further. The personal concept of a "good" recommendation may vary widely from one user to another. Some users may have a more exploratory profile, preferring recommendations for newer and more diverse items over their favorite ones. Others may have an opposite or intermediate profile. Even if the number of users of a given profile type is small, RSs must be prepared to satisfy their expectations.

MO methods that explore users' preferences concerning the objective functions have been named ***Preference-based*** methods [Bechikh et al., 2015; Fonseca and Fleming, 1993; Wang et al., 2019]. Typically, these methods are applied when there exist four or more objectives to be optimized, *i.e.*, *Many-Objective* problems [Ishibuchi et al., 2008; Li et al., 2015]. In this context, the utilization of users' preferences has the main objective of reducing the complexity in the search for optimal solutions by optimizing a small number of objective functions. However, the users' preferences in previous methods refer to the importance of the objective functions for **decision-makers**, usually represented by the business managers, who decide which criteria to prioritize (usually from a business perspective). In the RSs context, we claim that the focus should be mainly on the **end-users** who will receive recommendations.

In Fortes et al. [2018], we started to deal with this issue, presenting an MO search strategy based on users' preferences, pointing to promising preliminary results. This issue motivates the **third research question** we aim to investigate: ***RQ3: "Does explicitly incorporating individual preferences of users concerning the optimized objectives contribute to improving the results of multi-objective recommendation?"***. We investigate this question and aim to expand knowledge in this area by presenting a multi-objective recommendation strategy explicitly using users' preferences regarding the objectives in [Fortes et al., 2021]. Experimental results allow us to positively answer ***RQ3***, showing that in some scenarios, the explicit use of users' preferences can bring results closer to their specific expectations without degrading the overall results of the system. In Chapter 5, we have expanded this work by broadening the scope of the methods and experiments reinforcing the positive answer to ***RQ3***.

To summarize, the search for improvement in recommendation results is permanent. Several algorithms, methods, and approaches are defined or improved continuously. Indeed, Recommender Systems need to meet multiple conflicting objectives, involving different evaluation criteria, supporting the development of MOF systems. Many research challenges involving MO recommendations are identified; in this sense, exploring the new perspectives that we bring in this thesis, thought the incorporation of additional resources, can be necessary for advancing knowledge in the area.

The remainder of this chapter is organized as follows. The objectives we aim to achieve are formalized in Section 1.1. In Section 1.2, the thesis outline is presented to guide the reader through this document.

1.1 Objectives

The main objective of this work is to advance the knowledge and strategies of Weighted Hybrid Filtering (WHF) when considering Multi-Objective recommendations. As specific objectives, we intend to address the WHF problem from three new perspectives not yet explored in the scope of Multi-Objective Filtering (MOF) by incorporating new resources explicitly:

- ***Meta-featured***: explicitly exploiting data characterization measures to improve the MOF recommendations;
- **Risk-sensitiveness**: improving the results of MOF recommendations by explicitly exploiting risk-sensitive measures;
- **Preference-based**: explicitly exploiting users' preferences regarding the importance of each optimized objective function in MOF systems to meet their expectations better.

1.2 Thesis Outline

This chapter presents an introduction containing an overview of the proposed work and objectives. Following, we detail how we organize the thesis:

- Chapter 2, "Fundamental concepts", presents an overview of the main concepts related to this thesis.
- Chapter 3, "Meta-featured MOF", presents a strategy to explore *meta-features* in MOF recommendation, evaluating its effectiveness and answering our first research question.
- Chapter 4, "Risk-sensitive MOF", discusses the risk of harmful recommendations and presents a new risk-sensitive MOF method, evaluating its effectiveness and answering our second research question.
- Chapter 5, "Preference-based MOF", discusses users' expectations about the objective functions and presents a new preference-based MOF method, evaluating its effectiveness and answering our third research question.
- Finally, Chapter 6, "Conclusions and Future Work", concludes this thesis by presenting final considerations and suggestions for future works.

Chapter 2

Fundamental concepts

In this chapter, we present the main concepts related to this thesis that are important for the understanding of later chapters. Section 2.1 presents a formal definition of the recommendation problem. Section 2.2 discusses some issues regarding the evaluation of RSs, including risk-sensitive measures. Section 2.3 presents the main RSs approaches, including the particular case of Multi-Objective Filtering. Finally, Section 2.4 concludes the chapter with some considerations.

2.1 The Recommendation Problem

The primary purpose of **Recommender Systems** (**RSs**) [Aggarwal, 2016; Jannach et al., 2010; Ricci et al., 2011] is to help users find items that they would probably appreciate from a massive set of options. Otherwise, the users would have never noticed these items. It has the potential to increase users' satisfaction and also the revenue for content providers. However, many challenging problems involve suggesting personalized items to users and making complex recommendations. For instance, applications focus on specific domains composed of different items (*e.g.,* movies, news, and services), having their particular graphical user interfaces and their business models. Moreover, users have specific characteristics and have their own distinct needs and interests. Next, we formalize the recommendation problem.

Consider the dataset $\mathcal{D} = (\mathcal{U}, \mathcal{I}, \mathcal{R})$, composed of a variety of users ($\mathcal{U}$), items ($\mathcal{I}$), and user-item's interactions ($\mathcal{R}$). A set of k user's attributes A^u represents each user. Some examples include *age*, *gender*, *profession*, and *hobbies*. Thus, $\mathcal{U} = \{u_1, u_2, ..., u_m\}$ defines a set of users, where $u_x = (A^u_1, A^u_2, ..., A^u_k)$ represents each user and $1 \leq x \leq m$.

Similarly, a set of l item's attributes A^i represents each item. The items' attributes are dependent on the application domain. Examples in a movie recommendation domain, include *title*, *actors*, *synopsis*, and *genre*. For general product recommendations, some examples are *price*, *description*, *category*, and *manufacturer*. Thus, $\mathcal{I} = \{i_1, i_2, ..., i_n\}$

defines a set of items, where $i_y = (A^i_1, A^i_2, ..., A^i_l)$ represents each item and $1 \leq y \leq n$.

Let $r : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ be a function mapping user-item pairs to rating values representing the users' preferences on items. Thus, $\mathcal{R} = \{(u, i, r(u, i))\}$ defines a set of interactions for all users u who have shown interest on item i through rating $r(u, i)$, where $u \in \mathcal{U}$, $i \in \mathcal{I}$, and $r(u, i) \in \mathbb{R}$. Since users tend to rank or interact with a limited set of items, a sparse matrix usually represents the preferences $\mathcal{R}$.

Therefore, we formulate the recommendation problem as two primary tasks [Aggarwal, 2016]: (a) ***Rating prediction***: it deals with estimating the rating value a user would give to an item by learning a prediction function $\hat{p} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ to compose a set of predicted ratings $\mathcal{P} = \{(u, i, \hat{p}(u, i)) \mid u \in \mathcal{U} \wedge i \in \mathcal{I} \wedge \hat{p}(u, i) \in \mathbb{R}\}$, which minimizes a prediction error measure, $err(\mathcal{P})$. This measure is capable of expressing the quality of the learned model $\hat{p}$ regarding the users' rating on "unseen" items, such as Mean Absolute Error (MAE) and Root Mean Squared Error (RMSE); (b) ***Ranking*** or ***Top-N***: it deals with returning a list of the N items best ranked for a user by learning a prediction function $\hat{s} : \mathcal{U} \times \mathcal{I} \rightarrow \mathbb{R}$ to compose a set of relevance scores $\mathcal{S} = \{(u, i, \hat{s}(u, i)) \mid u \in \mathcal{U} \wedge i \in \mathcal{I} \wedge \hat{s}(u, i) \in \mathbb{R}\}$, which minimizes a ranking measure, $rank@N(\mathcal{S})$. This measure can express the quality of the learned model $\hat{s}$ according to the users' interests in the items contained in the ranked list, such as F-measure and Discounted Cumulative Gain (DCG).

In this thesis, we are interested in the *Top-N* task, given that it best mimics real systems [Cremonesi et al., 2010]. Additionally, it is more adherent to MOF systems, the main subject of this thesis.

2.2 Recommender Systems Evaluation

The evaluation of Recommender Systems can be performed according to two main protocols [Gunawardana and Shani, 2009]: (a) **Online**: measures the performance in real systems, typically measuring changes in the behavior of selected users according to their interactions with the recommendations from different algorithms; and (b) **Offline**: measures the performance in a pre-processed dataset, generally extracted from real systems, measuring the quality of the predictions or recommendations produced for selected users using data partitions to *train*, *test*, and *validate* the algorithms.

Unfortunately, *online* experimentation requires an operational application available and open for testing with real users. Therefore, only *offline* evaluations will be performed in this work, considering the evaluation measures we exploited in our experiments, discussed in the following subsections.

2.2.1 Ranking measures

Different aspects may be relevant when assessing the quality of the recommendations. As we are interested in the *Top-N* recommendation task, here, we describe measures that evaluate the list of recommended items according to three evaluation criteria: (a) **Accuracy**: the ability to recommend relevant items; (b) **Novelty**: the ability to recommend unknown (relevant) items; and (c) **Diversity**: the ability to recommend dissimilar (relevant) items.

Works on MO recommendation extensively explore these three criteria, which have conflicting and complementary characteristics. Promoting greater accuracy can severely sacrifice novelty and diversity criteria, and vice versa. Additionally, increasing diversity does not necessarily lead to novelty, and vice versa. Miscellaneous items may be known to the user, and items new to the user are not necessarily miscellaneous.

We present below only measures directly used in the optimization methods and experimental results presented in the following chapters. However, the defined methods are flexible and general, readily supporting other measures.

Normalized Discounted Cumulative Gain (**NDCG**) [Liu, 2011] evaluates the recommendation accuracy considering the number of relevant items recommended and their position in the list and is defined as:

$$NDCG = \frac{1}{IDCG} * \sum_{i=1}^{N} \frac{2^{rel_i} - 1}{log_2(i+1)}, \tag{2.1}$$

where: i is the position in the *Top-N* ranked list; rel_i is a boolean value indicating whether the item is relevant (1) or not (0); and $IDCG$ is a normalization factor defined by the maximum possible sum value.

Vargas and Castells [2011] present extensive work about evaluation measures in the scope of RSs, especially considering the novelty and diversity aspects. The following two measures, **Expected Profile Distance** (**EPD**) and **Expected Intra-List Distance** (**EILD**), aim to measure the novelty (EPD) or diversity (EILD) of a recommendation list. A common aspect of these measures is the concept of relevance, a notion of the user interest on items. Vargas and Castells [2011] express relevance as:

$$p(rel|i,u) \approx \frac{2^{r(u,i)}}{2^{r_{max}}}, \tag{2.2}$$

where: $r(u,i)$ is the rating that the user u gives for the item i; and r_{max} is the maximum possible rating value.

Another common aspect is the concept of items' distance or similarity between items. In this work, we use the Cosine Similarity to measure the items' distance, defined as:

$$d(i,j) = 1 - \frac{|U_i \cap U_j|}{\sqrt{|U_i|}\,\sqrt{|U_j|}}, \tag{2.3}$$

where: U_i and U_j are the set of users who liked the item i and j, respectively; and $|U|$ is the cardinality of the set U.

Finally, the last two common aspects are the ranking discount values, defined as $disc(k) = 0.85^{k-1}$, and the constant $C = 1/\sum_{i_k \in R} disc(k)$, where: k is the ranking position; R is the ordered recommended items for user u; and i_k is the k-th item in R.

Then, EPD measures the notion of novelty for the target user by assessing the distance of the recommended items to the items in the user profile and is defined as:

$$EPD = C' \sum_{i_k \in R} \sum_{i_j \in P} disc(k)\; p(rel|i_k,u)\; p(rel|i_j,u)\; d(i_k,i_j), \tag{2.4}$$

where: P is the set of items rated by the user u; i_j is the j-th item in P; $C' = C/\sum_{i_j \in P} p(seen|i_j)$; and $p(seen|i)$ is the popularity of item i, *i.e.*, the probability of it being seen, defined as the percentage of users that rated the item i.

On the other hand, EILD measures diversity, *i.e.*, the distance between items in the recommendation list, and is defined as:

$$EILD = C'' \sum_{i_k, i_l \in R \,\wedge\, l \neq k} disc(k)\; p(rel|i_k,u)\; p(rel|i_l,u)\; disc(l|k)\; d(i_k,i_l), \tag{2.5}$$

where: $disc(l|k) = disc(max(1, l-k))$; and $C'' = C/\sum_{i_l \in R - \{i_k\}} disc(l|k)$.

Note that the EPD and EILD measures privilege more relevant items at the top of the list through the term $p(rel|i,u)$, defined in Equation 2.2. However, using this term reduces the conflicting relationship of these two measures with the NDCG accuracy measure. In terms of MO optimization, since NDCG is the objective directly related to the accuracy criteria through relevance evaluation, we use EPD and EILD without the term $p(rel|i,u)$ in the following chapters. In this way, each measure exclusively evaluates one of the three criteria leaving the method to balance the trade-off between all criteria without the aforementioned bias.

2.2.2 Risk-sensitive measures

Ranking measures, such as the three measures described above, are traditionally used to evaluate RSs taking into account the ability of the system to satisfy users positively. However, on the other side, some works in RSs literature have evaluated the risk of "dissatisfying users". Knijnenburg et al. [2012] showed the importance of avoiding high variation in recommendation algorithms. Indeed, a few bad suggestions can lead to user dissatisfaction, even among many good suggestions.

The concept of **risk-sensitiveness** was initially proposed and has been developed in the Information Retrieval literature to address this issue in the context of Search Engines. Wang et al. [2012] provided a very intuitive description of sensitivity to risk by decomposing the effectiveness of a model in terms of **gain** and **degradation**. The **gain** of the ***main model*** is the positive difference compared to an Information Retrieval system baseline (aka ***risk-baseline***). **Degradation** is the negative difference between the same models, *i.e.*, *main* and *risk-baseline* models. Usually, in functions that assess the sensibility to the risk, the minimization of degradation receives a higher priority than the gain maximization. A risk-sensitive method improves the quality of overall queries and does not decrease the adequate performance of other ones compared to a baseline system [Zhang et al., 2014].

However, using only one *risk-baseline* system induces a biased evaluation [Dinçer et al., 2014]. Thus, Dinçer et al. [2016] explore the use of many *risk baselines*. The authors claimed in their experiments that using a set of systems as *risk baselines* induces an unbiased way to evaluate the risk sensitiveness, besides assessing the variability of distinct Information Retrieval systems for the same query. They proposed the Z_{RISK} function, which incorporates the Chi-square statistical test to compute the variability concerning all Information Retrieval systems and all queries, defined as:

$$Z_{RISK}(i) = \left[\sum_{q \in Q_+} z_{iq} + (1 + \alpha) \sum_{q \in Q_-} z_{iq} \right] \tag{2.6}$$

where: $z_{iq} = \frac{x_{iq} - e_{iq}}{\sqrt{e_{iq}}}$; $e_{iq} = S_i \times \frac{Q_q}{N}$; and x_{iq} is the effective performance of a query q obtained with the corresponding system i. The element i is defined as $i \in \{1, 2, ..., r\}$ for each system, where r is the number of systems and the element q is defined as $q \in \{1, 2, ..., c\}$, where c is the number of queries. The Q_+ set is composed of queries with positive differences between the *main* method and the *risk-baselines*, representing the **gains**. Similarly, the Q_- set is composed of queries with negative differences, representing the **degradations**. Let $S_i = \sum_{q=1}^{c} x_{iq}$ be the expected system performance for all queries in Information Retrieval system i, $Q_q = \sum_{i=1}^{r} x_{iq}$ the within-query Information Retrieval system effectiveness for the query q, and $N = \sum\limits_{i=1}^{r} \sum\limits_{q=1}^{c} x_{iq}$ the sum of all elements.

However, Z_{RISK} is agnostic to the effective average of the systems, and it does not allow a comparative evaluation for distinct methods. Therefore, in the same work, Dinçer et al. [2016] proposed a Geometric Mean of Z_{RISK}, called G_{RISK} function, defined as:

$$G_{RISK}(i) = \sqrt{S_i / c \times \Phi(Z_{Risk}(i) / c)} \tag{2.7}$$

where: Φ is the cumulative distribution function of the Standard Normal Distribution. The G_{RISK} function is a measure for risk-sensitiveness evaluation regarding distinct Information Retrieval methods. This function has been used in several works in the Learning-to-Rank literature to evaluate the risk-sensitiveness performance [Li et al., 2016; Manotumruksa et al., 2019; Sousa et al., 2016].

Another widely used concept related to the variability of recommendation results is *Fairness*. It is a general term commonly used in Search Engines and RSs concerning the ability of an algorithm or model to treat similar entities in a non-discriminatory way [Pitoura et al., 2021]. Indeed, the average optimization of some quality aspects, such as accuracy, naturally marginalizes minority user groups [Xiao et al., 2017]. RSs must be fair in serving all users, not just the majority, implying a related (but not the same) problem of multiple conflicting objectives.

This thesis focuses on MOF systems through WHF strategies that combine distinct models. However, *fairness* aims to evaluate differences between results for distinct users (or items) of the same model, while *risk-sensitiveness* aims to assess differences between results of separate (baseline) models for all users. Therefore, we will focus on G_{RISK} measure and leave fairness for future work.

2.3 Recommender Systems Approaches

The most commonly used approaches are Content-Based Filtering, Collaborative Filtering, and Hybrid Filtering [Aggarwal, 2016; Jannach et al., 2010; Ricci et al., 2011]. However, since various factors can assess the quality of the recommendations, a new approach has been gaining attention in recent years, the Multi-Objective Filtering. The following subsections discuss these approaches.

2.3.1 Content-based Filtering

The Content-Based Filtering approach explores items' attribute values based on Information Retrieval and Information Filtering techniques. A user profile is built based on items' attributes for which they have previously expressed interest. Thus, the recommendations are based on the similarities of candidate items with this profile.

Content-Based Filtering has three main advantages. First, the recommendation for a user does not depend on other users to achieve accurate results. Second, new items can be immediately recommended after being inserted into the dataset if the attributes' content is available. The third refers to transparency – given that recommendations are obtained based on the items' attributes content, it is easier to explain the recommendations using their attributes.

However, the acquisition of subjective features is a great challenge. Occasionally, user satisfaction would not be associated with the item's attributes but with some subjective impression about the item. Other limitations are the *lack of content*, *overspecialization*, and *acquiring preferences*. Lack of content concerns the availability and usefulness of the items' attributes. Overspecialization means the tendency to make obvious recommendations. The preference acquisition problem is related to the need to know users' preferences to produce a representative profile. If the profile is insufficient to represent the user's preferences, then recommended items tend to be useless to the user.

2.3.2 Collaborative Filtering

In the Collaborative Filtering approach, the input is a set of user ratings on items. The basic idea behind Collaborative Filtering is that users who expressed similar interests in the past will maintain similar claims in the future. Therefore, recommendations are based on predicting ratings for a given item-user pair based on their previously known assigned ratings.

The main advantages of Collaborative Filtering are listed below. Firstly, it is independent of items' attributes and the system's domain. Secondly, the recommendation quality tends to improve with the increase of ratings over time. Finally, it may be able to surprise users with unexpected and pleasant recommendations.

The main drawbacks of this approach are *data sparsity* and *cold start*. Data sparsity occurs when the number of ratings available is insufficient to produce good recommendations. Cold start refers to a particular case of data sparsity in which a new user or new item has no rating associated with them.

2.3.3 Hybrid Filtering

Knowing that each approach has strengths and weaknesses, Hybrid Filtering has the principle of combining strengths while minimizing weaknesses, regardless of the type of combination and exploited approaches [Burke, 2002, 2007]. According to Burke [2007], hybridization focuses on combining different information sources or approaches, or even variants of the same process. For example, a hybrid system that combines only Collaborative Filtering algorithms using ensemble techniques may be effective in improving the accuracy of a rating prediction task [Su and Khoshgoftaar, 2009]. However, more significant benefits may be obtained from hybridization when the considered algorithms address different aspects of the dataset [Ekstrand et al., 2011].

Burke [2002] defined taxonomy for Hybrid Filtering, classifying them into seven different classes, and Jannach et al. [2010] grouped these classes in a more general perspective composed of three base designs: *Monolithic*, *Parallelized*, and *Pipelined.* Table 2.1 shows the three designs, the classes, and the description by Burke [2002] for the types. The *Monolithic* design groups hybridization strategies incorporating aspects from several recommenders in one composed algorithm implementation. The *Parallelized* and *Pipelined* designs combine two or more constituent recommender implementations differing only in the type of the combination applied. In the *Parallelized* design, the constituent algorithms execute independently in parallel, and their combined results define the final recommendation. For the *Pipelined* design, the constituent recommenders execute in a predefined sequence, and the output of one is used as part of the input for the next one. The output from the last recommender in the sequence defines the final recommendation.

Design	**Class**	**Description**
Monolithic	Feature Combination	Features from different recommendation data sources are combined into a single recommendation algorithm.
	Feature Augmentation	Output from one technique is used as an input feature to another.
Parallelized	Mixed	Recommendations from several different recommenders are presented at the same time.
	Weighted	The scores (or votes) of several recommendation techniques are combined together to produce a single derived score.
	Switching	The system switches between recommendation techniques depending on the current situation.
Pipelined	Cascade	One recommender refines the recommendations given by another.
	Meta-level	The model learned by one recommender is used as input to another.

Table 2.1: Hybrid Filtering taxonomy by Jannach et al. [2010] and Burke [2002].

The main hybridization classes related to this thesis are the **Weighted** and the **Switching**. What makes these classes unique in the context of this thesis is that we find some works in the literature that explicitly benefit from *meta-features* to improve their results, an issue addressed in Chapter 3, Section 3.1.

In the Switching class, the strategy is to apply a learning method to choose one from several techniques to be used to comply with a specific request. On the other hand, in the Weighted class, the strategy combines the results of several techniques, typically produced as demonstrated in Equation 2.8. Therefore, this class applies a learning method to define the weight w to be applied to the feature f to obtain the combined score $\hat{s}$, defined as:

$$\hat{s} = \sum_{i=1}^{|\mathcal{F}|} w_i * f_i, \tag{2.8}$$

where: $\hat{s}$ is the hybrid score obtained; $\mathcal{F}$ is a vector of numerical features (traditionally composed of scores generated by various recommendation algorithms); f_i is the i-th feature; and w_i is the computed weight for the $f_i \in \mathcal{F}$.

To better understand these hybrid methods, consider the following hypothetical scenario. Three CF algorithms (*i.e.*, $|\mathcal{F}| = 3$) estimate the relevance of five items for a given user to recommend the two best items. Table 2.2 presents the values predicted by each algorithm for each item. The columns represents the features $\mathcal{F}$ of each item.

Algorithm	Item 1	Item 2	Item 3	Item 4	Item 5
Matrix Factorization	0.468	0.328	0.465	0.464	0.283
K-Nearest Neighbors	0.203	0.402	0.208	0.108	0.486
Singular Value Decomposition	0.356	0.241	0.345	0.311	0.124

Table 2.2: Hypothetical recommendation predictions for a user.

Consider that a model trained by the Switching method chooses the K-Nearest Neighbors algorithm, then the recommendation list is $[5, 2]$. On the other hand, consider that the model trained by the Weighted method determines the following weights for the three algorithms: $[0.48, 0.31, 0.21]$. Then, Table 2.3 presents the predicted hybrid scores for the five items using Equation 2.8. Therefore, according to the hybrid scores, the recommendation list is $[1, 3]$, differing from the Switching method.

Item	Equation 2.8 computation	$\hat{s}_i$
1	0.48 * 0.468 + 0.31 * 0.203 + 0.21 * 0.356	0.362
2	0.48 * 0.328 + 0.31 * 0.402 + 0.21 * 0.241	0.333
3	0.48 * 0.465 + 0.31 * 0.208 + 0.21 * 0.345	0.360
4	0.48 * 0.464 + 0.31 * 0.108 + 0.21 * 0.311	0.322
5	0.48 * 0.283 + 0.31 * 0.486 + 0.21 * 0.124	0.313

Table 2.3: Hypothetical WHF recommendation scores for a user.

Note that the predictions of the Switching method will be the predictions of the chosen base method, while the Weighted method generates new hybrid predictions from the predictions of all the base methods. Therefore, the Switching method is upper bounded by the best base method, and the Weighted method can generate better (or eventually worse) results than the base methods.

2.3.4 Multi-Objective Filtering

Some of the previously stated recommendation quality aspects may be conflicting. For example, accuracy and diversity may be impaired to achieve higher novelty. Thus, Multi-Objective optimization techniques can be applied in Recommender Systems to address multiple objective functions simultaneously. Therefore, in this context, the RSs are usually modeled as a maximization problem:

$$\begin{aligned} \underset{x}{\arg\max} \quad & \mathbf{O}(x) = (O_1(x), O_2(x), ..., O_m(x)), \\ \text{s.t.} \quad & x = (x_1, x_2, ..., x_n) \ \in \ \mathcal{X}, \end{aligned} \tag{2.9}$$

where: $x \in \mathcal{X}$ is a viable solution; $\mathcal{X}$ is the optimization parameter domain; $\mathbf{O} \in \Omega$ is the objective vector; and $\Omega = O(\mathcal{X})$ is the objective space.

This thesis aims to optimize the three ranking objectives described in Section 2.2.1. When more than three objectives are optimized, the complexity of the problem increases considerably, classifying the problem as *Many-Objective* due to the high dimensionality. Thus, when we add risk sensitivity (in Chapter 4) and users' preferences regarding the objectives (in Chapter 5), we introduce these new resources without increasing the number of objectives.

When objective functions are conflicting, it is impossible to obtain a viable solution that simultaneously optimizes all objectives. Thus, some multi-objective optimization methods aim to obtain estimates of the *Pareto-optimal set* [Ehrgott, 2005], which contains the set of non-dominated solutions to the problem. Such solutions arise from the partial ordering induced by the dominance relation defined below (for a maximization problem).

Definition 1. *(Pareto Dominance). One solution $x_2 \in \mathcal{X}$ is said to be weakly dominated by another solution $x_1 \in \mathcal{X}$, that is $x_1 \succ x_2$, in the following rules:*

- $O_i(x_1) \geq O_i(x_2), \ \forall \, i \in [1, m]$, *and*
- $O_j(x_1) > O_j(x_2), \ \exists j \in [1, m]$.

Consider the Pareto set $\mathcal{P}$ defined as the set of non-dominated solutions as $\mathcal{P} = \{x^* \mid \nexists \ x : x \succ x^*\}$. All solutions that are not weakly dominated by any other decision vector of a given set are called non-dominated regarding this set. A solution $x^* \in \mathcal{X}$ is called Pareto-optimal if another solution does not dominate it. The Pareto-optimal set of the multi-objective optimization problem is the set of all Pareto-optimal solutions. The image of this set in the objective space, defined by $\mathbf{O}(\mathcal{P})$, is called the *Pareto front.*

In a multi-objective optimization problem, the goal is to generate a set of samples for the set $\mathcal{P}$. From the generated solutions, the *decision-maker* must select the most promising one according to the application's requirements to produce the recommendations.

Traditional Multi-Objective modeling considers Decision Making a human task, which would be assisted by automated processes that select the most promising Pareto solutions helping the decision-maker choose a solution that best suits their needs. In the Recommender Systems scenario, besides the content provider, each user may potentially be considered a decision-maker themselves. However, it is unrealistic to ask all users to choose their solutions due to application requirements. Hence, the users' decisions need to be automated.

Previous works, such as [Geng et al., 2015; Jugovac et al., 2017; Wang et al., 2016; Zuo et al., 2015], are often based on re-ranking lists of candidate items for the target user. Hence, a preliminary stage defines a list of top-N things according to one objective function (usually based on rating prediction). A second stage produces a list of top-K ($K << N$) re-ranked items by reordering the top-N list according to other (different) objective functions. Jugovac et al. [2017] propose a strategy that considers *user's tendencies* measured by different quality dimensions while maintaining high accuracy.

However, re-ranking methods can naturally bias the objective used in the preliminary stage [Cai et al., 2020]. Indeed, the concept of a successful recommendation can vary widely among users. Certain groups of users may prefer recommendations for newer or more miscellaneous items to the detriment of relevant ones. Others may not admit less relevant items, and finally, we can have moderate users. Thus, modern Recommender Systems must meet these different needs by optimizing all multiple objectives simultaneously. Re-ranking strategies do not adequately deal with this requirement. In contrast, Ribeiro et al. [2012, 2014] define the ranking in just one stage by scoring the items considering multiple objectives. They proposed the Pareto-Efficient Hybridization (PEH), discussed in the following subsection, which inspired the definition of the *Meta-featured* MOF described in Chapter 3.

Pareto-Efficient Hybridization

The Pareto-Efficient Hybridization (PEH) [Ribeiro et al., 2012, 2014] is a typical WHF that combines a variety of *constituent algorithms*, as described in Section 2.3.3. PEH models a solution x as a vector of weights $[w_1, w_2, ..., w_{|\mathcal{F}|}]$, in which $\mathcal{F}$ is the *constituent algorithms*, each weight is associated with an algorithm, and the linear combination specified in Equation 2.8 defines the final scores. Therefore, a meta-heuristic obtains a set of Pareto solutions to optimize the multiple objectives and a decision-making process chooses one of the solutions found to carry out the recommendations.

For the decision-making process they select the most promising solution by applying a linear search in the solution space that maximizes the weighted average of the objective functions for all users:

$$\underset{x \in \mathcal{X}}{\arg\max} \quad \sum_{i=1}^{m} q_i * O_i(x), \tag{2.10}$$

where: q_i is the weight representing the importance of the objective function O_i.

Ribeiro et al. [2012, 2014] highlighted that the importance of each objective function might vary depending on the target user, considering that the system might prioritize accuracy for new users. In contrast, for old users, novelty may be more critical. Thus, they made it possible to select the most promising solution for the target user's needs, but considering **only *ad hoc* weights**. They exploited four configurations: (a) **PEH-mean:** [*acc*: 0.33, *nov*: 0.33, *div*: 0.33]; (b) **PEH-acc:** [*acc*: 0.70, *nov*: 0.30, *div*: 0.00]; (c) **PEH-nov:** [*acc*: 0.15, *nov*: 0.50, *div*: 0.35]; and (d) **PEH-div:** [*acc*: 0.10, *nov*: 0.35, *div*: 0.55]. Despite the importance of appropriate choices to meet the specific user's needs, the authors did not detail how to choose the proper weights for each user among these configurations. Another critical issue they do not consider is the influence of input data characteristics on the constituent algorithms, an issue addressed in Chapter 3, Section 3.1.

To better understand the PEH method, consider the hypothetical scenario described in Section 2.3.3. Table 2.4 presents the results of a set of Pareto solutions found during the simulation of an MO search, representing the weights for each feature (*i.e.*, algorithms from Table 2.2), the calculated value for each objective from the lists generated with the hybrid scores obtained by using these weights in Equation 2.8, and the result of the sum used in Equation 2.10 for each of the four previously defined PEH configurations.

Solution	Weights	Objectives			PEH Configuration results			
		NDCG	EPD	EILD	mean	acc	nov	div
1	0.07, 0.58, 0.35	0.5596	0.3664	0.1878	0.3713	0.5016	0.3329	0.2875
2	0.61, 0.07, 0.32	0.5802	0.3614	0.1836	0.3751	0.5146	0.3320	0.2855
3	0.29, 0.28, 0.43	0.5809	0.3757	0.1994	0.3853	0.5193	0.3448	0.2993
4	0.35, 0.24, 0.41	0.5627	0.4083	0.1888	0.3866	0.5164	0.3546	0.3030
5	0.18, 0.34, 0.48	0.5528	0.3873	0.2014	0.3805	0.5031	0.3471	0.3016
6	0.27, 0.17, 0.56	0.5651	0.3631	0.1743	0.3675	0.5045	0.3273	0.2795
7	0.33, 0.48, 0.19	0.5848	0.4028	0.1677	0.3851	0.5302	0.3478	0.2917
8	0.46, 0.23, 0.31	0.5844	0.3880	0.1935	0.3886	0.5255	0.3494	0.3007
9	0.08, 0.37, 0.55	0.5856	0.3865	0.2167	0.3963	0.5259	0.3569	0.3130
10	0.32, 0.27, 0.41	0.5811	0.4041	0.2170	0.4007	0.5280	0.3652	0.3189

Table 2.4: Hypothetical PEH recommendation search results.

Considering these simulated results, when applying Equation 2.10, the choice for the PEH-acc configuration is solution 7, while for all other three configurations, the choice is solution 10. Then, Table 2.5 presents the predicted hybrid scores for the five items using Equation 2.8 with the weights defined by solution 7. Therefore, according to the hybrid scores, the recommendation list defined by PEH-acc is $[5, 2]$.

Item	Equation 2.8 computation	$\hat{s}_i$
1	0.33 * 0.468 + 0.48 * 0.203 + 0.19 * 0.356	0.320
2	0.33 * 0.328 + 0.48 * 0.402 + 0.19 * 0.241	0.347
3	0.33 * 0.465 + 0.48 * 0.208 + 0.19 * 0.345	0.319
4	0.33 * 0.464 + 0.48 * 0.108 + 0.19 * 0.311	0.264
5	0.33 * 0.283 + 0.48 * 0.486 + 0.19 * 0.124	0.350

Table 2.5: Hypothetical recommendation scores considering solution 7.

On the other side, Table 2.6 presents the predicted hybrid scores for the five items using Equation 2.8 with the weights defined by solution 10. Therefore, according to the hybrid scores, the recommendation list defined by PEH-mean/nov/div is $[1, 3]$.

Item	Equation 2.8 computation	$\hat{s}_i$
1	0.32 * 0.468 + 0.27 * 0.203 + 0.41 * 0.356	0.351
2	0.32 * 0.328 + 0.27 * 0.402 + 0.41 * 0.241	0.312
3	0.32 * 0.465 + 0.27 * 0.208 + 0.41 * 0.345	0.346
4	0.32 * 0.464 + 0.27 * 0.108 + 0.41 * 0.311	0.305
5	0.32 * 0.283 + 0.27 * 0.486 + 0.41 * 0.124	0.273

Table 2.6: Hypothetical recommendation scores considering solution 10.

2.4 Concluding remarks

This chapter presents the main concepts related to this work to understand this thesis better. More specifically, we address the recommendation problem, the evaluation of recommender systems, and the main recommendation approaches. We emphasize that we focus on: (a) the Top-N recommendation problem; (b) the evaluation criteria of accuracy, novelty, and diversity; (c) the risk-sensitive of the three evaluation criteria; and (d) the hybrid and multi-objective filtering approaches. In the next chapter, we introduce a new multi-objective recommender method based on the explicit use of *meta-features* to answer our first research question.

Chapter 3

Meta-featured MOF

In this chapter, we exploit *meta-features* in the scope of MOF to answer our first research question: ***RQ1***: *"Does explicitly incorporating meta-features contribute to improving the results of multi-objective recommendation?"*. We also intend to acquire knowledge of MO methods compared to WHF methods.

To accomplish this, we firstly contextualize the influence of input data characteristics in the scope of RSs and hybrid methods that make explicit use of data characterization measures in Section 4.1. Then, in Section 3.2 we define a multi-objective strategy based on the PEH method discussed in the previous chapter and the hybridization strategies discussed in Section 4.1. We also define a set of *meta-features*, select a group of *constituent algorithms*, and define a strategy to reduce the number of features, selecting the features that supposedly have the most significant potential to contribute to the hybridization process in Section 3.3. In Section 3.4, the results of empirical experiments allow us to give a positive answer to our ***RQ1*** and provide interesting observations about the behavior of the methods in different experimentation scenarios. Finally, Section 3.5 concludes the chapter with some considerations.

3.1 The Influence of Input Data Characteristics

In previous studies, researchers have shown that characteristics of the input data may influence the accuracy performance of recommender algorithms. Breese et al. [1998]; Gunawardana and Shani [2009]; Herlocker et al. [2004]; and Adomavicius and Zhang [2012] compared the predictive accuracy of various Collaborative Filtering methods in distinct domains. Their conclusions highlighted that the best algorithm depends on specific factors, such as application domain, rating value scale, and the number of users, items, and ratings. Although these works focus on Collaborative Filtering, it would be feasible that the input data characteristics also influence other approaches.

Cheng et al. [2018] produced a comprehensive analysis of the influence of the rating data characteristics in a user-level perspective for the rating prediction task. They evaluated how six user rating characteristics (*mean*, *variance*, *density*, *popularity mean*, *Eigenvector centrality*, and *clustering coefficient*) influenced three popular Collaborative Filtering algorithms (user-based nearest neighborhood, item-based nearest neighborhood, and matrix factorization), demonstrating significant effects of the data characteristics for the prediction accuracy criterion. The authors argue that evaluating the performance of algorithms by measuring accuracy for all users does not adequately reflect the quality of recommendations from the individual users' interests.

Deldjoo et al. [2021] evaluated the explanatory power of various data characteristics on the performance of CF algorithms on accuracy and fairness. The Data characteristics used are related to the structure of the rating matrix, frequency and distribution of rating values, and item popularity, for instance. They observed that such characteristics have high explanatory power of the results, although much greater for accuracy than fairness. The authors focus on statistical techniques for feature selection, aiming to obtain a minimum set of features with maximum explanatory power. Therefore, despite proving the influence of data characteristics on the final results, they do not explicitly use this information to make better recommendations.

Indeed, few works make explicit use of the input data characteristics. Three works that we find in the literature, [Cunha et al., 2016, 2018a,b], [Sill et al., 2009], and [Bao et al., 2009], make explicit use of input data characteristics in Hybrid Filtering strategies for best recommendation results and will be described in the following subsections. Each work presented below uses its nomenclature for its features, often specific to the work context or generic to being confused with other contexts. Thus, we will use the term ***meta-feature*** in this thesis to generalize for the most varied characterization measures of input data but able to differentiate from other contexts. On the other hand, the recommendation algorithms that contribute as input to the hybridization process are referred to as ***constituent algorithms*** (term used by Ribeiro et al. [2012, 2014], described in Section 2.3.4). Therefore, these works use two types of data in their hybridization process: (a) the *constituent algorithms*; and (b) the *meta-features.*

In recent work, Penha and Santos [2020] also explored additional features to improve WHF. They used algorithm performance estimators as *meta-features* (instead of statistical measures from input data) in the single-objective optimization, achieving exciting results. Here, we are interested in multiple objectives and *meta-features* that can be computed from the input data regardless of the *constituent algorithms*, leaving other types of *meta-features* for future work.

3.1.1 Meta-learning for Switching Hybrid Filtering

Cunha et al. [2016, 2018a,b] explored Meta-learning techniques taking advantage of *meta-features* in the Collaborative Filtering scope. The authors explored a set of diverse *meta-features*, including *rating distribution*, *neighborhood*, and *graph-based measures* computed for particular items or users and the entire dataset, creating a model to predict the best *constituent algorithm* for a given dataset (*i.e.*, the algorithm selection problem) considering single and multiple objective optimizations.

Figure 3.1 illustrates general meta-learning processing for algorithm selection [Cunha et al., 2018b; Pinto et al., 2016]. Firstly, the performances of the *constituent algorithms* and the *meta-features* are computed, composing the Meta-Knowledge. Secondly, a Learning Algorithm is used to create the model for the algorithm selection. Finally, when it is necessary to use a new dataset, its *meta-features* are extracted, and the model is applied to select the best *constituent algorithm*. This process can be adapted to make choices for specific contexts, considering the user or item involved in the rating prediction or recommendation request, for instance. Thus, this method can be classified as *Switching Hybrid Filtering*.

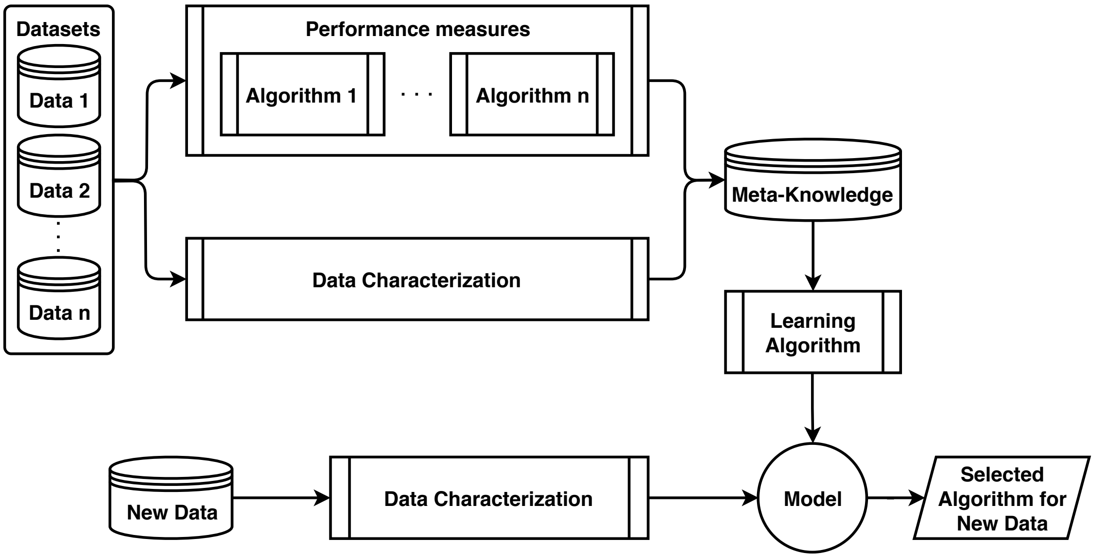


Figure 3.1: The Meta-learning general processing for algorithm selection (adapted from [Cunha et al., 2018b; Pinto et al., 2016]).

One significant limitation in Meta-learning is the computation of the *constituent algorithms*' performances. It is an essential requirement that the performance computation be faster than the execution of the algorithms themselves. Another relevant limitation is that the result obtained will be upper bounded by the performance of the best algorithm, also considering that occasionally it may not be the one chosen by the model. However, WHF methods mitigate this limitation through the linear combination of the *constituent algorithms* instead of choosing a single algorithm for each request. This strategy has the potential to generate better results than those presented individually by the *constituent algorithms*. We describe the two *meta-featured* WHF we found in the literature below.

3.1.2 Stacking Recommendation Engines with Additional Meta-features

Bao et al. [2009] proposed the **STacking Recommendation Engines with Additional Meta-features** (**STREAM**), based on a two-level stacking strategy. The first level consists of building input features composed of *constituent algorithms* and *meta-features*. The second level combines the input features to obtain a final prediction to produce recommendations, according to Equation 2.8. Ensemble strategies perform the combination of the input features, commonly by using regression methods (linear or non-linear), learning a model to define the weights applied to each input feature. Figure 3.2 shows the flowchart of the STREAM processes, and Figure 3.3.a shows their blended prediction scheme. Note that the weights are defined individually for each *meta-feature* (M_i) and each *constituent algorithm* (P_i).

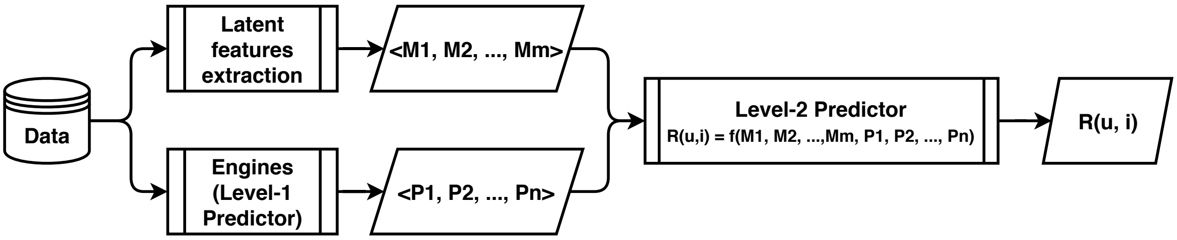

Data
Latent features extraction
<M1, M2, ..., Mm>
Engines (Level-1 Predictor)
<P1, P2, ..., Pn>
Level-2 Predictor
R(u,i) = f(M1, M2, ...,Mm, P1, P2, ..., Pn)
R(u, i)

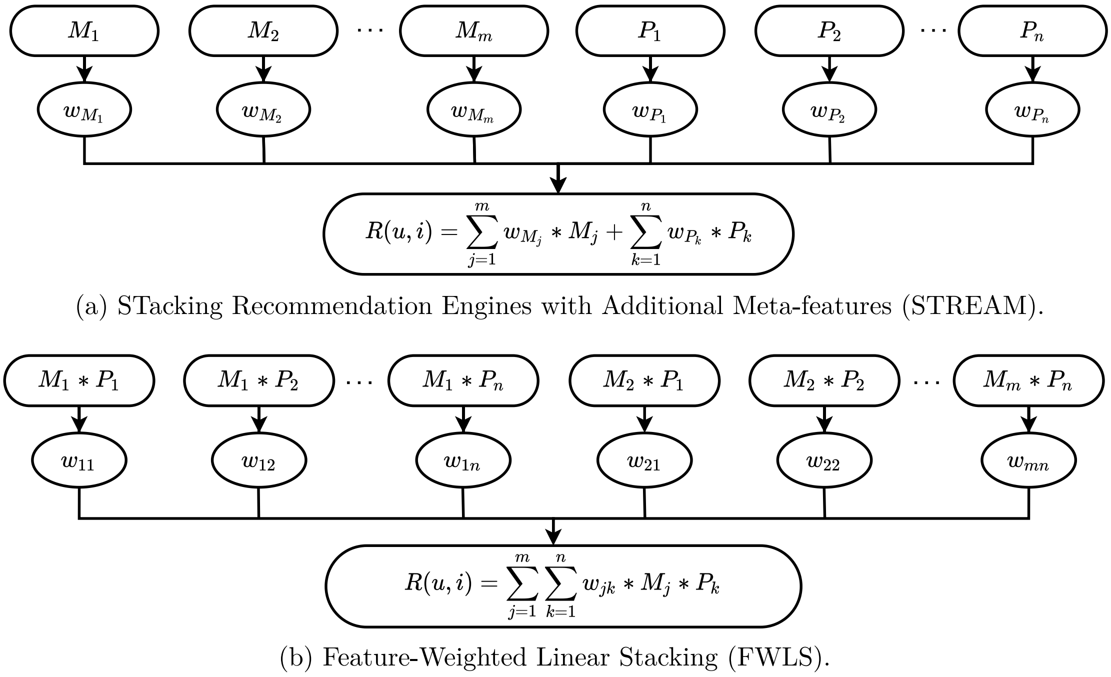
Figure 3.2: The STacking Recommendation Engines with Additional Meta-features framework (adapted from [Bao et al., 2009]).

The authors show empirical evidence that their strategy outperforms all the *constituent algorithms* used as Level-1 Predictors concerning the rating prediction accuracy, achieving better results for nonlinear methods. Although the authors report the exploration of Collaborative Filtering and Content-Based Filtering, the use of content is restricted to the users' and items' neighborhood definition, predicting ratings based on Collaborative Filtering approaches. They concluded that using different *meta-features* reached similar prediction accuracy as those obtained using only the number of ratings.

3.1.3 Feature-Weighted Linear Stacking

Sill et al. [2009] proposed the **Feature-Weighted Linear Stacking** (**FWLS**), combining a variety of *meta-features* and *constituent algorithms* similarly to STREAM. However, they argue that stacking strategies based on nonlinear methods usually require a lot of tuning and training time to learn the weights. Then, they propose a simple but efficient strategy that uses linear methods.

Standard linear regression methods become feasible by using a strategy to combine the *constituent algorithms* and *meta-features* to produce a more extensive set of input features. Figure 3.3.b shows the blended prediction scheme for FWLS. Note that, differently from the scheme used by STREAM (Figure 3.3.a), the input features are computed for each possible pair of *meta-feature* (M_i) and *constituent algorithm* (P_i), defined by multiplying their values. Defining a *meta-feature* and a *constituent algorithm* that always returns the value 1 (one) enables the individualization of the input features.



(a) STacking Recommendation Engines with Additional Meta-features (STREAM).

(b) Feature-Weighted Linear Stacking (FWLS).

Figure 3.3: Blended prediction schemes for STREAM and FWLS strategies (adapted from [Sill et al., 2009]).

The authors show empirically that their hybrid method outperforms the *constituent algorithms* when evaluating the rating prediction accuracy. Unfortunately, despite the similarities with STREAM, they did not compare both methods, exploiting *meta-features* and *constituent predictions* combined in the same way as STREAM for blending prediction using only the same linear regression model applied in their method.

To better understand the STREAM and FWLS methods, consider the hypothetical scenario described in Section 2.3.3. In addition to the predictions of the algorithms presented above in Table 2.2, we hypothesized two *meta-features*: *ratings average* and the *percentage of ratings*. Table 3.1 presents the *meta-features* values for each item followed by the same values of Table 2.2.

Meta-feature (M) or Algorithm (P)	Item 1	Item 2	Item 3	Item 4	Item 5
Ratings average (M_1)	0.419	0.585	0.510	0.451	0.127
Percentage of ratings (M_2)	0.549	0.265	0.167	0.257	0.803
Matrix Factorization (P_1)	0.468	0.328	0.465	0.464	0.283
K-Nearest Neighbors (P_2)	0.203	0.402	0.208	0.108	0.486
Singular Value Decomposition (P_3)	0.356	0.241	0.345	0.311	0.124

Table 3.1: Hypothetical *meta-features* and *constituent algorithms* values.

The STREAM method defines the features $\mathcal{F}$ for each item as the five values of the concatenation of *meta-features* and *constituent algorithms* (*i.e.*, $|\mathcal{F}| = 5$ and $\mathcal{F} = [M_1, M_2, P_1, P_2, P_3]$). Consider that the model trained by the STREAM method determines the following weights for the five features: $[0.03, 0.09, 0.18, 0.35, 0.40]$. Then, Table 3.2 presents the predicted STREAM scores for the five items using Equation 2.8. Therefore, according to the STREAM scores, the recommendation list is $[1, 5]$.

Item	Equation 2.8 computation	$\hat{s}_i$
1	0.03 * 0.419 + 0.09 * 0.549 + 0.18 * 0.468 + 0.35 * 0.203 + 0.4 * 0.356	0.360
2	0.03 * 0.585 + 0.09 * 0.265 + 0.18 * 0.328 + 0.35 * 0.402 + 0.4 * 0.241	0.338
3	0.03 * 0.510 + 0.09 * 0.167 + 0.18 * 0.465 + 0.35 * 0.208 + 0.4 * 0.345	0.325
4	0.03 * 0.451 + 0.09 * 0.257 + 0.18 * 0.464 + 0.35 * 0.108 + 0.4 * 0.311	0.282
5	0.03 * 0.127 + 0.09 * 0.803 + 0.18 * 0.283 + 0.35 * 0.486 + 0.4 * 0.124	0.347

Table 3.2: Hypothetical STREAM recommendation scores for a user.

On the other hand, the FWLS method defines the features $\mathcal{F}$ for each item as the combination of *meta-features* and *constituent algorithms* (*i.e.*, $|\mathcal{F}| = 6$ and $\mathcal{F} = [M_1 * P_1, M_1 * P_2, M_1 * P_3, M_2 * P_1, M_2 * P_2, M_2 * P_3]$). Consider that the model trained by the FWLS method determines the following weights for the five features: $[0.23, 0.28, 0.18, 0.05, 0.24, 0.02]$. Then, Table 3.3 presents the predicted FWLS scores for the five items using Equation 2.8. Therefore, according to the FWLS scores, the recommendation list is $[2, 1]$.

Item	Equation 2.8 computation	$\hat{s}_i$
1	0.23 * 0.196 + 0.28 * 0.085 + 0.18 * 0.149 + 0.05 * 0.257 + 0.24 * 0.111 + 0.02 * 0.195	0.139
2	0.23 * 0.192 + 0.28 * 0.235 + 0.18 * 0.141 + 0.05 * 0.087 + 0.24 * 0.107 + 0.02 * 0.064	0.167
3	0.23 * 0.237 + 0.28 * 0.106 + 0.18 * 0.176 + 0.05 * 0.078 + 0.24 * 0.035 + 0.02 * 0.058	0.129
4	0.23 * 0.209 + 0.28 * 0.049 + 0.18 * 0.140 + 0.05 * 0.119 + 0.24 * 0.028 + 0.02 * 0.080	0.101
5	0.23 * 0.036 + 0.28 * 0.062 + 0.18 * 0.016 + 0.05 * 0.227 + 0.24 * 0.390 + 0.02 * 0.100	0.135

Table 3.3: Hypothetical FWLS recommendation scores for a user.

Considering the limitation of Switching Hybrid Filtering, which is upper bounded by the best *constituent algorithm*, in this thesis, we focus on the potential of *meta-features* to provide gains in WHF methods such as STREAM and FWLS.

To the best of our knowledge, *meta-features* has not been explicitly exploited in MOF systems. In the next section, we begin to tackle this challenge by proposing a *Meta-featured* MOF strategy.

3.2 The Meta-featured MOF strategy

This section presents a multi-objective recommendation strategy that explicitly uses *meta-features*. We improve the PEH strategy (described in Section 2.3.4) by incorporating the STREAM or FWLS strategies. From an inverse perspective, we also improve the STREAM or FWLS strategies by incorporating the PEH strategy.

As discussed in Section 2.3.4, a multi-objective optimization is traditionally composed of two main tasks: (a) *Search*: the search for a general Pareto solution set; and (b) *Decision making*: the choice for a Pareto solution to be used in generating the result. However, to define a *Meta-featured* Multi-Objective Filtering strategy, one must consider many other tasks. Thus, we followed the PEH strategy [Ribeiro et al., 2012, 2014], adapting the input data to explore the *meta-features* instead of just *constituent algorithms*. Figure 3.4 shows the overall Meta-featured MOF strategy into three stages: (a) *Pre-processing*: responsible for building the hybridization features, which will be composed of a combination of *meta-features* and *constituent algorithms*; (b) *Modeling*: responsible for finding a Pareto set and selecting the solutions to be used to make recommendations; and (c) *Recommending*: responsible for defining scores and ranking items, generating and presenting recommendation lists, collecting users' feedback, and updating the dataset.

The pre-processing stage composes the set of features $\mathcal{F}$ according to different strategies, such as STREAM and FWLS, discussed in Sections 3.1.2 and 3.1.3, respectively. Therefore, the multi-objective strategy in the modeling stage represents a WHF recommender. As described earlier, *meta-features* have not yet been explored in MOF systems, becoming one of the motivations for this work through ***RQ1***. Thus, we detail the processing of the feature in the next section.

For the MO *search* task, an evolutionary algorithm receives the set of features $\mathcal{F}$ and the solutions are modeled as weights, one for each input feature. Several MO general-purpose evolutionary algorithms include SPEA2 [Zitzler et al., 2001], AMOSA [Li et al., 2016], and NSGA-II [Deb et al., 2002]. They all apply Pareto dominance (Definition 1) to deal with MO optimization. Here we use the Non-dominated Sorting Genetic Algorithm II, or NSGA-II, an important state-of-the-art evolutionary MO algorithm [Yliniemi and Tumer, 2016], successfully used in similar recommendation works [Filatovas et al., 2017; Jain et al., 2020; Lin et al., 2018]. We use the implementation available in the JMetal

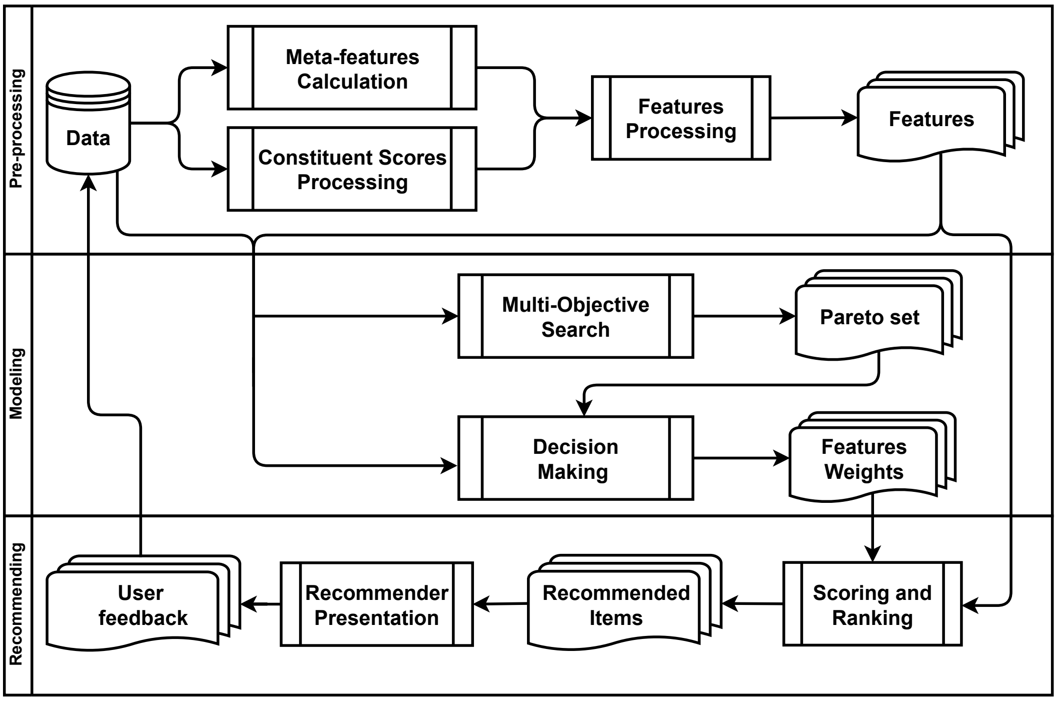


Figure 3.4: The overall Meta-featured MOF strategy.

framework [Durillo and Nebro, 2011; Nebro et al., 2015] as a black box.

We also followed the *decision-making* task from the PEH strategy, applying their original linear search defined in Equation 2.10. However, they make only one global choice, choosing a single Pareto solution to recommend to all users. In addition to this global mechanism, we apply the same equation individually to each target user.

After selecting a Pareto solution, we compute the scores for candidate items by applying Equation 2.8. Further, we determine recommendations by sorting the scores obtained. Finally, the Recommender Presentation shows the recommended items and updates the dataset with collected feedback.

The data pre-processing and the multi-objective search consume more processing time. These processes run offline whenever the data undergoes several changes that affect the features. On the other hand, decision-making and recommending are less time-consuming processes, generating recommendations online or even pre-define suggestions offline. This flexibility makes the method applicable to generate real-time recommendations for real applications.

3.3 Features processing

The features processing consists of building the input features $\mathcal{F}$, based on *meta-features* and *constituent algorithms*, to be used in the MO and hybridization processes. Thus, the computation of *meta-features* and *constituent algorithm* results are the first tasks that can run in parallel. However, the processing of the feature consists of a sequence of other two tasks: *features building* and *features selection.* These four tasks are detailed next.

3.3.1 *Meta-Features* computation

The *meta-features* definition and extraction would be a complex task and can vary greatly depending on input data, the recommender approaches and algorithms, and the application domain. Thus, the definition of applicable *meta-features* can be seen as an *art* [Sill et al., 2009], which these many factors may guide.

In the Content-Based Filtering approach, the input contains items' attributes representing their content. Thus, the *meta-features* would be related to measures applied to the item content and similarities calculated for pairs of items. On the other hand, in the Collaborative Filtering approach, the input is a *rating matrix* mapping the user satisfaction with items. Thus, the characteristics extracted from input data would be related to: (a) the number of ratings involved; and (b) the distribution of their values. Therefore, we compute CF *meta-features* by applying statistical measures to the ratings of an item or a user.

We exploit a set of general measures to characterize the input data related to Content-Based Filtering and Collaborative Filtering approaches in a broad scope based on Bao et al. [2009]; Hurley and Rickard [2009]; Sill et al. [2009]; and Adomavicius and Zhang [2012]. More specific metrics that consider dataset and *constituent algorithms* characteristics have the potential to contribute more to the final results. However, we will leave this issue for future work, and we will focus on achieving our goals for this work with general measures.

Table 3.4 presents the measures used in our experiments. We restricted the *meta-feature* values from 0 (zero) to 1 (one). Thus, if the measure returns values in other ranges, the returned values are normalized. For details on the *meta-features* computation, see Appendix A.

Class	Name	Description
CB	Cosine	Quantifies the similarity between an item and other items using the Cosine Similarity [Baeza-Yates and Ribeiro-Neto, 1999] between two items.
	Dice	Quantifies similarities, as well as COSINE. The difference from COSINE is only using the Dice Coefficient [Adar et al., 2009] as a similarity measure. Adar et al. [2009] initially used Dice Coefficient to compute the differences between two versions of the same *document* over time. However, we use it to compute the differences between the content from two different *items* without loss of generality.
	Entropy	Quantifies the cohesiveness of the item's content via the Entropy measure [Bendersky et al., 2011]. Low values for Entropy indicate a tendency for the content to cover a single topic.
	Jaccard	Ii is another measure that quantifies similarities, as well as COSINE and DICE. The difference is using the Jaccard Index [Baeza-Yates and Ribeiro-Neto, 1999] as a similarity measure.
CF	PCR	The Proportion of Common Ratings (PCR) captures a notion of the size of the neighborhood via the concept of *users in common* and *items in common* [Fortes et al., 2017]. *Users in common* are those who ranked the same items that a particular user ranked and *items in common* are those that were ranked by the same users who ranked a particular item.
	PR	The Proportion of Ratings (PR) captures a notion of the amount of ratings available via the percentage of the number of ratings given by a user or received by an item [Fortes et al., 2017].
	Gini	Captures a notion of the rating values distribution using the Gini Index [Hurley and Rickard, 2009]. Gini measures the inequality of the ratings values, when the result is zero it expresses a perfect equality, but when the result is one it expresses the maximal inequality.
	Pearson	Captures the rating values distribution using Pearson's Coefficient of Variation (CV).
	PqMean	Captures the rating values distribution using the *pq*-mean metric [Hurley and Rickard, 2009].
	SD	Captures the rating values distribution using the Standard Deviation (SD).

Table 3.4: *Meta-feature* measures exploited in this work (see Appendix A for details about the *meta-features* computation).

3.3.2 *Constituent Algorithms* computation

We compute the *Constituent Algorithms* by running recommendation algorithms that produce the scores for the candidate items. Table 3.5 summarizes all *constituent algorithms* exploited in this work.

We implement the purely Content-Based Filtering algorithm using Apache Lucene [McCandless et al., 2010]. The algorithm indexes the items' content, building the user profile from their known preferred item's content and returning a list of ranked items defined by the similarity scores from the user's profile and item's content. Thus, this algorithm does not perform rating predictions.

For the *CF constituent algorithms*, we include most of the state-of-the-art techniques, such as K-Nearest Neighbors (KNN), Matrix Factorization (MF), and Singular Value Decomposition (SVD), as black boxes. For detail about their implementations, consult their respective references.

Class	Name	Short Description and references
CB	CB-Lucene	Based on similarity scores between user's profile and item's content [Fortes et al., 2017].
CF	ALS	A biased MF using Alternating Least Squares [Ekstrand, 2020].
	BP-SlopeOne	The Bi-Polar Frequency-Weighted Slope-One [Gantner et al., 2011].
	Bias	A basic user-item bias algorithm [Ekstrand, 2020].
	Biased-MF	An MF using user and item bias [Gantner et al., 2011].
	Biased-SVD	A biased Singular Value Decomposition [Ekstrand, 2020].
	BPR	The Bayesian Personalized Ranking [Rendle et al., 2012].
	Implicit-MF	An implicit MF using Alternating Least Squares [Ekstrand, 2020].
	ItemKNN	An Item-based KNN [Ekstrand, 2020].
	NCF	The Neural Collaborative Filtering [He et al., 2017].
	SlopeOne	The Frequency-Weighted Slope-One [Gantner et al., 2011].
	SVDPlusPlus	The Singular Value Decomposition Plus Plus [Gantner et al., 2011].
	UserKNN	A User-based KNN [Ekstrand, 2020].

Table 3.5: *Constituent algorithms* exploited in this work.

3.3.3 Features building

The features building is responsible for combining the *meta-features* and the *constituent algorithms* with the features used as input for WHF. This thesis exploits the strategies defined in the WHF methods from STREAM [Bao et al., 2009] and FWLS [Sill et al., 2009], discussed in Section 3.1. For comparison reasons, we exploited the third strategy as a baseline, which did not use *meta-features*, and we called this method Hybrid Recommender (HR).

We mathematically formalized the three feature building methods exploited in this thesis, generalizing beyond the task of predicting ratings from the scope of Collaborative Filtering. Consider $\hat{\mathcal{S}}$ as the vector of *constituent algorithm* scores and $\mathcal{M}$ as the vector of *meta-features*. The STREAM, FWLS, and HR features are defined as:

$$\mathcal{F}_{\text{STREAM}} = [\ m \mid \forall m \in \mathcal{M}\ ] \frown \left[\ s \mid \forall s \in \hat{\mathcal{S}}\ \right], \tag{3.1}$$

$$\mathcal{F}_{\text{FWLS}} = \left[\ m * s \mid \forall (m, s) \in \mathcal{M} \times \hat{\mathcal{S}}\ \right], \tag{3.2}$$

$$\mathcal{F}_{\text{HR}} = \left[\ s \mid \forall s \in \hat{\mathcal{S}}\ \right]. \tag{3.3}$$

where: $\frown$ represents the concatenation operator, and × represents the Cartesian product between the two vectors.

Therefore, the Features building creates the vector $\mathcal{F}$ according to these equations. A vector $\mathcal{F}$ represents the features for an item-user pair, combining the scores generated by the *constituent algorithms* for the user-item pair and the *meta-feature* values specifically computed for the user and the item individually.

3.3.4 Features selection

The vector of features $\mathcal{F}$ is composed of various numerical values originating from *meta-features* and *constituent algorithms*. Some of them may be useless in some aspects and thus could be discarded. Therefore, feature selection is responsible for analyzing the feature values and for selecting those that have the potential to be more helpful. Reducing the number of features can promote more accurate results, facilitate the storage used, or reduce the processing time required.

In this work, three tasks executed in sequence complete the features selection, described as follows:

- **Variability selection**: evaluates the variability of the values to discard those that have a low variability according to a threshold value, considering that if the values assumed by a feature have low variability, it will be little descriptive to be helpful in a regression method;

- **Correlation selection**: evaluates the correlation between all pairs of features, discarding one when they have a high correlation according to a threshold, considering that if two features have a high correlation, they are equivalent;

- **Regression selection**: interactively learn linear regression models by assuming one feature as the dependent variable and the remaining features as independent variables, discarding this feature if the coefficient of determination is greater than or equal to a defined threshold, considering that if it is possible to find a good linear regression model, this feature is equivalent to a combination of the others.

3.4 Experimental results

This section presents the experiments carried out to answer the research question ***RQ1*** and increase our knowledge of *Meta-featured* MOF systems. We analyzed the results of the MO strategy presented above, comparing each feature-building process described. In addition, we also evaluate whether the proposed feature selection and the individual decision-making are also helpful in producing better results.

Firstly, Section 3.4.1 describes the primary resources and setup to conduct the experiments. Section 3.4.2 presents three experimental analyses. Finally, in Section 3.4.3, some discussions about the observed results.

3.4.1 Experimental setup

Experimental Strategy. We exploit a 5-fold cross-validation procedure. Firstly, we select five folds by applying a stratified random sampling based on the number of users' ratings. Secondly, we tune the *constituent algorithms* and calculate the *meta-features* for each combination of folds (always leaving one of the folds out for testing), preparing the input features for tuning the hybrid algorithms. Thirdly, we tune the hybrid algorithms with the features prepared earlier. With the best configurations chosen, we prepare the data for the final tests (considering each fold left out previously), run the constituent algorithms, calculate the *meta-features*, prepare the new features for testing, and run the hybrid algorithms. Finally, we generate the final recommendations and evaluate the results for the *Top-5* task. For details on the experimental strategy and methods tuning, see Appendix B.

Datasets. We exploit four datasets: (a) Amazon (Books) [He and McAuley, 2016; McAuley et al., 2015]; (b) Bookcrossing [Ziegler et al., 2005]; (c) Jester [Goldberg et al., 2001]; and (d) Movielens (20M) [Herlocker et al., 1999]; with particular characteristics, such as application domain, number of elements, rating scales, and descriptive content. We normalize the rating values in the range of $[0, 1]$ by dividing them by the maximum rating value (*e.g.,* for Amazon, we divide by 5. Thus, the normalized ratings are in the range of $[0.2, 1.0]$). For Jester, before the normalization, the original ratings are processed to be in the range of $[1, 21]$. Therefore, since the lowest rating value before normalization is always 1, we will not have normalized ratings with a value of 0 (zero). For the Bookcrossing dataset, we have used only explicit ratings. To evaluate recommendation lists composed of more than 5 items, we eliminated users with less than 10 ratings. From the remaining users, we randomly selected 15.000 users for Amazon, Jester, and Movielens and 1.250 users for Bookcrossing of each testing folder to generate and evaluate the final results. Table 3.6 lists the characteristics of these datasets.

Name (domain)	#Ratings (scale)	#Users #Items	Sparsity	Content
Amazon (books)	≈22.5M (1 to 5)	8,026,324 2,330,066	0.9999987	Books' title, description and related items (*e.g.,* also bought, also viewed, and bought together).
Bookcrossing (books)	≈433K (1 to 10)	77,805 185,973	0.9999700	Books' language, category, description, and editorial review.
Jester (jokes)	≈4M (-10 to 10)	73,421 100	0.4366244	Jokes' text content.
Movielens (movies)	≈22M (1 to 5)	138,493 26,744	0.9946001	Movies' title, genre, and plot and users' age, gender, and occupation.

Table 3.6: Datasets used in the experiments. The numerical characteristics are extracted from the original datasets. Sparsity is defined as $1 - \frac{\#\text{Ratings}}{\#\text{Users} * \#\text{Items}}$.

Features. Section 3.3 describes how we built features $\mathcal{F}$ used for hybridization. We exploit the set of measures listed in Table 3.4 (and detailed in Appendix A), totaling twenty-eight different *meta-features*: (a) sixteen from content-based measures computed for items; and (b) twelve from collaborative filtering measures computed for items and users. The datasets exploited are very diverse, making it difficult to execute all *constituent algorithms* listed in Table 3.5: CB-Lucene for Amazon and Movielens; and both Slope One algorithms for Amazon. Consequently, CB *meta-features* were not used in Amazon and Movielens. The feature selection exploited *Gini Index* [Hurley and Rickard, 2009] for variability, *Pearson* for correlation, and R^2 for regression. We preliminary evaluated different threshold values and chose the one that discarded the least number of features: Gini < 0.05, Pearson > 0.95, and $R^2 \geq 0.95$.

Objective functions. We exploit three different aspects of recommendation as described in Section 2.2, through the measures: (a) NDCG for accuracy; (b) EPD for novelty; and (c) EILD for diversity. To compute the objective function values, we used the RankSys framework [Vargas and Castells, 2011]. We configured each objective function to evaluate a single quality aspect, *i.e.*, EPD and EILD do not take *accuracy* into account (as discussed in Section 2.2.1).

Evolutionary Algorithm Search. As stated before, we exploited NSGA-II as a black box. According to the previous definitions, we used the JMetal framework [Durillo and Nebro, 2011; Nebro et al., 2015], to model the problem as a WHF recommender. We exploited nine different configurations for tuning, varying the population size, crossover operator, mutation operator, and selection operation (Table B.2 details these tuning parameters).

Methods, configurations, and baselines. We exploited different configuration settings defined from three parameters: (a) the **Features Building** (**FB**) strategy: one of the three strategies described in Section 3.3.3: *STREAM* (Equation 3.1), *FWLS* (Equation 3.2), and *HR* (Equation 3.3); (b) the **Feature Selection** (**FS**) strategy: if the features selection defined in Section 3.3.4 is applied (*Sel*) or not (*All*); and (c) the **Decision Making** (**DM**) strategy: a global choice, *i.e.*, a solution chosen for all users, based on Equation 2.10 (*SUM*) and an individual choice, also based on Equation 2.10, but selecting each user individually (*IndSUM*). For the two DM strategies, we applied equal weights to all objectives. The *MO-Rank* prefix indicates a generalization of the MO method described in this chapter; each configuration is a combination of the three previously defined parameters, named by the pattern **MO-Rank-{FB}-{FS}-{DM}**. In particular, what this chapter brings innovation is the use of *meta-features* (FB = STREAM or FWLS), the feature selection strategy (FS = Sel), and individual decision making (DM = IndSUM). Thus, the *MO-Rank-HR-All-SUM* configuration is a baseline corresponding to the PEH method (using *PEH-mean* objective weights), proposed initially by Ribeiro et al. [2012, 2014], with only one difference, the use of NSGA-II instead of Strength Pareto Evolutionary Algorithm as an evolutionary search algorithm. We believe that MO methods based on Pareto frontiers should be more effective. To confirm this, we include a baseline method that reduces the MO problem into a SO problem, using the sum of the three objectives as an objective function and performing the optimization with the Particle Swarm Optimization (PSO) [Kennedy and Eberhart, 1995] because it is a simple, general, and effective meta-heuristic. For this method, we used the *SO-Rank* prefix, configuring it with the **FB** and **FS** parameters, so the name pattern is **SO-Rank-{FB}-{FS}**. We also evaluate the results of traditional WHF methods, represented by the name pattern **{FB}-{FS}**. Finally, we evaluate the *constituent algorithms* listed in Table 3.5 as baseline methods. Appendix B also details the learning methods applied in our experimental strategy.

3.4.2 Experimental analysis

In this section, we present three experimental analyses. Firstly, we evaluate the final recommendation results by ranking the various configurations of the methods under analysis in Section 3.4.2. Then, we performed a factor analysis to identify whether the parameters introduced in this thesis can influence the results of each method in Section 3.4.2. Finally, we evaluated the multi-objective search process to verify the influence of *meta-features* during the search for Pareto solutions in Section 3.4.2.

Overall recommendation analysis

We started our experimental analysis with a general evaluation of the final recommendations. We assess the quality of the recommendations according to the three ranking measures presented in Section 2.2.1 and the G_{RISK} measure presented in Section 2.2.2, applied to each of the ranking criteria individually.

Tables 3.7, 3.8, 3.9, and 3.10 summarize the recommendations' results. We used the *Fractional Ranking* computed for each evaluation measure to make the comparative analysis. The *Fractional Ranking* consists of the average of the *Ordered Ranking* when there are statistical ties (identified by the Confidence Interval with 95% confidence in this thesis). For instance, in the *fractional ranking* with values $1, 2.5, 2.5, 4$, the value 2.5 is obtained by $(2 + 3)/2$, representing a tie between the $2nd$ and $3rd$ in the equivalent ordered ranking $(1, 2, 3, 4)$. The sum of the fractional rankings defines the overall ranking. We rank all configurations and present only the results of the best configurations of each method and unique configurations for some specific comparisons. Here we present only the resulting fractional rankings. In Appendix C we present the mean values of the measures and their confidence intervals.

Table 3.7 presents the results for Amazon. We can observe a predominance of the MO-Rank method, with three configurations ranked best for all evaluation measures except NDCG. All these configurations used the global DM, but feature building and selection strategies vary. Considering the NDCG metric, the best result was the WHF method, a not very surprising result. On the other hand, a *constituent algorithm* obtained a good result. Biased-MF tied for first place in EILD and achieved a good performance in the other evaluation measures, giving it an honorable fourth place. The SO-Rank method did not get good results; there was always a configuration of the other methods better ranked than its best configuration. Among the SO-Rank configurations, the best configuration uses *meta-features*, the STREAM strategy, and the best three configurations use all features.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Rank-FWLS-Sel-SUM	7.0	**4.0**	**6.0**	**3.0**	**3.0**	**3.0**	**26.0**
1	MO-Rank-HR-All-SUM	7.0	**4.0**	**6.0**	**3.0**	**3.0**	**3.0**	**26.0**
1	MO-Rank-STREAM-All-SUM	7.0	**4.0**	**6.0**	**3.0**	**3.0**	**3.0**	**26.0**
4	Biased-MF	7.0	9.5	**6.0**	11.5	10.5	10.0	54.5
5	STREAM-All	**1.5**	13.0	13.5	7.0	13.5	13.0	61.5
7	HR-All	**1.5**	16.5	13.5	7.0	13.5	13.0	65.0
9	SO-Rank-STREAM-All	17.5	16.5	17.0	11.5	7.5	7.5	77.5
10	SO-Rank-FWLS-All	19.5	16.5	17.0	11.5	7.5	7.5	79.5
10	SO-Rank-HR-All	19.5	16.5	17.0	11.5	7.5	7.5	79.5
12	ALS	3.0	21.0	20.0	11.5	16.0	16.0	87.5

Table 3.7: Fractional rankings for **Amazon**. Measures values in Table C.1.

Table 3.8 presents the results for Bookcrossing. We can observe three MO-Rank configurations in the top three positions in the overall ranking. However, the HR strategy has taken first place alone due to outstanding performance for all evaluation measures. Once again, there was unanimity regarding the DM strategy and variation regarding the feature selection. The WHF method performed well in NDCG, emphasizing the FWLS strategy, which also ranked first for EPD, EILD, and G_{RISK}(NDCG). Despite a good performance for EPD and EILD, the *constituent algorithm* was outperformed by the WHF method in the overall ranking, while the SO-Rank method had the worst results one more time. Among the SO-Rank configurations, again, the best configuration uses *meta-features*, now with the FWLS strategy, but there was a variation between the feature selection strategies.

Table 3.9 presents the results for Jester. The WHF method using the HR strategy obtained a surprising result, achieving the best ranking for all evaluation measures alongside the MO-Rank method using the same HR strategy. Among the best MO-Rank configurations, we only observed the global DM strategy and a variation between the feature selection strategies. Another unexpected result was for WHF using FWLS strategy, which did not get the best outcome for NDCG, but got good results for the three

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Rank-HR-Sel-SUM	**3.0**	**8.0**	**8.0**	**2.0**	**3.0**	**3.0**	**27.0**
2	MO-Rank-HR-All-SUM	10.0	**8.0**	**8.0**	**2.0**	**3.0**	**3.0**	34.0
3	MO-Rank-STREAM-All-SUM	10.0	**8.0**	**8.0**	6.0	**3.0**	**3.0**	38.0
3	MO-Rank-FWLS-All-SUM	10.0	**8.0**	**8.0**	6.0	**3.0**	**3.0**	38.0
4	FWLS-All	**3.0**	**8.0**	**8.0**	**2.0**	9.5	13.0	43.5
6	HR-Sel	**3.0**	16.5	16.5	11.5	16.0	15.5	79.0
7	ALS	17.5	**8.0**	**8.0**	21.5	16.0	15.5	86.5
8	Biased-SVD	21.5	**8.0**	**8.0**	21.5	16.0	15.5	90.5
9	SO-Rank-FWLS-All	17.5	16.5	16.5	17.5	16.0	15.5	99.5
10	SO-Rank-HR-All	17.5	20.0	20.0	17.5	16.0	20.0	111.0
11	SO-Rank-STREAM-Sel	17.5	20.0	20.0	17.5	20.5	20.0	115.5

Table 3.8: Fractional rankings for **Bookcrossing**. Measures values in Table C.2.

G_{RISK} measures. Again, SO-Rank was outperformed by the best configurations of all other methods. Among the SO-Rank configurations, the best configuration uses the HR strategy, and again there was a variation between the feature selection strategies.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	HR-All	**3.0**	**4.0**	**4.0**	**3.5**	**3.5**	**3.5**	**21.5**
1	MO-Rank-HR-All-SUM	**3.0**	**4.0**	**4.0**	**3.5**	**3.5**	**3.5**	**21.5**
2	MO-Rank-STREAM-Sel-SUM	6.5	**4.0**	**4.0**	**3.5**	**3.5**	**3.5**	25.0
3	FWLS-All	10.0	10.0	10.0	**3.5**	**3.5**	**3.5**	40.5
4	Biased-SVD	10.0	10.0	10.0	8.0	8.0	8.0	54.0
4	UserKNN	10.0	10.0	10.0	8.0	8.0	8.0	54.0
5	MO-Rank-FWLS-All-SUM	13.5	13.5	13.5	8.0	8.0	8.0	64.5
8	SO-Rank-HR-All	15.0	15.5	16.0	11.0	11.0	11.0	79.5
11	SO-Rank-FWLS-All	18.0	18.0	18.0	11.0	11.0	11.0	87.0
13	SO-Rank-STREAM-Sel	19.5	19.5	20.0	14.0	14.0	14.0	101.0

Table 3.9: Fractional rankings for **Jester**. Measures values in Table C.3.

Table 3.10 presents the results for Movielens. We observed two configurations of the MO-Rank method dividing the first place, using HR and STREAM strategies, but with different results only for the EILD and G_{RISK}(NDCG) evaluation measures. MO-Rank configurations are followed by the WHF method, which ranked first for three evaluation measures. Moreover, SO-Rank was again outperformed by the best configurations of all other methods. Among the SO-Rank configurations, the best configuration uses *meta-features*with the FWLS strategy and varies the feature selection between the three configurations one more time.

These results show a great advantage for the MO-Rank method over multiple evaluation measures for all datasets. This is expected because this method simultaneously optimizes the three ranking measures. Although risk-sensitive is not explicitly optimized, MO-Rank has proven robust across all three G_{RISK} measures. WHF stands out in NDCG in most cases, and in some situations, it presents promising results for other measures, outperforming all other methods except MO-Rank (with a tie in the first place for Jester). On the other hand, although considering all ranking measures in its optimization process, SO-Rank was consistently outperformed by the other methods. Regarding the features building, features selection, and decision-making strategies, the only absolute was the last

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Rank-HR-Sel-SUM	**4.5**	**2.5**	6.0	**2.5**	**3.5**	**2.5**	**21.5**
1	MO-Rank-STREAM-Sel-SUM	**4.5**	**2.5**	**2.5**	6.0	**3.5**	**2.5**	**21.5**
2	MO-Rank-HR-All-SUM	**4.5**	7.5	10.0	**2.5**	**3.5**	**2.5**	30.5
3	MO-Rank-FWLS-All-SUM	10.5	7.5	**2.5**	6.0	**3.5**	**2.5**	32.5
4	FWLS-All	**4.5**	7.5	13.5	**2.5**	**3.5**	6.5	38.0
4	HR-All	**4.5**	7.5	13.5	**2.5**	**3.5**	6.5	38.0
8	ItemKNN	13.0	15.0	15.0	8.5	9.0	9.0	69.5
10	Biased-SVD	16.0	16.0	16.5	10.0	10.0	10.0	78.5
13	SO-Rank-FWLS-All	17.0	17.0	18.5	13.5	13.0	12.5	91.5
14	SO-Rank-HR-Sel	20.5	18.5	16.5	13.5	13.0	12.5	94.5
14	SO-Rank-STREAM-Sel	18.5	18.5	18.5	13.5	13.0	12.5	94.5

Table 3.10: Fractional rankings for **Movielens**. Measures values in Table C.4.

one, where the global choice of a solution always obtained the best results. In the next section, we evaluate these three parameters.

Factor analysis

Our second experimental analysis assesses whether the main parameters introduced in this chapter influence the recommendation results. We do this through the Analysis of Variance (ANOVA) test, with 95% confidence, applied to the following factors:

- **MF** (*meta-featured*): indicating whether the method uses *meta-features* (STREAM or FWLS) or whether the method does not use *meta-features* (HR).

- **Sel** (feature selection): indicating whether the method performs the feature selection described in Section 3.3.4 or whether it uses all features built.

- **DM** (decision making): indicating whether the choice of the best Pareto solution to make the recommendations was made globally (a choice for all users) or individually (a choice for each user).

We run several ANOVA tests evaluating each method (*i.e.*, WHF, SO-Rank, and MO-Rank) individually for each evaluation measure. To keep the restriction of the same number of individual results between the factors for the ANOVA test, we use the best configuration using HR and the best configuration using STREAM or FWLS, thus, considering only the best of the *meta-featured* strategy. Tables 3.11, 3.12, 3.13, and 3.14 summarize the ANOVA test results, where “Y” denotes factor influence on final results and “N” means non-influence. The WHF and SO-Rank methods do not have the decision-making task, so for them, the analysis were performed with the first two factors (MF and Sel) and their interactions. For the MO-Rank method, we evaluated the three factors and their interactions.

Table 3.11 presents the ANOVA test results for Amazon. We can observe that all factors and their interactions can influence the recommendation results for all three

G_{RISK} measures for all methods. On the other hand, there was some discrepancy regarding ranking measures between the different methods. For the WHF method, no factor influenced the recommendation results for the three ranking measures. For SO-Rank method, NDCG was influenced by both factors and their interaction, while the Sel factor influenced all ranking measures, and EILD was also influenced by the MF:Sel interaction. For the MO-Rank method, the factors MF, Sel, and MF:Sel influenced all ranking measures, while the DM factor and all interactions in which this factor was involved did not affect any ranking measure.

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
WHF						
MF	N	N	N	**Y**	**Y**	**Y**
Sel	N	N	N	**Y**	**Y**	**Y**
MF:Sel	N	N	N	**Y**	**Y**	**Y**
SO-Rank						
MF	**Y**	N	N	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:Sel	**Y**	N	**Y**	**Y**	**Y**	**Y**
MO-Rank						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**
Sel:DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel:DM	N	N	N	**Y**	**Y**	**Y**

Table 3.11: Factor analysis for **Amazon** recommendations.

Table 3.12 presents the ANOVA test results for Bookcrossing. We can observe a similar result to Amazon regarding the three G_{RISK} measures, with only two scenarios not influencing the results, Sel for G_{RISK}(NDCG) and MF:DM for G_{RISK}(EPD), both for the MO-Rank method. On the other hand, the results regarding ranking measures are quite different. Some factors influenced the WHF method: MF influencing EPD and EILD, Sel influencing NDCG, and the MF influencing NDCG and EPD. The SO-Rank method was less influenced, but MF influenced NDCG, and MF and Sel influenced EILD. The MO-Rank method was even less influenced: only MF influenced NDCG.

Table 3.13 presents the ANOVA test results for Jester. Once again, we observe all the three G_{RISK} measures being influenced by the factors, as well as for Amazon. However, we observe a very different pattern concerning the ranking measures. For the WHF method, all ranking measures were influenced only by the MF factor. On the other hand, the SO-Rank method was influenced by all factors for all ranking measures. The result for the MO-Rank method, in turn, was the same as for Amazon, where all ranking measures were influenced by factors that do not involve the DM factor.

Table 3.14 presents the ANOVA test results for Movielens. Again, all factors influenced all G_{RISK} measures. On the other hand, we have a different behavior concerning ranking measures. WHF and SO-Rank methods were heavily influenced,

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
WHF						
MF	N	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	N	N	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	N	**Y**	**Y**	**Y**
SO-Rank						
MF	**Y**	N	**Y**	**Y**	**Y**	**Y**
Sel	N	N	**Y**	**Y**	**Y**	**Y**
MF:Sel	N	N	N	**Y**	**Y**	**Y**
MO-Rank						
MF	**Y**	N	N	**Y**	**Y**	**Y**
Sel	N	N	N	N	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel	N	N	N	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	N	**Y**
Sel:DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel:DM	N	N	N	**Y**	**Y**	**Y**

Table 3.12: Factor analysis for **Bookcrossing** recommendations.

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
WHF						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	N	N	N	**Y**	**Y**	**Y**
MF:Sel	N	N	N	**Y**	**Y**	**Y**
SO-Rank						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MO-Rank						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**
Sel:DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel:DM	N	N	N	**Y**	**Y**	**Y**

Table 3.13: Factor analysis for **Jester** recommendations.

excluding two scenarios without influence. WHF was not influenced by the MF:Sel factor for EPD and EILD, while for SO-Rank, the MF factor did not influence EPD, and Sel did not influence EILD. Again, for MO-Rank, all factors in which DM participates did not influence any ranking measure. Additionally, EILD was not influenced by the MF factor.

These results show that G_{RISK} measures are significantly influenced by all factors, even though they are not optimized. On the other hand, besides being less influenced, ranking measures show many variation in results among different datasets. However, all factors influence the results of ranking measures in different scenarios, except DM.

Multi-Objective optimization analysis

Our last experimental analysis aims to assess the usefulness of *meta-features* in the multi-objective optimization search. An excellent way to compare multi-objective methods is to

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
WHF						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:Sel	**Y**	N	N	**Y**	**Y**	**Y**
SO-Rank						
MF	**Y**	N	**Y**	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	N	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MO-Rank						
MF	**Y**	**Y**	N	**Y**	**Y**	**Y**
Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**
Sel:DM	N	N	N	**Y**	**Y**	**Y**
MF:Sel:DM	N	N	N	**Y**	**Y**	**Y**

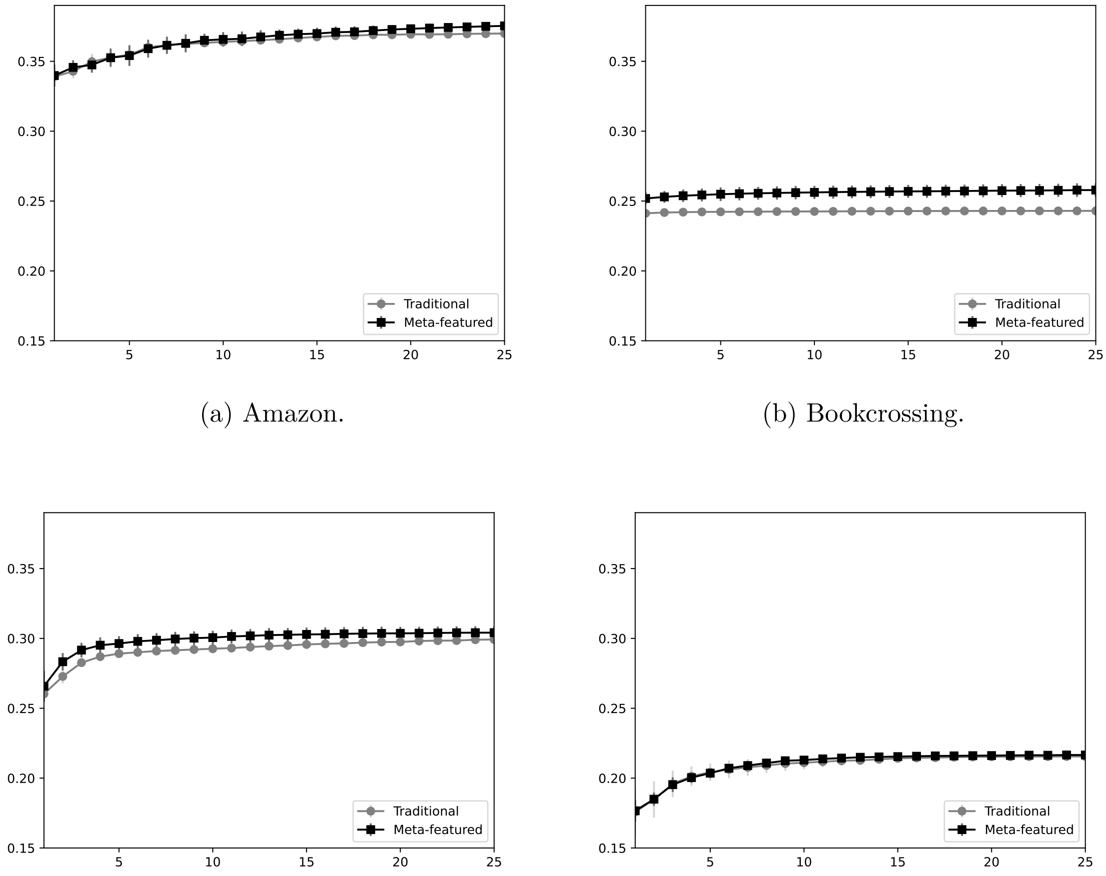
Table 3.14: Factor analysis for **Movielens** recommendations.

assess their ability to get good Pareto sets during the search task. A widely used quality indicator is the Hypervolume [Zitzler and Thiele, 1999], which measures the coverage and diversity of the Pareto set in the search space. The greater the hypervolume value, the better the Pareto set evaluated.

Figure 3.5 shows the hypervolume values for every 15 minutes throughout the multi-objective search considering only the best *traditional* (HR) configuration and the best *meta-featured* (STREAM or FWLS) configuration of the MO-Rank method. All graphics are on the same scale. The Confidence Interval with 95% of confidence was computed and is displayed as error bars.

A more significant difference between the methods can be observed for Bookcrossing, with considerable advantage for the *meta-featured* configuration. Despite being very close to each other, for Amazon and Jester, one can observe a slight advantage for the *meta-featured* configuration in some contexts. For Amazon, there is a trend of similar results at the beginning of the search, a slight advantage for *meta-featured* at the end, and an opposite behavior for Jester, with *meta-featured* doing better at the beginning and closer results at the end. On the other hand, for Movielens, the results are practically identical throughout the entire search.

These results show potential for better multi-objective search results when using *meta-features* as additional information. At worst, the result tends to be the same as the traditional method.



(a) Amazon.

(b) Bookcrossing.

(c) Jester.

(d) Movielens.

Figure 3.5: Hypervolume evaluation for traditional and *meta-featured* MO. The X-axis represents the slots of 15 minutes elapsed throughout the MO search and the Y-axis represents the hypervolume value.

3.4.3 Discussion

In the previous section, three analysis of experimental results were presented, which allowed us to observe the behavior of the methods and the parameters introduced in this chapter and will enable us to answer our first research question: ***RQ1**: "Does explicitly incorporating meta-features contribute to improving the results of multi-objective recommendation?"*.

Before discussing the results, it is essential to highlight that the baseline methods are very competitive. CF algorithms can surprise users with unexpected recommendations, leveraging novelty and diversity. Additionally, sparsity problems are reduced in our experiments, selecting users who have a minimum number of items to be recommended, making it even more competitive for all ranking measures. WHF methods combine results from various algorithms and thus have the potential to generate even better results. In addition to being hybrid, SO-Rank optimizes the three objectives through an objective function that reduces the MO problem to such a problem. Finally, the PEH method (represented by the MO-Rank-HR-All-SUM configuration) differs because it does not explore the *meta-features*, features selection, and individual decision-making.

Through the factor analysis, we could observe that in several scenarios the introduced parameters influence the results, especially for the G_{RISK} measures. In the analysis of the MO-Rank method, we could observe that *meta-features* can contribute to the multi-objective search result, getting better or similar results to the quality of the Pareto set obtained. However, by analyzing the rankings of the final recommendations, we observed very similar results regarding the use of *meta-features*, where the *meta-featured* strategies (using STREAM or FWLS) showed promising results but not enough to consistently surpass the results of the traditional method (using HR). The PEH baseline method has always been first or second in the overall ranking.

On the other hand, the analysis of the SO-Rank and WHF methods showed that the *meta-features* were useful for the final recommendation results for the overall ranking. In just one dataset, Jester, the *meta-featured* configurations have surpassed by traditional configurations. An essential difference between SO-Rank and WHF methods and the MO-Rank method is the need for the Decision Making task, choosing a solution from the Pareto set to carry out the recommendations. However, the decision process that we exploit does not consider the *meta-features* and is the only factor that did not influence the results in all scenarios evaluated by the ANOVA tests. Additionally, Jester is the least conventional dataset, with only 100 items and many ratings, providing the lowest sparsity of all datasets.

Therefore, our experiments empirically show that *meta-features* can be helpful to contribute to improving the results for multi-objective recommendations, thus allowing a positive answer to our ***RQ1*** question. However, concerning the MO-Rank method, two tasks are essential for the generation of recommendations: (a) *search*, aiming to obtain a better Pareto set, which was benefited in some scenarios by the explicit use of *meta-features*; and (b) *decision*, which aims to choose a suitable solution from this set to carry out the recommendations, which did not explicitly use *meta-features* in our experiments. These two tasks motivate two research areas, MO optimization and Multi-Criteria Decision Making. As described by Wismans et al. [2011], the decision-making process is an open problem in MO optimization. In this thesis, we focus on the search process, but it is observed that it is still necessary to act in the decision process in future works to value the eventual gains obtained in the search using *meta-features*.

Finally, an intriguing result caught our attention: risk-sensitive measures were heavily influenced by the parameters introduced in this chapter, even though they were not explicitly optimized. This result motivated us to explore such measures during the optimization process to improve the results of multi-objective recommendations.

3.5 Concluding remarks

This chapter presents a *meta-featured* multi-objective recommender method to answer our first research question and improve our knowledge of this particular recommendation method. In the next chapter, we introduce a new multi-objective recommender method based on the explicit use of risk-sensitive measures to answer our second research question and continue contributing to the advancement of knowledge in this area.

Chapter 4

Risk-sensitive MOF

In this chapter, we exploit risk-sensitive measures as resources to be explicitly considered within the scope of MOF to answer our second research question: ***RQ2****: "Does explicitly incorporating risk-sensitive measures contribute to improving the results of multi-objective recommendation?"*. We intend to keep the ranking measures as the primary objectives for optimization while ensuring better performance in terms of risk sensitivity for those measures.

To accomplish this, we firstly contextualize risk-sensitiveness in the scope of RSs in Section 4.1. Then, we define a new multiple-objective strategy based on the *Meta-featured* MOF described in the previous chapter. We propose a new dominance relation and adapt the G_{RISK} computation and the NSGA-II algorithm based on the specificities of the new measures to be considered in the optimization process within the scope of MOF recommendation in Section 4.2. Results of empirical experiments allow a positive answer to our ***RQ2*** and provide some interesting observations about the behavior of the methods in different experimentation scenarios, as presented in Section 4.3. Finally, Section 4.4 concludes the chapter with some considerations.

4.1 Risk-sensitiveness and Recommender Systems

Recommender algorithms aim to maximize the quality of the recommendations for most users using the optimization of the **average value** of some quality measure, usually using machine learning strategies, as described in Chapter 2. However, due to considerable differences among users' profiles and interests, such an optimization process may incur a large effectiveness variability across the range of users and recommendations, despite the good general "average" results. When the algorithm tries to optimize an average measure for many users, there is a *risk* (*non-negligible likelihood*) of providing harmful recommendations to many of them.

Indeed, having many satisfied users is fundamental for RSs success, whereas a few dissatisfied ones may negatively impact the overall system performance. Users receiving harmful recommendations may stop using the RS due to distrust and spread the "bad word". A related phenomenon happens when a given user receives good suggestions from the RS most of the time but sometimes receives such disappointing recommendations that they become suspicious about the actual effectiveness of the system. Knijnenburg et al. [2012] provide evidence that users tend to remember the failures of a Recommender System more quickly than the many successful results they have received from it.

The consequences of "optimizing by the average" have been addressed in the Information Retrieval literature related to Search Engines (see Section 2.2.2). They have paid attention to models that consider the *risk* of producing poor effectiveness for specific *queries* [Wang et al., 2012], a line of research known in that community as *risk-sensitive* solutions. Risk-sensitive measures such as G_{RISK} [Dinçer et al., 2016] have been developed to maximize some overall average measures (such as Mean Average Precision or NDCG) while avoiding the risk of incurring poor results for a few but essential *queries* [Dinçer et al., 2014; Sousa et al., 2016; Wang et al., 2012].

However, risk-sensitiveness has been historically considered in the search realm only in accuracy-related measures. Observe that, when optimizing multiple conflicting objectives, the effects on average optimization can be even more significant, resulting in worse results on specific aspects for some users.

Specifically, in the RSs realm, we observe that Hybrid Filtering [Burke, 2002, 2007] also has a connection with risk-sensitive strategies. Both try to avoid bad results by exploiting multiple systems and data to reduce the variability of results over several users. Here, we hypothesize that hybrid filtering models can further reduce the risk of bad results for specific users when coupled with risk-sensitive strategies, with the ability to further improve outcomes in multi-objective recommendations.

Some works in RS literature use the terms **risk** and **risk-aware** with distinct connotations. For instance, Bouneffouf et al. [2013] and Bouneffouf [2016] present improvements in *Context-Aware Recommender System*, which consider the *risk* for the current user's context, *i.e.*, an item can be good or bad depending on the user's context. Liang et al. [2015] mentioned the *risk* of recommending relevant applications in mobile systems with a high potential to open the doors to security and privacy intrusions. Jeunen [2019] considers the risk of suggesting unsafe mobile applications by applying permission warnings and malware detection to help users stay safe and avoid the risky use of smartphones. Ge et al. [2020] consider the risk concerning the "users' attitude", which in turn is related to the "purchase risk" due to the product quality (items with good ratings and reviews are *less risky*). They assume that this risk depends on the user's tolerance based on their past behavior. Recently Manotumruksa et al. [2019] proposed a framework for Context-Aware Venue Recommendation and only evaluated the final recommendation results through risk sensitivity.

There have been efforts to reduce poor RSs results for users with specific profiles and specific situations, such as [Khusro et al., 2016]: (a) the absence of data for new users and items (*cold-start*); (b) the scarcity of data concerning the proportion of the number of users and items (*sparsity*); (c) users who have lower similarity or agreement with other users (*grey-sheep users*); and (d) ensuring that all users are treated equally by an algorithm or model (*fairness*). In this thesis, we deal with potentially harmful results by assessing the effectiveness variability in the optimization process across the range of different solutions or models in the context of Hybrid Filtering without considering any specific user profile or situation. In this sense, our proposed method is *profile-agnostic* and, therefore, it is different from but complementary to the previous efforts.

To the best of our knowledge, risk-sensitiveness with this particular connotation and goal has not been explicitly exploited in the RSs literature. In the next section, we begin to tackle this challenge by proposing a Risk-sensitive MOF strategy.

4.2 The Risk-sensitive MOF strategy

Our main goal is to improve the general recommendation results while simultaneously reducing the risk of harmful recommendations regarding different quality aspects. We define the recommendation task as a MO optimization problem that embeds the improvement of a risk-sensitive measure applied to distinct recommendation objectives in the optimization process. Section 4.2.1 presents the definition of the MO problem. Section 4.2.2 defines a general strategy for the risk-sensitive measure computation in the MO optimization. Finally, Section 4.2.3 presents the optimization strategy proposed.

4.2.1 Problem Definition

Consider a set of *quality functions* $\mathcal{Q} = \{q_1,\ q_2,\ ...,\ q_m\}$, where each function q_i measures a distinct recommendation quality aspect (*e.g.*, accuracy, novelty, and diversity). Let *RiskS* be a function that measures the risk-sensitiveness concerning one quality aspect. We formally define the risk-sensitive recommendation problem as:

$$\begin{aligned} \underset{x}{\arg\max} \quad & \mathbf{O}(x) = (q_1(x),\ ...,\ q_m(x), \mathit{RiskS}(q_1(x)),\ ...,\ \mathit{RiskS}(q_m(x))), \\ \text{s.t.} \quad & x = (x_1, x_2, ..., x_n)\ \in\ \mathcal{X}, \end{aligned} \tag{4.1}$$

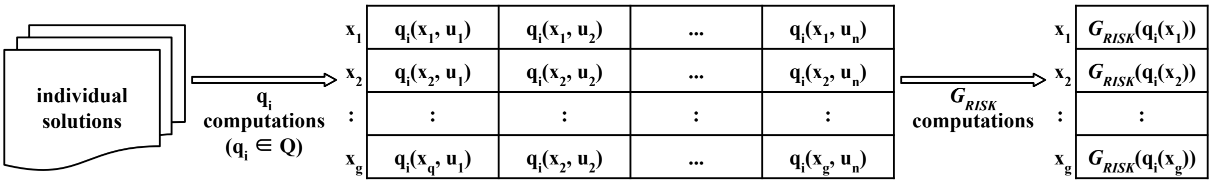
where: $x \in \mathcal{X}$ is a viable solution; $\mathcal{X}$ is the optimization parameter domain; $\mathbf{O} \in \Omega$ is the objective vector; $\Omega = O(\mathcal{X})$ is the objective space; and $\mathit{RiskS}(q_i(x))$ evaluates the risk-sensitiveness for solution x regarding the quality function q_i for all available users.

Currently, there is no convex function available for risk-sensitive optimization to the best of our knowledge. Consequently, convex optimization methods, *e.g.*, Matrix Factorization or Deep Neural-based RSs, do not apply to solve the proposed problem. Thus, we propose an evolutionary strategy that aims to find the best solutions that meet all the defined quality aspects and improve their risk sensitiveness.

4.2.2 Computing risk-sensitiveness for MO optimization

To solve Equation 4.1 through an optimization strategy, a mechanism for the *RiskS* computation is necessary. This work explores G_{RISK} as a risk-sensitive measure representing *RiskS*, defined in Section 2.2.2 for the Search Engines context. Next, in addition to presenting the computation strategy for the MOF and WHF recommendations, we also offer an adaptation of the mathematical formulations to the context of RSs.

A WHF recommender computes a score $\hat{s}$ for a user-item pair according to Equation 2.8. Thus, it uses a set of scores considering all candidate items for all users to generate ranked lists. These ranked lists are evaluated according to the quality aspects in $\mathcal{Q}$, which are used to compute the G_{RISK} measure. Figure 4.1 generalizes the G_{RISK} computation for one quality aspect q_i, where distinct solutions in $\mathcal{X} = \{x_1, x_2, ..., x_g\}$ generate the ranking for each user depending on the quality aspect q_i. For instance, $q_i(x, u)$ denotes the quality aspect q_i, for the user u, applying the solution x. Note that the figure describes only one matrix, generalizing for any quality aspect $q_i \in \mathcal{Q}$.



	u_1	u_2	...	u_n
x_1	$q_i(x_1, u_1)$	$q_i(x_1, u_2)$	...	$q_i(x_1, u_n)$
x_2	$q_i(x_2, u_1)$	$q_i(x_2, u_2)$	...	$q_i(x_2, u_n)$
:	:	:	:	:
x_g	$q_i(x_q, u_1)$	$q_i(x_2, u_2)$	...	$q_i(x_g, u_n)$

x_1	$G_{RISK}(q_i(x_1))$
x_2	$G_{RISK}(q_i(x_2))$
:	:
x_g	$G_{RISK}(q_i(x_g))$

Figure 4.1: G_{RISK} computation for the evolutionary process.

A critical issue of the matrix in Figure 4.1 is the variability in the quality of ranked lists for distinct users through distinct solutions x_j, with each solution representing a ranking model. Considering many users, distinct solutions may considerably vary over the users' ranked lists. In this scenario, selecting the best solutions considering only the average values can penalize some users while favoring others to achieve higher average values. We apply a more intelligent solutions selection process to avoid harmful recommendations for some users by assessing the variability of distinct users and solutions composing the matrix. Therefore, we select the best solution by evaluating how sensitive to risk x_j is compared to several other solutions, using the $G_{RISK}(q_i(x_j))$ function. In essence, $G_{RISK}(q_i(x_j))$ follows the definition of Dinçer et al. [2016] (see Section 2.2.2), here adapted to the RSs context.

To compute $G_{RISK}(q_i(x_j))$ we start with the function Z_{RISK}, adapting Equation 2.6 to:

$$Z_{RISK}(x_j) = \left[\sum_{u \in U_+} z(x_j, u) + (1 + \alpha) \sum_{u \in U_-} z(x_j, u) \right], \tag{4.2}$$

where: $z(x_j, u) = \frac{q_i(x_j, u) - e(x_j, u)}{\sqrt{e(x_j, u)}}$; $e(x_j, u) = q_i(x_j) \times \frac{T_u}{N}$; and $q_i(x_j, u)$ is the effectiveness of a solution x_j for a user u regarding the quality aspect q_i. Let $q_i(x_j) = \sum_{y=1}^{|U|} q_i(x_j, u_y)$ be the expected performance of solution x_j for all users regarding quality aspect q_i and $T_u = \sum_{j=1}^{|X|} q_i(x_j, u)$ be the sum of the effectiveness of user u when varying in all available solutions. Both U_+ and U_- are sets of positive and negative $z_{x,u}$, respectively, and $N = \sum_{j=1}^{|X|} \sum_{u=1}^{|U|} q_i(j, u)$. Parameter α defines the weight of degradation – distinct values for the parameter may significantly impact the risk-sensitive evaluation of the method.

Then we define G_{RISK}, adapting Equation 2.7 to:

$$G_{RISK}(q_i(x_j)) = \sqrt{q_i(x_j)/|U| \times \Phi(Z_{RISK}(x_j)/|U|)}, \tag{4.3}$$

where: Φ is the cumulative distribution function of the Standard Normal Distribution.

Note that the variability of one solution is assessed against the other solutions in the matrix. G_{RISK} explores the use of many distinct solutions by improving the evaluation regarding the average, the variance, and the shape of the score distribution. As a result, G_{RISK} can be effective in the selection process over the solutions in the search space, improving the resulting Pareto set concerning the risk-sensitiveness of the solutions obtained throughout the evolutionary search. The following section describes how the solutions change over the generations, improving the analysis and searching for the best ones.

4.2.3 A Pareto-based optimization strategy

This section presents a new Pareto-based strategy for the previously defined risk-sensitive multi-objective recommendation problem. For this, we also introduce a new dominance relation, called *Risk-sensitive dominance*, that explicitly takes into account the risk-sensitiveness in this context and is defined as:

Definition 2. ***(Risk-sensitive dominance).*** *One solution x_1 risk-sensitively dominates another solution x_2 ($x_1 \succ_{RiskS} x_2$) with regard to a set of quality functions $\mathcal{Q} = \{q_1, q_2, ..., q_m\}$ (with $m = |\mathcal{Q}|$), if and only if:*

- $RiskS(q_i(x_1)) \geq RiskS(q_i(x_2)),\ \forall\, i \in [1, m]$;
- $RiskS(q_j(x_1)) > RiskS(q_j(x_2)),\ \exists j \in [1, m]$.

There are several evolutionary algorithms for MO optimization, as discussed in Section 3.2. We follow the same choice above, adapting NSGA-II to incorporate risk-sensitiveness. NSGA-II is based on Genetic Algorithms [Srinivas and Patnaik, 1994] and explores the solution space by breeding better solutions over several generations. Considering the general NSGA-II strategy, we have implemented our extensions reflecting the risk-sensitive dominance proposal. Algorithm 1 describes this strategy.

Algorithm 1 Multi-objective method based on NSGA-II [Deb et al., 2002]

The m

```
    Input: Stopping criteria: $S$; Features: $\mathcal{F}$; Number of solutions: $NumSolutions$
    Output: Pareto Solutions List: $ParetoSet$
 1: $Population \leftarrow$ InitializePopulation($NumSolutions$)
 2: $ParetoSet \leftarrow \emptyset$
 3: EvaluateSolutions($Population \cup ParetoSet$, $\mathcal{F}$)
 4: while not $S$ do
 5:     $Parents \leftarrow$ Selection($Population$)
 6:     $NewSolutions \leftarrow$ Crossover($Parents$)
 7:     $NewSolutions \leftarrow$ Mutation($NewSolutions$)
 8:     $Solutions \leftarrow Population \cup NewSolutions$
 9:     EvaluateSolutions($Solutions \cup ParetoSet$, $\mathcal{F}$)
10:     $Frontiers \leftarrow$ DefineFrontiers($Solutions$, $ParetoSet$, $\mathcal{F}$)
11:     $Population \leftarrow$ UpdatePopulation($Frontiers$, $NumSolutions$)
12:     $ParetoSet \leftarrow$ getNonDominated($\mathcal{D}$, $ParetoSet \cup Frontiers[1]$)
13: end while
14: return $ParetoSet$
```

In this process, the algorithm receives as input the stopping criteria (S), usually defined as many generations or an execution timeout, the distinct hybrid features used for recommendations ($\mathcal{F}$), and the maximal number of solutions in each generation (*NumSolutions*). The algorithm begins by setting a random initial population (line 1). The *ParetoSet* variable stores the non-dominated solutions found throughout the evolutionary search and is used to compute the risk-sensitive measures. Before generating new populations, the initial population is evaluated (line 3). The evaluation of each solution consists of the computation of the quality functions and their associated G_{RISK}. This process combines the WHF and G_{RISK} computation, which is an innovation of this work, detailed in Section 4.2.2. To compose the matrix used for the G_{RISK} computation, we define the set of solutions by the union between the population of the current generation and the set of non-dominated solutions originated over all previous generations. This strategy allows a co-evolutionary search over the search space, avoiding an static comparison throughout the evolutionary process. The G_{RISK} measures for the non-dominated solutions are updated to make future assessments more accurate and fair. Then, the new generation loop begins.

The algorithm uses the population of solutions to perform the genetic operators, such as selection (line 5), crossover (line 6), and mutation (line 7); detailed in [Srinivas and Patnaik, 1994]. The algorithm joins the new solutions with the current population (line 8) and evaluates their simulated recommendations (line 9). Next, the algorithm builds the dominance *Frontiers* vector (line 10). *Frontiers* is a vector of sets of solutions. The first *frontier* (*Frontiers[1]*) is composed of non-dominated solutions, the second *frontier* (*Frontiers[2]*) is composed of solutions dominated only by one other solution, the third one is composed of solutions dominated only by two different solutions, and so on.

The population for the next generation is defined with solutions from the *Frontiers* sets (line 11), starting from the first *frontier* and sequentially to the other sets until obtaining the number of solutions (*NumSolutions*). The *Crowding Distance* sorts the solutions of each frontier to accommodate the solutions to the population size. Finishing the current generation, the *ParetoSet* is updated by evaluating the dominance relation between all its solutions and the non-dominated solutions from the actual *Population*, *i.e.*, *Frontiers*[1] (line 12). The algorithm returns the *ParetoSet* obtained.

For the decision-making process, we apply the same strategy described in Section 3.2, adapting to consider risk-sensitiveness:

$$\underset{x \in \mathit{ParetoSet}}{\arg\max} \sum_{i=1}^{m} \mathit{RiskS}(q_i(x)). \tag{4.4}$$

4.3 Experimental results

This section presents the experiments carried out to answer the research question ***RQ2*** and increase our knowledge of the risk-sensitiveness issue in the scope of *Meta-featured* MOF systems. We analyzed the results of the multi-objective strategy presented above, comparing different possible configurations from the proposed and baseline methods.

Firstly, Section 4.3.1, describes the primary resources and setup to conduct the experiments. Section 4.3.2 presents three experimental analyses. Finally, in Section 4.3.3, some discussions about the observed results.

4.3.1 Experimental setup

For the experiments in this chapter, we follow the same experimental strategy and many of the resources used in Chapter 3, described in Section 3.4.1 and detailed in Appendix B. The differences are described below.

Objective functions. In addition to the three ranking measures used in the previous chapter, *i.e.*, NDCG, EPD, and EILD, we also use the G_{RISK} applied to each of them in the risk-sensitive MOF method. For the G_{RISK} computation, we implemented Equation 4.3 using $\alpha = 5$. We observe that this value is also used in [Dinçer et al., 2016; Sousa et al., 2016], and, even though it supports our claims, we intend to explore other values in future work.

Evolutionary Algorithm Search. In addition to the resources for the evolutionary algorithm of the JMetal framework [Durillo and Nebro, 2011; Nebro et al., 2015] used in the previous chapter, we adapted the JMetal classes to accommodate the definitions described in Section 4.2.

Methods, configurations, and baselines. As baseline methods, we selected the best configurations, one with HR strategy and one *meta-featured*, for each WHF and MOF methods exploited in the previous chapter. We have also included the two best *constituent algorithms*. We exploited the same parameters previously defined for the new risk-sensitive MOF method, including two new DM strategies: a global choice using risk-sensitiveness, Equation 4.4 (*Risk*), and an individual option using risk-sensitiveness for each user, also based on Equation 4.4 (*IndRisk*). The *MO-Risk* prefix indicates a generalization of the MO method described in this chapter, and each configuration is a combination of the three parameters, named by the pattern **MO-Risk-{FB}-{FS}-{DM}**. As in the previous chapter, we also explored the reduction of the MO problem to an SO problem, using the **SO-Risk-{FB}-{FS}** name pattern for the SO method that considers risk-sensitiveness in the optimization process. To reduce the experimentation time for *MO/SO-Risk*, we explored only the best pair of **{FB}-{FS}** configuration obtained for the MO/SO-Rank methods using HR strategy and the best *meta-featured* strategy (STREAM or FWLS) in the experimental results of the previous chapter.

4.3.2 Experimental analysis

In this section, we present three experimental analyses. Firstly, in Section 3.4.2, we evaluate the final recommendation results by the same fractional ranking strategy used in the previous chapter. Then, we performed a factor analysis to identify whether the *meta-features* and decision-making parameters can influence the results of each risk sensitiveness method in Section 3.4.2. Finally, we performed a quantitative evaluation of several characteristics related to risk-sensitive measures in Section 3.4.2.

Overall recommendation analysis

Once again, we start by doing a general ranking analysis of the final recommendations, following the same strategy applied in Section 3.4.2. Tables 4.1, 4.2, 4.3, and 4.4 summarize the resulting fractional rankings of the best configurations of each method and unique configurations for some specific comparisons. Again, in Appendix C we present the mean values of the measures and their confidence intervals.

Table 4.1 presents the results for Amazon. We can observe four configurations tying in the first ranking position. Among them, three are from the MO-Risk method, alternating configurations with and without *meta-features* and DM using or not G_{RISK}. The other one of the best configurations is MO-Rank using the FWLS strategy. However, these configurations are in the first position for all evaluation measures except NDCG. For this measure, the best configurations are from the WHF method. The SO-Risk method obtained the worst results and even surpassed the SO-Rank method. In addition, their HR and STREAM configurations were tied.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Risk-HR-All-SUM	8.0	**5.0**	**5.5**	**3.5**	**3.5**	**3.5**	**29.0**
1	MO-Rank-FWLS-Sel-SUM	8.0	**5.0**	**5.5**	**3.5**	**3.5**	**3.5**	**29.0**
1	MO-Risk-HR-All-Risk	8.0	**5.0**	**5.5**	**3.5**	**3.5**	**3.5**	**29.0**
1	MO-Risk-FWLS-Sel-SUM	8.0	**5.0**	**5.5**	**3.5**	**3.5**	**3.5**	**29.0**
2	MO-Rank-HR-All-SUM	13.5	**5.0**	**5.5**	**3.5**	**3.5**	**3.5**	34.5
4	Biased-MF	8.0	10.5	**5.5**	10.0	10.0	10.0	54.0
5	HR-All	**1.5**	13.5	12.0	7.5	10.0	10.0	54.5
5	STREAM-All	**1.5**	13.5	12.0	7.5	10.0	10.0	54.5
7	SO-Rank-STREAM-All	15.0	13.5	14.5	10.0	10.0	10.0	73.0
8	ALS	3.0	18.0	17.0	10.0	14.0	14.0	76.0
9	SO-Rank-HR-All	17.0	13.5	14.5	13.0	10.0	10.0	78.0
10	SO-Risk-HR-All	17.0	16.5	17.0	13.0	10.0	10.0	83.5
10	SO-Risk-STREAM-All	17.0	16.5	17.0	13.0	10.0	10.0	83.5

Table 4.1: Fractional rankings for **Amazon**. Measures values in Table C.5.

Table 4.2 presents the results for Bookcrossing. We now observe less competitiveness, with only two methods in the first place. However, we have a configuration of each method, a MO-Risk and another MO-Rank, both using HR strategy, showing that the *meta-features* did not contribute to better recommendation results for these methods in this scenario. On the other hand, these two configurations achieved the best performance in all evaluation measures. It is noteworthy that they were the absolute best configurations for G_{RISK} evaluation measures and that there was great competition with other methods for ranking criteria. Once again, the SO methods had the worst results compared to the other methods. However, when comparing the SO-Risk and SO-Rank methods, each way's *meta-featured* configurations outperformed its corresponding HR configurations, showing the usefulness of the *meta-features* for the SO methods in this scenario.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Rank-HR-Sel-SUM	**3.5**	**7.0**	**7.0**	**2.0**	**2.0**	**2.0**	**23.5**
1	MO-Risk-HR-Sel-Risk	**3.5**	**7.0**	**7.0**	**2.0**	**2.0**	**2.0**	**23.5**
4	MO-Rank-STREAM-All-SUM	9.5	**7.0**	**7.0**	6.0	6.0	6.0	41.5
4	MO-Risk-STREAM-All-Risk	9.5	**7.0**	**7.0**	6.0	6.0	6.0	41.5
5	FWLS-All	**3.5**	**7.0**	**7.0**	6.0	11.0	10.0	44.5
8	HR-Sel	**3.5**	16.0	14.5	10.5	14.0	13.0	71.5
9	ALS	17.5	**7.0**	**7.0**	17.5	11.0	13.0	73.0
9	Biased-SVD	17.5	**7.0**	**7.0**	17.5	11.0	13.0	73.0
10	SO-Risk-FWLS-All	14.5	16.0	14.5	14.5	16.0	16.0	91.5
11	SO-Rank-FWLS-All	14.5	16.0	17.0	14.5	16.0	16.0	94.0
11	SO-Risk-HR-All	14.5	16.0	17.0	14.5	16.0	16.0	94.0
12	SO-Rank-HR-All	14.5	16.0	17.0	14.5	18.0	18.0	98.0

Table 4.2: Fractional rankings for **Bookcrossing**. Measures values in Table C.6.

Table 4.3 presents the results for Jester. The WHF method using the HR strategy showed the best result, being isolated in the first place. It performed very well in all evaluation measures, confirming first place for its advantage concerning the accuracy measured by the NDCG. Once again, the MO methods show competitiveness between MO-Risk and MO-Rank, with one configuration in the second place, both without *meta-features*. The SO methods again had the worst results, with SO-Risk outperformed by SO-Rank and the *meta-featured* configurations surpassed by their corresponding configurations using HR.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	HR-All	**1.0**	**4.0**	**6.0**	**4.5**	**4.5**	**4.5**	**24.5**
2	MO-Rank-HR-All-SUM	4.0	**4.0**	**6.0**	**4.5**	**4.5**	**4.5**	27.5
2	MO-Risk-HR-All-SUM	4.0	**4.0**	**6.0**	**4.5**	**4.5**	**4.5**	27.5
3	MO-Rank-STREAM-Sel-SUM	10.5	**4.0**	**6.0**	**4.5**	**4.5**	**4.5**	34.0
4	MO-Risk-STREAM-Sel-SUM	10.5	11.0	**6.0**	**4.5**	**4.5**	**4.5**	41.0
5	FWLS-All	10.5	11.0	13.0	**4.5**	**4.5**	**4.5**	48.0
6	Biased-SVD	10.5	11.0	13.0	9.0	9.5	9.0	62.0
8	UserKNN	10.5	11.0	13.0	10.0	9.5	10.0	64.0
10	SO-Rank-HR-All	15.0	15.0	15.0	12.5	12.5	12.5	82.5
11	SO-Rank-FWLS-All	17.0	16.5	16.0	12.5	12.5	12.5	87.0
11	SO-Risk-HR-All	16.0	16.5	17.0	12.5	12.5	12.5	87.0
12	SO-Risk-FWLS-All	18.0	18.0	18.0	12.5	12.5	12.5	91.5

Table 4.3: Fractional rankings for **Jester**. Measures values in Table C.7.

Table 4.4 presents the results for Movielens. In this scenario, we observe an advantage for the MO-Risk method using *meta-features*, which was isolated in the first place. The result is very close to MO-Rank, with a configuration in second place, differing only for NDCG and EILD evaluation measures. Concerning SO methods, we again observe an advantage for SO-Rank over SO-Risk, but with their *meta-featured* configurations outperforming the HR configurations for this scenario.

#	Method	Ranking measures			G_{RISK} measures			Overall
		NDCG	EPD	EILD	NDCG	EPD	EILD	
1	MO-Risk-STREAM-Sel-SUM	7.5	**5.5**	**2.0**	**3.5**	**3.0**	**3.5**	**25.0**
2	MO-Rank-HR-Sel-SUM	**2.5**	**5.5**	9.5	**3.5**	**3.0**	**3.5**	27.5
3	MO-Risk-HR-Sel-SUM	7.5	**5.5**	6.0	**3.5**	**3.0**	**3.5**	29.0
3	MO-Rank-STREAM-Sel-SUM	7.5	**5.5**	6.0	**3.5**	**3.0**	**3.5**	29.0
6	FWLS-All	**2.5**	11.5	11.5	**3.5**	7.0	7.5	43.5
6	HR-All	**2.5**	11.5	11.5	**3.5**	7.0	7.5	43.5
7	ItemKNN	13.0	13.0	13.0	9.0	9.0	9.0	66.0
10	Biased-SVD	14.0	14.0	14.0	10.0	10.0	10.0	72.0
11	SO-Rank-FWLS-All	15.0	15.0	17.0	12.0	12.0	12.5	83.5
12	SO-Risk-FWLS-All	16.0	16.0	17.0	12.0	12.0	12.5	85.5
13	SO-Rank-HR-Sel	17.0	17.5	17.0	12.0	12.0	12.5	88.0
14	SO-Risk-HR-Sel	18.0	17.5	15.0	14.0	14.0	12.5	91.0

Table 4.4: Fractional rankings for **Movielens**. Measures values in Table C.8.

These results show that the MO-Risk method is competitive in many scenarios, tying with MO-Rank for the first three datasets and being isolated in first place for Movielens. There is an alternation regarding the usefulness of *meta-features* for the MO-Risk and SO-Risk methods. In some scenarios, the *meta-featured* configuration tie with the corresponding HR configuration. In others, it loses, and considering the Movielens dataset, MO-Risk benefited from the *meta-features*. Regarding the DM strategy, we again observe the predominance of global choice using SUM, with few appearances of the Risk choice strategy and no occurrence of individual preferences among the best configurations.

Factor analysis

This section presents a factor analysis similar to that performed in Section 3.4.2 through the Analysis of Variance (ANOVA) test, with 95% confidence, applied only for the MO-Risk and SO-Risk methods and only for **MF** and **DM** factors. Tables 4.5, 4.6, 4.7, and 4.8 summarize the ANOVA test results.

We can observe a typical result for all datasets, with all factors influencing the G_{RISK} measures. On the other hand, there is variation in the results from one dataset to another for ranking measures that we highlight below.

For Amazon and Jester, in Tables 4.5 and 4.7, we observe that *meta-features* are relevant for results influencing SO-Risk in all ranking measures and MO-Risk in NDCG and EPD. The decision-making factors did not influence the results. For Bookcrossing, in Table 4.6, we observe only one influence scenario on the results: MO-Risk with the MF factor influencing NDCG. For Movielens, in Table 4.8, we observe the same result for SO-Risk, with MF factor influencing all ranking measures. However, unlike previous datasets, the DM factor influenced NDCG and EILD results.

Similar to the previous chapter, these results show how all factors are essential for G_{RISK} measures in all datasets. But now, with the explicit use of G_{RISK} in the

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
SO-Risk						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MO-Risk						
MF	**Y**	**Y**	N	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**

Table 4.5: Factor analysis for **Amazon** recommendations.

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
SO-Risk						
MF	N	N	N	**Y**	**Y**	**Y**
MO-Risk						
MF	**Y**	N	N	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**

Table 4.6: Factor analysis for **Bookcrossing** recommendations.

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
SO-Risk						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MO-Risk						
MF	**Y**	**Y**	N	**Y**	**Y**	**Y**
DM	N	N	N	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**

Table 4.7: Factor analysis for **Jester** recommendations.

Factor	Ranking measures			G_{RISK} measures		
	NDCG	EPD	EILD	NDCG	EPD	EILD
SO-Risk						
MF	**Y**	**Y**	**Y**	**Y**	**Y**	**Y**
MO-Risk						
MF	**Y**	N	**Y**	**Y**	**Y**	**Y**
DM	**Y**	N	**Y**	**Y**	**Y**	**Y**
MF:DM	N	N	N	**Y**	**Y**	**Y**

Table 4.8: Factor analysis for **Movialens** recommendations.

optimization process. On the other hand, the DM factor showed a minor influence on the results for ranking measures, with an impact on NDCG and EILD in Movielens. This dataset was also the only one for which MO-Risk was isolated in the first place in the final recommendation results. Movielens has similar characteristics to Amazon, with approximately the same number of ratings and sparsity, but differs by application domain: movies and books, respectively (see Table 3.6). Again we observe varied results for the different datasets when considering ranking measures, with the factors influencing the results in different scenarios in line with the varied final recommendation results.

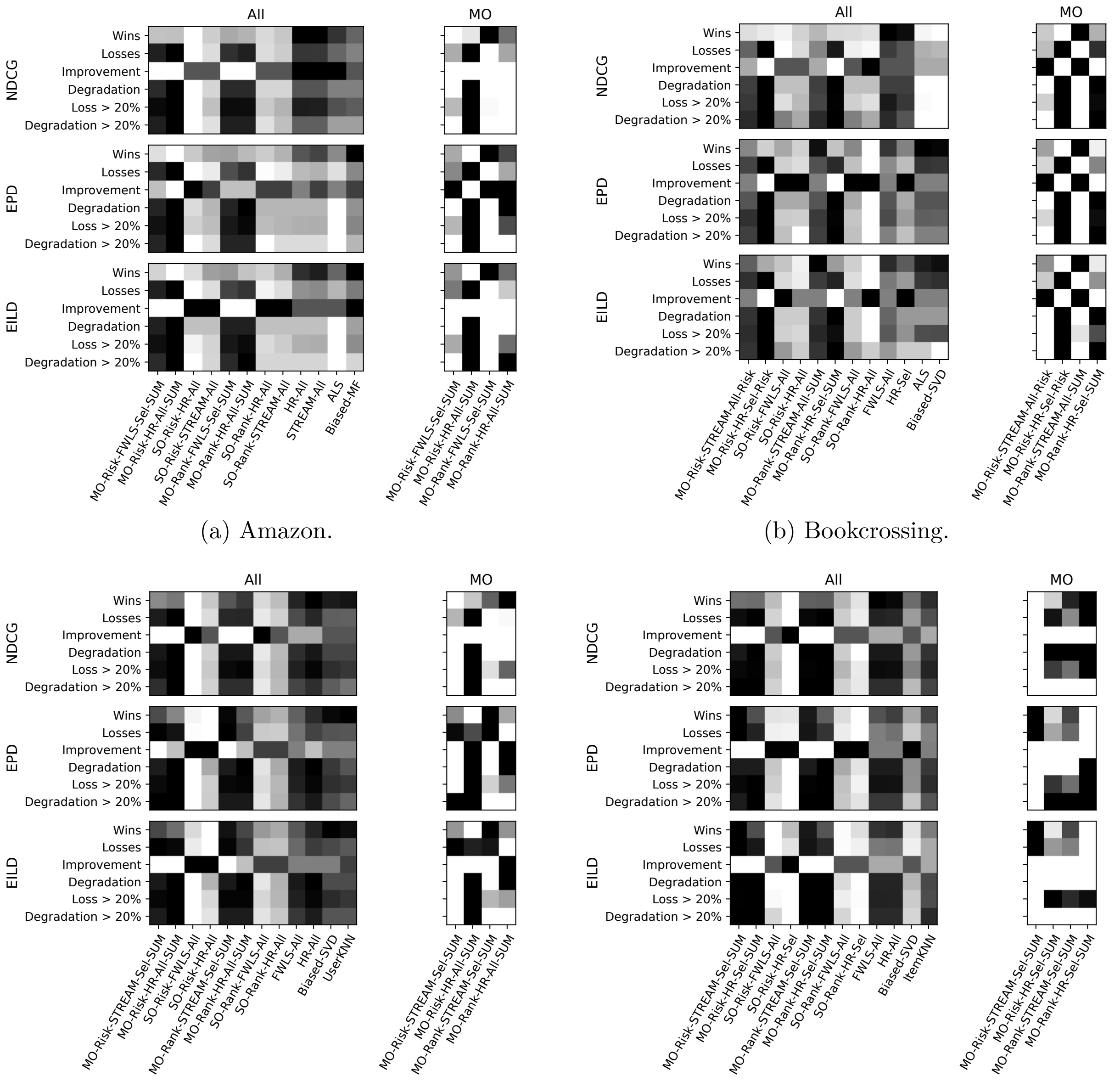
Quantitative risk analysis

This section presents a quantitative analysis of several characteristics related to risk-sensitive measures that we describe below. As presented in Section 2.2.2, risk-sensitive measures aim to assess the effectiveness of an algorithm in terms of **gain** and **degradation** concerning other models/algorithms. Therefore, we compare the recommendations from one configuration against all others for all users considering each evaluation measure individually. The quantitative characteristics we evaluate in this section are: (a) **Wins**: the number of wins; (b) **Losses**: the number of losses; (c) **Improvement**: the average of the percentage gain; (d) **Degradation**: the average of the percentage loss; (e) **Loss > 20%**: the number of losses greater than 20%; (f) **Degradation > 20%**: the average of the percentage loss greater than 20%.

Figure 4.2 presents heatmaps of the quantitative values obtained to evaluate these characteristics. Since the measured characteristics have distinct natures and ranges of values, they were normalized to the range of [0, 1]. The black color represents the best results, grayscales represent intermediate values, and the white color represents the worst results. For each dataset, we present two heatmaps. One considers the best configurations of all methods. The other considers only the MO methods. In both, we consider a configuration using *meta-features* and another using HR to evaluate the effectiveness of explicitly using the *meta-features*.

Figure 4.2 presents common patterns between the four datasets when comparing all methods. We observed a trend of the MO methods to reduce loss and degradation at the expense of the number of wins and improvement. On the other hand, SO methods have the opposite behavior, expanding wins and improvement at the expense of loss and degradation. Both present similar results for all evaluation measures. It is essential to highlight that for the G_{RISK} measure, we use $\alpha = 5$, consequently applying a higher penalty for degradation. Therefore, this kind of behavior is to be expected, remembering that our goal is to reduce terrible recommendations trying to keep the average result, even if, in return, there is some reduction of excellent recommendations. The MO methods outperform the others, with WHF methods and *constituent algorithms* achieving an intermediate result and certain variations between different datasets. Additionally, in heatmaps with all methods, we could not more accurately differentiate the results of the MO method, which motivated us to present a second heatmap comparing only this method. Below we present some specific patterns observed for each dataset.

Figure 4.2a presents the results for Amazon. The WHF method and *constituent algorithms* show a trend towards better results for NDCG than the other evaluation measures. Considering the heatmap with only MO methods, we observe that MO-Risk using HR outperformed others, followed by MO-Rank using HR. Thus, for this scenario, the MO methods proved to be effective in reducing loss and extensive degradation at the



(a) Amazon.

(b) Bookcrossing.

(c) Jester.

(d) Movielens.

Figure 4.2: Quantitative evaluation for risk-sensitive related characteristics.

expense of the number of gains and improvement, with an advantage for the risk-based method but without using *meta-features*.

Figure 4.2b presents the results for Bookcrossing. Similar to Amazon, for Bookcrossing, the WHF methods showed better results for NDCG compared to other evaluation measures, but with a minor difference for EPD and EILD. On the other hand, the *constituent algorithms* did not get good results for NDCG, but they had good results for EPD and EILD. Although WHF shows consistent results for all characteristics, it still loses to MO methods in terms of loss and degradation characteristics. Again, considering the heatmap with only MO methods, we observe the effectiveness of MO methods without *meta-features*. Both MO-Risk and MO-Rank using HR showed promising results for all characteristics except degradation, with a slight advantage for MO-Risk. On the other hand, *meta-features* seem to have contributed to the degradation reduction.

Figure 4.2c presents the results for Jester. In this scenario, the WHF methods and the *constituent algorithms* achieved consistent results for all characteristics on all evaluation measures, with a slight disadvantage for the MO methods concerning loss and degradation. The results considering only MO methods are similar to Amazon, with an advantage for MO-Risk followed by MO-Rank, both without *meta-features.*

Figure 4.2d presents the results for Movielens. Once again, the WHF methods showed consistent results regarding the risk for all characteristics across all evaluation measures. While also consistent, *constituent algorithms* lose to WHF. Considering the heatmap with only MO methods, we observed behavior that differs from other datasets. MO-Rank outperforms MO-Risk, with an advantage for the HR configuration.

These results show that optimizing risk sensitivity concerning the different desirable characteristics of this theme is complex and resembles MO optimization problems, with attributes that conflict. This conflict is easily noticeable when looking at the results of the MO and SO methods. On the other hand, the WHF methods proved to be effective in obtaining a good performance on all characteristics simultaneously, tending towards NDCG in some datasets, but with similar results for all evaluation measures in other datasets. However, considering the primary goal of reducing loss and degradation, the MO-Risk method showed a more remarkable aptitude to achieve better results on all evaluation measures, fulfilling its role. A piece of relevant information is that configurations that use *meta-features* get better results for improvement, while HR configurations get better results for loss and degradation. Thus, *meta-features* are not helpful in this regard.

4.3.3 Discussion

The previous section presents analysis of experimental results allowing us to observe the behavior of the methods and the parameters introduced in this chapter and enabling us to answer our second research question: ***RQ2:*** *"Does explicitly incorporating risk-sensitive measures contribute to improving the results of multi-objective recommendation?".*

The baseline methods also demonstrated particular effectiveness for risk optimization, especially WHF methods and *constituent algorithms*, in the quantitative evaluation of various characteristics involved with risk-sensitiveness. We highlight the ranking result for the final recommendations, with the isolated first place in the overall ranking for the Jester dataset.

However, once again, the MO methods proved to be more effective within the general context, with the MO-Risk and MO-Rank methods disputing the first places in the final recommendations, but with an advantage in three out of the four datasets for the quantitative analyses of the risk characteristics related to loss and degradation for the three evaluation measures. We highlight the ranking result for the final recommendations, with the isolated first place for the MO-Risk-STREAM-Sel-SUM configuration in the overall ranking for the Movielens dataset.

Through the factor analysis, we could observe that the two introduced parameters evaluated, MF and DM, maintains a significant influence in the recommendation results for the G_{RISK} measures, now explicitly considering risk-sensitive in the optimization process. This influence is even less significant for ranking measures, with some variation between different datasets. We highlight that the DM factor, which in the previous chapter did not influence any dataset, now affects one out of the four datasets (Movielens).

Looking at the final recommendation results, for two datasets (Bookcrossing and Jester), the *meta-featured* MO-Risk configurations did not outperform their counterparts using HR. In contrast, for Movielens, the isolated first place uses *meta-features*, and for Amazon, they were all tied. Confronting these results with the quantitative characteristics of risk, considering loss and degradation as the most important, we can observe the choice of the best MO configuration for those that explicitly use the risk-sensitive measures for three datasets (Amazon, Bookcrossing, and Jester). The exception is Movielens, the best place for the final recommendation ranking is MO-Risk, but the best placed on quantitative characteristics is MO-Rank. Although MO-Risk achieved a good result for large (> 20) loss and degradation for NDCG and EPD.

The dominant DM process was again the global choice for ranking metrics (SUM configurations, following Equation 2.10) with only two appearances of the global choice by risk (Risk configuration, following Equation 4.4). These results reinforce the importance of further studies on the DM process, which we will leave for future work.

Therefore, our experiments empirically show that explicitly exploring risk-sensitive measures can be helpful to contribute to improving the results for multi-objective recommendations, reducing loss and degradation, thus allowing a positive answer to our ***RQ2*** question. However, the use of *meta-features* has not proved to be very useful in improving the results of the final recommendations and reducing loss and degradation. On the other hand, we can observe *meta-features* contributing to increase wins and improvement without deteriorating the final recommendation results. Thus, *meta-features* cannot be ignored in future work, exploring new measures and classes of *meta-features* even in the decision-making process.

4.4 Concluding remarks

This chapter presents a risk-sensitive multi-objective recommender method to answer our second research question and improve the knowledge in this particular recommendation problem. We have considered that all objectives are equally important to users. In the next chapter, we introduce a new multi-objective recommender method based on the explicit use of users' preferences on the optimized objectives to answer our third research question and continue contributing to the advancement of knowledge in this area.

Chapter 5

Preference-based MOF

In this chapter, we explore the users' preferences regarding the optimized objectives in the scope of MOF to answer our third research question: ***RQ3****: "Does explicitly incorporating individual preferences of users concerning the optimized objectives contribute to improving the results of multi-objective recommendation?".* We intend to keep the original optimized objectives as the primary focus while obtaining results that prioritize the optimized objectives in proportion to each user's importance.

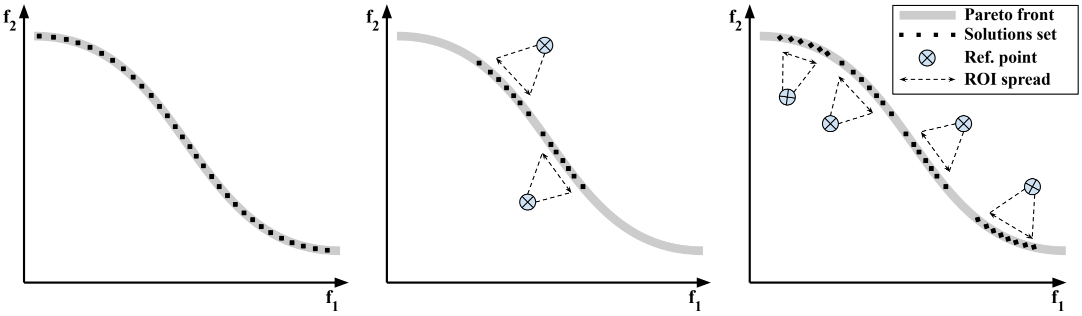
We firstly discuss MO preference-based methods in Section 5.1. Then we define a new MOF method that explores users' preferences concerning the objective functions during both tasks, *search* and *decision-making*, in Section 5.2. In particular, the search for optimal solutions maximizes objective functions guided by a newly defined dominance relation, the **Extreme Pareto Dominance**, that conjointly exploits the notions of *Pareto Dominance*, *Extreme Dominance*, and *Statistical Significance Tests.* The decision is made by choosing the optimal solution from the set found during the search that most closely matches the users' preferences. Results of empirical experiments allow a positive answer to our ***RQ3*** and provide some interesting observations about the behavior of the methods in different experimentation scenarios, as presented in Section 5.3. Finally, Section 5.4 concludes the chapter with some considerations.

5.1 Multi-objective preference-based methods

A critical issue of the Pareto-based strategy, especially when the objectives are conflicting, is the large size of the Pareto set, making it difficult to choose a solution in the decision-making phrase that best fits the users' needs [Bechikh et al., 2015]. Some methods guide the optimization phase in searching for Pareto solutions through preference information regarding the objectives to improve the results and simplify this task.

The most commonly used strategy is defining *reference points* (or *vectors*) in the objective space to define a *Region of Interest* (ROI), focusing on the solutions of the

Pareto set closest to these points [Bechikh et al., 2015; Fonseca and Fleming, 1993; Li et al., 2015; Wang et al., 2019]. Figure 5.1 shows simulations of the Pareto set obtained in the optimization phase (maximization problem) for traditional methods and preference-based methods for different numbers of reference points.



(a) Traditional. (b) Two reference points. (c) Many reference points.

Figure 5.1: Different visualizations for the Solutions set produced by MO methods regarding the number of reference points (adapted from Fonseca and Fleming [1993]; Wang et al. [2017]).

Traditional methods try to produce a more diverse set of solutions, covering the entire Pareto front (Figure 5.1(a)). On the other hand, MO preference-based methods try to obtain solutions closer to users' preferences, producing results more concentrated in the regions of interest. In the scope of MO optimization, the decision-makers are usually represented by a few users or reference points (Figure 5.1(b)). However, in this chapter, we are interested in representing the decision-makers as the complete set of users of the RSs. In other words, every user is, in fact, a decision-maker. Since the number of users is very high (Figure 5.1(c)), this brings challenges in terms of scalability and effectiveness. It seems intuitive to think that, in this case, it is better to explore the entire objective space (Figure 5.1(a)) and to explore the users' preferences only during the decision-making phase. However, we claim that explicitly exploring users' preferences in the optimization phase has the potential to generate results closer to the users' needs in RSs. Experimental results corroborate our hypotheses.

Another MO preference-based method, *objective weighting*, usually aggregates objectives [Bechikh et al., 2015; Li et al., 2015; Wang et al., 2017], reducing the problem by decreasing the number of objectives. One challenge is the difficulty for decision-makers (in our case, the end-users) to define the weight values, as they are usually subjective [Bechikh et al., 2015]. However, this task should not be done explicitly by the end-users. Instead, it should be done implicitly, as most users would not be able to do it properly. Indeed, this task can use preference data related to consumed or evaluated items. We consider that this is the best strategy to represent users' preferences when searching for optimal solutions to produce recommendations instead of reference points. The main problem is that these strategies, similarly to re-ranking methods, cannot adequately deal with conflicts among the objectives. Again, we deal with this issue by proposing a Pareto-based optimization strategy that preserves the individuality of the objectives even when using weights.

Alternative definitions of the dominance relation constitute another strategy to guide the **optimization phase**. Works such as [Ben Said et al., 2010; Filatovas et al., 2017; Hu et al., 2017; Molina et al., 2009] propose new dominance relations that guide the search to concentrate the solutions around reference points, achieving good results when the number of objectives is no more than three [Wang et al., 2019]. For instance, Cvetkovic and Parmee [2002] proposed the Weighted-Dominance relation, which consists of assessing the relative importance of each objective and a parameter defining a minimum requirement for the dominance relation. However, their dominance relation is very coarse-grained. They evaluate only the number of improvements a solution obtains over another, not differentiating fine-grained (high or low) improvements [Bechikh et al., 2015]. Consequently, they lose the precision in comparing alternative solutions through their dominance relation. In this chapter we define a new weight-based dominance relation that does not have this limitation.

Only a handful of MOF systems can be considered preference-based, as defined in our work. Jannach et al. [2015] and Kapoor et al. [2015], for instance, developed methods that promote the adaptation of items' ranking scores according to the *user's tendencies* for popularity and novelty, respectively. However, these works consider only one dimension regarding the users' tendencies, reducing the optimization phase to a Single-Objective problem. We found in the literature only two preference-based works that optimize multiple objectives without reducing the optimization phase to a single objective in the scope of MOF, described below.

Jugovac et al. [2017] proposed a strategy that considers *user's tendencies* for different quality dimensions while maintaining high accuracy. Their strategy considers statistical measures on the optimization objectives, such as mean and standard deviation, to represent the users' preferences regarding the objectives. The method is a re-ranking strategy: (a) firstly, reordering the *top-K* most relevant items for the accuracy objective; and (b) secondly, minimizing the absolute difference between "*other objectives*" (such as diversity and popularity) computed for the users profile and the ones calculated for the reordered items. Therefore, they consider users' preferences only regarding the "*other objectives*" in the second stage.

Finally, Liu et al. [2019] proposed a preference-based method for RSs in the service composition domain. The optimization objective is a set of Quality of Services (QoS), and the user's preferences are used as weights computed for each objective. They apply deep neural networks in two stages. In the first one, they learn the users' preferences concerning the optimization objectives. In contrast, they use another deep neural network in the second stage to produce recommendations considering the learned weights. Their solution is not a general-purpose RS method, restricting to the domain of service composition recommendation. It is not Pareto-based, resulting only in one single solution model.

To the best of our knowledge, we present the first MO preference-based method for general-purpose RSs, using evolutionary algorithms and Pareto-based dominance (with statistical tests), avoiding the drawbacks of re-ranking and region of interest strategies while keeping the advantages of Pareto-based and preference-based methods. In the next section we propose our preference-based MOF strategy.

5.2 The preference-based MOF strategy

We defined our preference-based strategy similar to the *meta-featured* methods presented in the two previous chapters. Therefore, we use the evolutionary algorithm NSGA-II and define specific strategies to extract the users' preferences and for the optimization tasks, maintaining the modeling of the MO optimization problem as a WHF recommender. The following subsections present these strategies.

5.2.1 Users' preferences extraction

For a preference-based method, it is first necessary to define the computation of the specific users' interests related to the objective functions through their historical data.

Jannach et al. [2015] defined the *User Bias* (UB) to determine the user's preference to like or dislike popular items based on the items rated by the user considering both the overall popularity and the ratings provided by the user for those items. They modeled the decision as a Single-Objective Optimization Problem, minimizing the difference between the recommended *List Bias* (LB) and the computed UB. The LB is computed similarly to the UB but using predicted ratings. Kapoor et al. [2015] developed a logistic regression model to predict the user's preferences for novelty in the music recommendation domain. However, in this chapter, we still need to process the user's preferences for each of the multiple objectives individually. We want to extract them from historical data and not predict them from a learned model.

Therefore, the relevance of the objective o for the user u is expressed by $\mu_{u,o}$:

$$\mu_{u,o} = \frac{\mathcal{W}_{u,o}}{\sum_o \mathcal{W}_{u,o}}, \tag{5.1}$$

where: $\mathcal{W}_{u,o}$ is the *Normalized Expected Utility* for the objective function o for the target user u; $\sum_o \mathcal{W}_{u,o}$ is the sum of the *Normalized Expected Utility* for all objective functions; and, consequently, $\sum_o \mu_{u,o} = 1$.

The *Normalized Expected Utility* is defined based on the Gunawardana and Shani [2009] adaptation and generalization of the *Expected Utility* of a ranked list of items defined by Breese et al. [1998]:

$$\mathcal{W}_{u,o} = \frac{w_{u,o}}{w_{u,o}^{max}}, \tag{5.2}$$

where: $w_{u,o}$ is the *Expected Utility* for the objective o and user u; $w_{u,o}^{max}$ is the maximum achievable $w_{u,o}$, *i.e.*, considering all items at the top of the list; and:

$$w_{u,o} = \sum_{j=1}^{N} \frac{\Theta_{o,i_j}}{2^{(j-1)/(\alpha-1)}}, \tag{5.3}$$

where: i_j is the j-th item from the ranked list of the N items rated by the user; and Θ_{o,i_j} is a measure related to the objective o computed for the item i_j.

In this chapter, we ranked the items by the inverse order of the rating values and for the Θ measures we used: (a) for **accuracy**: the rating value; (b) for **novelty**: the expected popularity of an item, defined as $1 - p(seen|i)$ [Vargas and Castells, 2011], where $p(seen|i)$ is the ratio between the number of users who rated item i and the total number of users; and (c) for **diversity**: the average of the items Cosine distances, defined in Equation 2.3.

5.2.2 Multi-Objective Search

As discussed in Section 5.1, many preference-based works use the definition of *reference points*, and some works define new concepts for the dominance relation to guiding the MO search. However, these works traditionally attempt to reduce the complexity of optimizing many objectives in *Many-Objective* problems (*i.e.*, greater than or equal to four objectives) by using the decision-maker's preferences to guide the search for solutions. Moreover, business managers usually represent the decision-makers. Since the number of users is much greater than the number of business managers, this represents a challenge. To the best of our knowledge, we present the first preference-based work in the scope of Recommender Systems, paying attention to their specificities.

Therefore, considering that the users' preferences are represented as weights for each objective function, we will be inspired by the concept of *Extreme Dominance* [Ehrgott, 1997]. It considers the relevance of each objective function adapted for a maximization problem, as shown in Definition 3.

Definition 3. *(Extreme Dominance). A solution $x_1 \in \mathcal{X}$ dominates another solution $x_2 \in \mathcal{X}$ ($x_1 \succ_\lambda x_2$) if, given a weighted vector λ such that $\sum_i \lambda_i = 1$:*

- $\sum_i \lambda_i * O_i(x_1) > \sum_i \lambda_i * O_i(x_2),\ \forall\ i \in [1, m]$.

Extreme Dominance is, conceptually, an approach that reduces the MO problem into a single-objective problem by maximizing the weighted sum of the objective functions. This chapter intends to guide the search for optimal solutions by considering the weights of objectives while maintaining their individuality rather than using weights to reduce the problem.

Moreover, to obtain a more robust and precise dominance relation, the concept of statistical significance tests can be applied to improve the best choice. This strategy was successfully and initially applied in the scope of Learning to Rank, as described in [Sousa et al., 2019]. When using statistical tests, we are more strict in assigning the difference between two solutions, providing an improved selection when compared. In the case of a Pareto set, this process ensures that the non-dominated solutions are better than others over a Null Hypotheses comparison. Thus, the inequality of the two solutions is considered only when there is statistical significance.

Therefore, we defined the concept of ***Individualized Extreme Dominance***, which combines the *Pareto Dominance* (Definition 1), *Extreme Dominance* (Definition 3), and the *Statistical Significance Tests* to determine the dominance relation introduced in Definition 4.

Definition 4. ***(Individualized Extreme Dominance).*** *A solution $x_1 \in \mathcal{X}$ dominates another solution $x_2 \in \mathcal{X}$ ($x_1 \succ_{\mu,\alpha} x_2$) if, given a weighted vector μ for each user, mapping the user's preferences about each objective, such that $\sum_i \mu_i = 1$, for every user:*

- $\mu_{u,o_i} * O_{i,u}(x_1) \leq_\alpha \mu_{u,o_i} * O_{i,u}(x_2),\ \forall\ i \in [1, m]$, *and*
- $\mu_{u,o_j} * O_{j,u}(x_1) <_\alpha \mu_{u,o_j} * O_{j,u}(x_2),\ \exists\ j \in [1, m]$,

where: μ_{u,o_i} is the objective function weight for user u and objective i; $O_{i,u}$ is the result for the objective i and user u; $\mu_{u,i} * O_{i,u}(x)$ results in a vector containing all users, in which each element corresponds to the result of the objective weighted by the user's preferences; and $\leq_\alpha$ is an operation performed in two stages: (a) applies a statistical test to assess whether the two vectors are different with confidence level α; and (b) if different, compares the mean values of two given vectors to assess the dominance relation.

Within the scope of Multi-Objective optimization, two Euclidean spaces are traditionally defined. The ***Solution Space*** maps the representation of solutions according to the modeling of the problem, *e.g.*, the weights w_i for each feature in $\mathcal{F}$ representing the solutions (Equation 2.8). The ***Objective Space*** maps the values of the objective functions, *i.e.*, the $O(x)$ values obtained by applying the solutions (Equation 2.9). In this chapter, we define a new Euclidean space, the ***Objective Relevance Space***, which maps the estimated users' preferences $\mu_{u,o}$ (Equation 5.1) or even the relevance obtained by applying the solutions computed from $O(x)$ values similar to the computation of $\mu_{u,o}$.

To better understand the new dominance relation, we present Algorithm 2, based on the Pareto dominance evaluation available in JMetal framework [Durillo and Nebro, 2011; Nebro et al., 2015], described below.

Algorithm 2 Individualized Extreme Dominance relation evaluation

```
    Input: Solutions: $s1$, $s2$; objective function weights: $\mu$; significance level: $\alpha$
    Output: Dominance relation
 1: s1Dominates $\leftarrow$ False; s2Dominates $\leftarrow$ False
 2: for each objective function o do
 3:     Initialize $values1$ and $values2$ with empty vectors
 4:     for each user $u$ $in$ $\mathcal{U}$ do
 5:         $values1$.add($\mu$[u] * $s1$[o][u])
 6:         $values2$.add($\mu$[u] * $s2$[o][u])
 7:     end for
 8:     if statisticalTest(values1, values2, $\alpha$) then
 9:         if mean(values1) == mean(values2) then flag = 0
10:         else if mean(values1) < mean(values2) then flag = -1
11:         else flag = 1
12:         end if
13:     else flag = 0
14:     end if
15:     s1Dominates = flag == -1 ? True : s1Dominates
16:     s2Dominates = flag == 1 ? True : s2Dominates
17: end for
18: if s1Dominates == s2Dominates then return 0
19: else if s1Dominates then return -1
20: else return 1
21: end if
```

Algorithm 2 receives the results from the evaluated solutions, $s1$ and $s2$, the objective function weights μ, and the significance level α. The output is an integer value indicating the dominance relation. Firstly, it initializes two boolean variables with the *false* value. Then, it updates the boolean variables for each objective function o (lines 2 to 17). Two vectors are filled with the objective function values for o from $s1$ and $s2$ weighted by μ for each user u (lines 3 to 7). Next, the statistical test evaluates if the two vectors are statistically different (line 8). We used the Wilcoxon test, as used by Sousa et al. [2019]. If the vectors are considered different, their mean values are compared to define the flag value indicating if non-dominance is identified or if one solution dominates the other concerning the objective function o. Finishing the individual objective function evaluation, if the flag value is different from 0 (zero), one of the boolean variables is updated to the actual value (lines 15 and 16). Finally, after the individual evaluation of the objective functions, the boolean variables are used to define the dominance relation result (lines 18 to 21).

Following the choices of the two previous chapters, we chose the NSGA-II as the evolutionary algorithm for performing the MO search.

5.2.3 Decision Making

The decision-making process is responsible for selecting one solution from the Pareto set to produce the best recommendation to a target user u considering their preferences for the objective functions.

Ribeiro et al. [2012, 2014] deal with this issue by maximizing the sum of objectives weighted by their respective importance, as defined in Equation 2.10. However, to better approximate of the weights obtained for each objective with the users' preferences, we define new selection criteria by minimizing the distance between the Pareto solution recommendation results and the user's preferences, as shown in Equation 5.4.

$$\underset{x \in \mathcal{X}}{\arg\min} \quad Dist(\mu_u, \mu_{R(x)}), \tag{5.4}$$

where: μ_u are the preferences of user u mapped in the *Objective Relevance Space*; $\mu_{R(x)}$ is the achieved objective functions values obtained by solution x, also mapped in the *Objective Relevance Space*; and *Dist* is a distance measure (we used Euclidean distance in our experiments).

5.3 Experimental results

This section presents the experiments carried out to answer the research question ***RQ3*** and increase our knowledge of the preference-based MO issue in the scope of *Meta-featured* MOF systems. We analyzed the results of the MO strategy presented above, comparing different possible configurations from the proposed and baseline methods.

Firstly, Section 5.3.1 describes the primary resources and setup to conduct the experiments. Section 5.3.2 presents a comparative analysis. Finally, in Section 5.3.3, some discussions about the observed results.

5.3.1 Experimental setup

For the experiments in this chapter, we follow the same experimental strategy and many of the resources used in Chapter 3, described in Section 3.4.1 and detailed in Appendix B. The differences are described below.

Objective functions. In addition to the three ranking measures, *i.e.*, NDCG, EPD, and EILD, we also want to reduce the Euclidean distance between the results and users' preferences in the *Objective Relevance Space* defined in Section 5.2.2.

Evolutionary Algorithm Search. In addition to the resources for the evolutionary algorithm of the JMetal framework [Durillo and Nebro, 2011; Nebro et al., 2015], we adapted the JMetal classes to accommodate the definitions described in Section 5.2.2.

Methods, configurations, and baselines. As baseline methods, we selected the best configurations of the previous methods: (a) the two best *constituent algorithms*; (b) the best HR configuration of the WHF, MO-Rank, and SO-Rank methods; and (c) the best *meta-featured* (STREAM or FWLS) configuration of the WHF, MO-Rank, and SO-Rank methods. For the preference-based method, we follow a similar nomenclature used before: **MO-Rank-{FB}-{FS}-{DM}** and **SO-Rank-{FB}-{FS}**, replacing the prefixes *MO-Rank* and *SO-Rank* with: (a) **PrefMO** and **PrefMO-St**, for the MO configurations applying Definition 4 with and without statistical relevance, respectively; and (b) **PrefSO**, for the SO configurations. Finally, we only include the new decision-making strategy defined in Section 5.2.3 (DM = IndDIST) for **PrefMO** and **PrefMO-St**. We also apply the previous decision-making strategies (DM = SUM and DM = IndSUM) for the preference-based method.

It is essential to highlight that we did not optimize G_{RISK} in this chapter (as we did in Chapter 4). We use this measure only for additional evaluation criteria (as done in Chapter 3). We consider that the users' preferences when considering risk sensitivity demand additional attention, and we will leave it for future works.

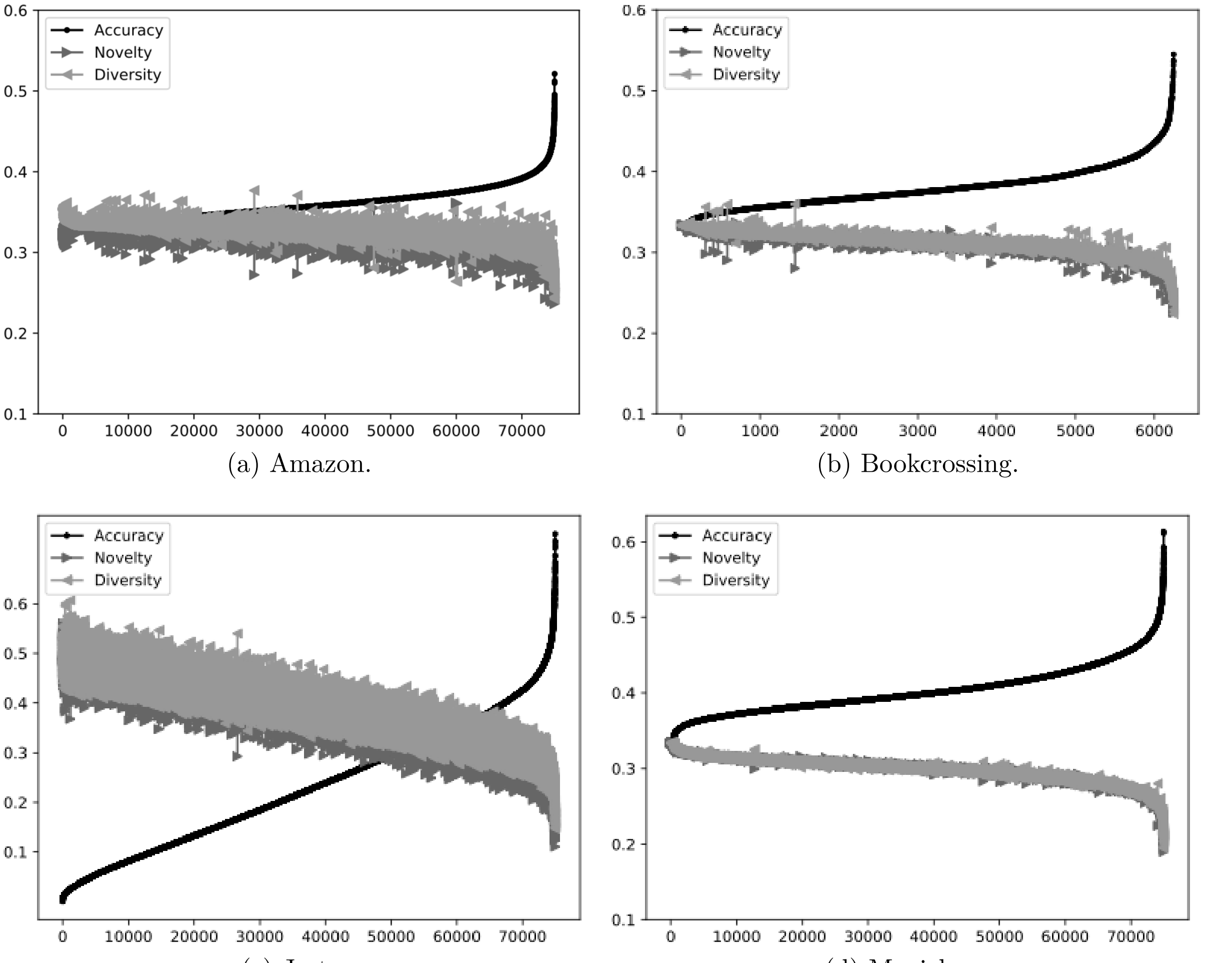
5.3.2 Experimental analysis

In this section, we present three experimental analyses. Firstly, in Section 5.3.2 a characterization of the users' preferences. Then, in Section 5.3.2, we evaluate the final recommendation results by the same fractional ranking strategy used in previous chapters. Finally, we evaluated the MO search process in Section 5.3.2.

User's preferences characterization

Since our goal is to meet users' preferences, a characterization of these data is essential to guide our analysis. A general analysis without assessing specific profiles of the users' preferences might not be sufficient since meeting users' preferences may lead to losses in some evaluation criteria in distinct scenarios.

Figure 5.2 presents the preferences of the users selected for testing, in which we sort users by the weight defined for accuracy. The x-axis represents the users, and the y-axis the weights for each objective. As described in Section 3.4.1, we apply a 5-fold cross-validation strategy. The charts in the figure present all selected users in all folds. Even though some users may appear more than one fold, the training data differs and can generate different preferences.



(a) Amazon.

(b) Bookcrossing.

(c) Jester.

(d) Movielens.

Figure 5.2: User's preferences (sorted by Accuracy).

Despite presenting some differences, users' preferences for Amazon, Bookcrossing, and Movielens datasets have many similarities. Accuracy is undoubtedly the essential recommendation quality factor for most users. In any case, we can observe in the charts that, for users who accept less accurate results (left side), there is a convergence of weights towards an equilibrium, *i.e.*, with equal weights for all objectives. Only for a minimal number of users (right side) does accuracy have a much higher priority, moving further away from the equilibrium above. Finally, novelty and diversity often have similar weights, although they may be conflicting objectives in the optimization process. We can observe a more significant differentiation in Amazon, a little less in Bookcrossing, and excellent proximity in Movielens.

Meanwhile, Jester turned out to be quite different from the other datasets. We can observe very opposite behaviors between the left and right sides of the chart. Some users accept a small accuracy in favor of novelty and diversity on the left side. On the opposite side, we have users who do not tolerate the loss of accuracy more severely than in the other three datasets. Jester has been quite divergent over the previous chapters. This behavior may be due to significant numerical differences such as the number of items and ratings, promoting a much less sparse dataset. However, the application domain, *jokes*, could also explain the very different preferences among users. The concept of a good joke can vary greatly and make defining ratings more difficult, especially when we remember that Jester has the broadest range of rating values available to users (-10 to 10).

According to the previous characterization, we perform the subsequent analyses based on three groups of users: (a) **All**: a general performance analysis considering all users; (b) (Accuracy) **Tolerant**: an analysis considering only 20% of users who have the lowest weights for accuracy and potentially higher weights for the other objectives; and (c) **High accuracy**: an analysis considering only 20% of users who have the highest weights for accuracy.

Overall recommendation analysis

Once again, we perform a general ranking analysis of the final recommendations, following the same strategy applied in previous chapters. Tables 5.1, 5.2, 5.3, and 5.4 summarize the resulting fractional rankings of the best configurations of each method for the three groups of users. Now we present configurations of the same method only when there is a statistical tie. In addition to the six evaluation criteria used above, we added the Euclidean distance calculated between the results obtained and the users' preferences in the *Objective Relevance Space*. Again, in Appendix C we present the mean values of the measures and their confidence intervals.

Table 5.1 presents the results for Amazon. We can observe great results for the PrefMO method, with a remote configuration in the first position of the overall ranking for the three groups of users, All, Tolerant, and High Accuracy. The use of *meta-features* (through FWLS) and the feature selection proved useful for High Accuracy users, while without *meta-features* (through HR) and feature selection obtained better results for Tolerant users. On the other hand, PrefMO configured with HR and FWLS were tied in the first place for All users. None of the best PrefMO configurations used the statistical significance test. On the other hand, PrefSO did not present very satisfactory results, being continually in the middle or lower part of the rankings. The WHF methods always occupied the first place in the NDCG ranking and sometimes a good ranking for other criteria but got a good overall ranking only for High Accuracy users. Regarding decision making, we can observe an advantage for the SUM configurations. In only one scenario, IndSUM appeared in one of the best configurations. We emphasize that PrefMO was always in the first place concerning the Distance criterion but still maintained competitive results for the other evaluation criteria.

#	Method	Distance	Ranking measures			G_{RISK} measures			Overall
			NDCG	EPD	EILD	NDCG	EPD	EILD	
	All users								
1	PrefMO-FWLS-Sel-IndDIST	**6.0**	9.0	**5.0**	**3.5**	**9.5**	**6.0**	**8.0**	**47.0**
1	PrefMO-HR-All-SUM	**6.0**	9.0	**5.0**	**3.5**	**9.5**	**6.0**	**8.0**	**47.0**
5	MO-Rank-HR-All-SUM	**6.0**	13.5	**5.0**	11.0	**9.5**	**6.0**	**8.0**	59.0
10	Biased-MF	19.0	13.5	12.5	11.0	**9.5**	14.5	**8.0**	88.0
12	SO-Rank-STREAM-All	15.0	19.0	17.5	18.5	**9.5**	**6.0**	**8.0**	93.5
14	HR-All	20.5	**1.5**	17.5	16.5	**9.5**	14.5	17.5	97.5
14	STREAM-All	20.5	**1.5**	17.5	16.5	**9.5**	14.5	17.5	97.5
15	PrefSO-STREAM-Sel	**6.0**	21.5	20.5	20.5	**9.5**	14.5	**8.0**	100.5
	Tolerant users								
1	PrefMO-HR-All-SUM	**8.0**	**4.0**	**9.5**	**9.5**	**6.5**	**2.5**	**4.0**	**44.0**
3	MO-Rank-HR-All-SUM	**8.0**	13.0	**9.5**	**9.5**	**6.5**	6.5	**4.0**	57.0
7	SO-Rank-HR-All	**8.0**	19.5	**9.5**	**9.5**	**6.5**	11.5	12.5	77.0
7	SO-Rank-STREAM-All	**8.0**	19.5	**9.5**	**9.5**	**6.5**	11.5	12.5	77.0
9	PrefSO-HR-All	**8.0**	21.5	**9.5**	**9.5**	15.5	11.5	12.5	88.0
9	PrefSO-STREAM-Sel	**8.0**	21.5	**9.5**	**9.5**	15.5	11.5	12.5	88.0
10	Biased-MF	17.5	13.0	**9.5**	**9.5**	15.5	15.0	15.0	95.0
12	HR-All	21.0	**4.0**	20.5	20.5	15.5	17.5	18.5	117.5
12	STREAM-All	21.0	**4.0**	20.5	20.5	15.5	17.5	18.5	117.5
	High accuracy users								
1	PrefMO-FWLS-Sel-SUM	**8.0**	4.5	8.5	7.5	**4.5**	**3.5**	**3.0**	**39.5**
3	HR-All	**8.0**	**1.5**	**2.0**	**1.5**	10.5	11.5	11.5	46.5
3	STREAM-All	**8.0**	**1.5**	**2.0**	**1.5**	10.5	11.5	11.5	46.5
4	MO-Rank-FWLS-Sel-SUM	**8.0**	11.0	8.5	7.5	**4.5**	**3.5**	8.0	51.0
11	Biased-MF	18.5	11.0	16.0	15.5	14.0	13.5	13.5	102.0
14	SO-Rank-STREAM-All	18.5	19.0	19.5	19.5	14.0	13.5	13.5	117.5
16	PrefSO-HR-All	18.5	20.5	21.5	21.5	19.0	16.5	19.5	137.0

Table 5.1: Fractional rankings for **Amazon**. Measures values in Table C.9.

Table 5.2 presents the results for Bookcrossing. We observed great competitiveness between the methods and some of their configurations. There are many ties in the first place of almost all rankings for the three user groups. Despite this, PrefMO was always present in the first place for the Distance criterion and overall ranking in the three user groups, maintaining great competitiveness with the other evaluation criteria. Now we can observe the occurrence of the PrefMO-St in the first place for overall ranking besides PrefMO and others. For MO methods, *meta-features* proved helpful for All users and not helpful for High accuracy users, while there are many occurrences of both for Tolerant users. We can observe the PrefMO-St configuration with a better overall ranking. However, they are always accompanied by their corresponding version without the statistical test in the dominance relation. Regarding decision making, there is again a great advantage for the SUM strategy, which was predominant for all MO methods, with few occurrences of IndSUM and no occurrence of IndDIST. Finally, the *constituent algorithms* were competitive against WHF, while PrefSO and SO-Rank methods showed the worst results for the three user groups.

#	Method	Distance	Ranking measures			G_{RISK} measures			Overall
			NDCG	EPD	EILD	NDCG	EPD	EILD	
All users									
1	PrefMO-HR-Sel-SUM	**6.0**	8.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	**59.5**
1	MO-Rank-STREAM-All-SUM	**6.0**	8.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	**59.5**
1	PrefMO-STREAM-All-IndSUM	**6.0**	8.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	**59.5**
1	PrefMO-STREAM-All-SUM	**6.0**	8.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	**59.5**
1	PrefMO-St-STREAM-All-IndSUM	**6.0**	8.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	**59.5**
3	Biased-SVD	**6.0**	20.5	**4.5**	**6.5**	**11.5**	**11.5**	**11.0**	71.5
6	FWLS-All	15.5	**2.5**	13.0	15.0	**11.5**	**11.5**	**11.0**	80.0
10	SO-Rank-FWLS-All	15.5	15.5	19.0	18.5	**11.5**	**11.5**	**11.0**	102.5
11	PrefSO-FWLS-All	15.5	15.5	19.0	20.5	**11.5**	**11.5**	**11.0**	104.5
Tolerant users									
1	MO-Rank-HR-Sel-SUM	**8.5**	**7.0**	**8.0**	**7.5**	**3.5**	**5.5**	**6.0**	**46.0**
1	PrefMO-HR-Sel-SUM	**8.5**	**7.0**	**8.0**	**7.5**	**3.5**	**5.5**	**6.0**	**46.0**
1	MO-Rank-STREAM-All-SUM	**8.5**	**7.0**	**8.0**	**7.5**	**3.5**	**5.5**	**6.0**	**46.0**
1	PrefMO-STREAM-All-SUM	**8.5**	**7.0**	**8.0**	**7.5**	**3.5**	**5.5**	**6.0**	**46.0**
1	PrefMO-St-STREAM-All-SUM	**8.5**	**7.0**	**8.0**	**7.5**	**3.5**	**5.5**	**6.0**	**46.0**
7	ALS	**8.5**	21.5	**8.0**	**7.5**	21.5	13.5	14.0	94.5
7	Biased-SVD	**8.5**	21.5	**8.0**	**7.5**	21.5	13.5	14.0	94.5
9	FWLS-All	19.0	**7.0**	19.0	17.0	12.0	18.5	18.5	111.0
10	SO-Rank-HR-All	19.0	**7.0**	19.0	21.0	12.0	18.5	18.5	115.0
11	PrefSO-FWLS-All	19.0	17.0	19.0	17.0	12.0	18.5	18.5	121.0
High accuracy users									
1	MO-Rank-HR-Sel-SUM	**11.5**	**7.5**	**10.5**	**9.5**	**3.5**	**2.0**	**7.0**	**51.5**
1	PrefMO-HR-Sel-SUM	**11.5**	**7.5**	**10.5**	**9.5**	**3.5**	**2.0**	**7.0**	**51.5**
1	PrefMO-St-HR-Sel-SUM	**11.5**	**7.5**	**10.5**	**9.5**	**3.5**	**2.0**	**7.0**	**51.5**
2	FWLS-All	**11.5**	**7.5**	**10.5**	**9.5**	**3.5**	8.0	**7.0**	57.5
7	ALS	**11.5**	18.5	**10.5**	**9.5**	19.0	19.0	19.0	107.0
7	Biased-SVD	**11.5**	18.5	**10.5**	**9.5**	19.0	19.0	19.0	107.0
8	SO-Rank-FWLS-All	**11.5**	18.5	**10.5**	20.5	19.0	19.0	19.0	118.0
8	PrefSO-FWLS-All	**11.5**	18.5	**10.5**	20.5	19.0	19.0	19.0	118.0

Table 5.2: Fractional rankings for **Bookcrossing**. Measures values in Table C.10.

Table 5.3 presents the results for Jester. We now observe an even more competitive scenario, but with a significant difference: an outstanding performance of the HR-All configuration of the WHF method, which was always in first place in the overall ranking for the three user groups alongside PrefMO and MO-Rank configurations. The *meta-features* proved to be more discreet, with few configurations among the first places. We observed the occurrence of PrefMO-St configurations with better overall ranking, once again accompanied by their corresponding version without the statistical test. The SUM decision-making strategy appears again in the spotlight but now sharing the top positions with IndDIST configurations. All methods showed promising results regarding the Distance criterion, except PrefSO and SO-Rank. Something similar occurred for almost all other evaluation criteria for the SO methods, confirming its inferior performance to the other methods in the overall ranking.

Table 5.4 presents the results for Movielens. We can observe the most divergent result among the four datasets for All and Tolerant users. In these two groups, PrefMO configurations were isolated in the first place in the Distance and EILD criteria. However, despite sharing the first place with other methods in some criteria, it obtained less expressive results in others, which moved it away from the first place in the overall ranking. In these two scenarios, the MO-Rank method was isolated in the first place in

#	Method	Distance	Ranking measures			G_{RISK} measures			Overall
			NDCG	EPD	EILD	NDCG	EPD	EILD	
All users									
1	HR-All	**9.5**	**2.0**	**5.0**	**6.5**	**7.5**	**8.5**	**8.5**	**47.5**
1	MO-Rank-HR-All-SUM	**9.5**	**2.0**	**5.0**	**6.5**	**7.5**	**8.5**	**8.5**	**47.5**
1	PrefMO-HR-All-IndDIST	**9.5**	**2.0**	**5.0**	**6.5**	**7.5**	**8.5**	**8.5**	**47.5**
4	Biased-SVD	**9.5**	14.0	13.0	15.0	**7.5**	**8.5**	**8.5**	76.0
10	SO-Rank-HR-All	20.0	19.0	19.0	19.0	16.5	**8.5**	**8.5**	110.5
11	PrefSO-HR-All	20.0	20.0	20.0	20.0	16.5	**8.5**	**8.5**	113.5
Tolerant users									
1	HR-All	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	MO-Rank-HR-All-SUM	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	PrefMO-HR-All-IndDIST	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	PrefMO-HR-All-SUM	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	PrefMO-St-HR-All-IndDIST	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	PrefMO-St-HR-All-SUM	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
1	MO-Rank-STREAM-Sel-SUM	**10.0**	**5.0**	**8.5**	**8.5**	**5.0**	**6.5**	**4.0**	**47.5**
4	UserKNN	**10.0**	14.0	**8.5**	**8.5**	13.5	13.5	13.5	81.5
9	SO-Rank-HR-All	**10.0**	19.0	19.0	19.0	16.5	16.5	15.5	115.5
10	PrefSO-HR-All	21.0	20.0	20.0	20.5	16.5	16.5	15.5	130.0
High accuracy users									
1	HR-All	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	MO-Rank-HR-All-SUM	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	PrefMO-HR-All-IndDIST	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	PrefMO-HR-All-SUM	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	PrefMO-St-HR-All-IndDIST	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	PrefMO-St-HR-All-SUM	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	MO-Rank-STREAM-Sel-SUM	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
1	PrefMO-STREAM-Sel-IndDIST	**8.5**	**6.0**	**9.0**	**7.5**	**5.0**	**5.0**	**5.0**	**46.0**
3	Biased-SVD	**8.5**	15.0	**9.0**	**7.5**	11.5	11.5	11.5	74.5
9	SO-Rank-HR-All	18.5	19.5	19.5	19.0	16.0	16.0	16.0	124.5
10	PrefSO-HR-All	18.5	19.5	19.5	20.5	16.0	16.0	16.0	126.0

Table 5.3: Fractional rankings for **Jester**. Measures values in Table C.11.

the overall ranking. On the other hand, the results are a little more like previous datasets for the HIGH ACCURACY users. However, the MO methods excel in all evaluation criteria and emphasize WHF, which obtained good results for the Distance criterion and the ranking measures. MO-Rank took advantage of *meta-features*, with the STREAM strategy always appearing in the configurations with the best rankings. Meanwhile, PrefMO did not benefit much with *meta-features*, having the HR as the highest-ranked configuration for and HIGH ACCURACY users. The same occurred concerning the use of the statistical test in the dominance relation, with no PrefMO-St configuration present among those selected. The IndDIST decision-making strategy proved to help obtain the smallest distances from users' preferences for and TOLERANT users. However, with worse results for other evaluation criteria, it did not appear among the best configurations in the overall ranking, leaving the spotlight again for the SUM decision-making strategy.

These results show a significant variation in the rankings between the three groups of users and the different datasets, demonstrating how competitive and complex the MO optimization problem can be when considering the users' preferences regarding such objectives. However, the proposed PrefMO method showed effective results, achieving excellent results concerning the Distance criterion while maintaining good results relating to the other criteria in all datasets, with a very positive highlight for the Amazon dataset.

#	Method	Distance	Ranking measures			G_{RISK} measures			Overall
			NDCG	EPD	EILD	NDCG	EPD	EILD	
All users									
1	MO-Rank-STREAM-Sel-SUM	7.5	5.0	**2.5**	4.5	**7.0**	**7.0**	**7.5**	**41.0**
3	PrefMO-HR-Sel-SUM	12.0	**2.0**	**2.5**	8.5	**7.0**	**7.0**	**7.5**	46.5
4	PrefMO-HR-Sel-IndDIST	**2.0**	15.0	7.0	**1.5**	**7.0**	**7.0**	**7.5**	47.0
7	HR-All	12.0	**2.0**	7.0	13.5	**7.0**	**7.0**	**7.5**	56.0
13	ItemKNN	17.5	10.5	14.0	17.0	**7.0**	**7.0**	**7.5**	80.5
19	PrefSO-FWLS-Sel	17.5	20.0	19.5	18.0	16.0	16.0	16.5	123.5
20	SO-Rank-FWLS-All	20.5	19.0	19.5	21.0	16.0	16.0	16.5	128.5
20	SO-Rank-HR-Sel	19.0	21.0	21.0	19.0	16.0	16.0	16.5	128.5
Tolerant users									
1	MO-Rank-STREAM-Sel-SUM	10.0	**4.5**	**5.0**	7.5	**2.5**	**3.0**	**2.5**	**35.0**
2	PrefMO-STREAM-Sel-SUM	10.0	**4.5**	**5.0**	7.5	6.5	**3.0**	**2.5**	39.0
5	PrefMO-HR-Sel-IndDIST	**2.0**	14.5	**5.0**	**2.0**	12.0	8.0	**2.5**	46.0
10	FWLS-All	19.5	**4.5**	13.0	14.5	**2.5**	8.0	11.0	73.0
15	ItemKNN	21.0	11.0	17.0	18.0	9.5	12.0	13.0	101.5
16	PrefSO-FWLS-Sel	5.5	20.0	20.0	18.0	16.0	16.0	15.5	111.0
17	SO-Rank-HR-Sel	5.5	21.5	20.0	18.0	16.0	16.0	15.5	112.5
High accuracy users									
1	MO-Rank-HR-Sel-SUM	**6.0**	**3.5**	**5.5**	**6.0**	**2.5**	**2.5**	**3.0**	**29.0**
1	PrefMO-HR-Sel-SUM	**6.0**	**3.5**	**5.5**	**6.0**	**2.5**	**2.5**	**3.0**	**29.0**
1	MO-Rank-STREAM-Sel-SUM	**6.0**	**3.5**	**5.5**	**6.0**	**2.5**	**2.5**	**3.0**	**29.0**
3	FWLS-All	**6.0**	**3.5**	**5.5**	**6.0**	7.5	7.0	8.0	43.5
3	HR-All	**6.0**	**3.5**	**5.5**	**6.0**	7.5	7.0	8.0	43.5
9	ItemKNN	14.5	9.5	13.0	17.0	12.0	11.5	12.0	89.5
13	PrefSO-FWLS-Sel	19.5	19.5	19.5	19.0	16.0	16.0	16.5	126.0
14	SO-Rank-FWLS-All	19.5	19.5	19.5	20.5	16.0	16.0	16.5	127.5

Table 5.4: Fractional rankings for **Movielens**. Measures values in Table C.12.

In this complex context, we observe that *meta-features* can be helpful in some scenarios, while the decision-making strategy can also vary between the experimented scenarios.

Multi-objective optimization analysis

Our last experimental analysis aims to assess the usefulness of the *Individualized Extreme Dominance* in the MO search. In addition to the Hypervolume (**HV**), used in Section 3.4.2, we used other two widely used quality indicators [Durillo and Nebro, 2011]: (a) *Generational Distance* (**GD**): measures the approximation of the *individuals* returned by the MO search and *reference points*; and (b) *Inverted Generational Distance* (**IGD**): measures the approximation of the *reference points* and the *individuals* returned by the MO search. As *reference points*, we use the users' preferences. Therefore, GD and IGD have a close relationship with our goal of reducing the distance of the obtained results concerning the users' preferences in the *Objective Relevance Space*.

Table 5.5 presents the ranking result of all quality indicators for the four datasets. We use the same fractional ranking strategy and generate an overall ranking similar to the above results. Following the same strategy, we present only the best-ranked configuration of each method in each criterion, including some tied where relevant. GD and IGD were evaluated for the three groups of users individually.

We can observe an excellent result for the PrefMO method, ranking first for all GD and IGD measures and, consequently, in the overall ranking, for all datasets. The PrefMO-

#	Method	HV	Users Group						Overall
			All		Tolerant		High accuracy		
			GD	IGD	GD	IGD	GD	IGD	
	Amazon								
1	PrefMO-St-HR-All	6.0	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**1.0**	**12.0**
4	MO-Rank-HR-All	3.5	4.0	4.0	4.0	4.0	4.0	4.0	27.5
5	MO-Rank-FWLS-Sel	**1.5**	5.5	5.5	5.5	5.5	5.5	5.5	34.5
5	PrefMO-Rank-FWLS-Sel	**1.5**	5.5	5.5	5.5	5.5	5.5	5.5	34.5
	Bookcrossing								
1	PrefMO-St-STREAM-All	4.0	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**13.0**
3	MO-Rank-HR-Sel	4.0	3.5	3.5	3.5	3.5	3.5	3.5	25.0
	Jester								
1	PrefMO-St-HR-All	5.5	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**14.5**
1	PrefMO-St-STREAM-Sel	5.5	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**1.5**	**14.5**
2	MO-Rank-HR-All	**2.0**	4.0	4.0	4.0	4.0	4.0	4.0	26.0
	Movielens								
1	PrefMO-HR-Sel	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**17.5**
1	PrefMO-STREAM-Sel	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**2.5**	**17.5**
3	MO-Rank-HR-Sel	**2.5**	5.5	5.5	5.5	5.5	5.5	5.5	35.5
3	MO-Rank-STREAM-Sel	**2.5**	5.5	5.5	5.5	5.5	5.5	5.5	35.5

Table 5.5: Ranking analysis of quality indicators for the MO search results.

St configuration ranked first for GD and IGD on Amazon, Bookcrossing, and Jester, showing its great utility for the MO search. On the other hand, the best configuration in Movielens does not use the statistical test, highlighting that PrefMO also achieved a good result for HV in this scenario.

5.3.3 Discussion

The previous section presents analysis of experimental results allowing us to observe the behavior of the methods and the parameters introduced in this chapter and enabling us to answer our third research question: ***RQ3****: "Does explicitly incorporating individual preferences of users concerning the optimized objectives contribute to improving the results of multi-objective recommendation?"*.

The analysis of the MO search results shows the excellent performance of the proposed strategy. It achieves a closer approximation of user expectations through a better Pareto set concerning GD and IGD quality indicators in all datasets and groups of users. This result demonstrates that the explicit use of users' preferences during the optimization process is beneficial to obtaining better MO search results. Additionally, PrefMO achieved very competitive results in all experimental scenarios regarding the final recommendation results. We have an exceptional result in Amazon, great competitiveness in Bookcrossing and Jester, and a slightly less expressive result in Movielens. These results allow a positive answer to ***RQ3***.

However, the final recommendation results were achieved by different configurations from those that obtained the best MO search results. Surprisingly, the best Pareto sets were not enough to generate the best recommendations after the decision-making. This issue demonstrates that it is possible to find better ways to combine the search and decision-making strategies to obtain better results, opening the door for intriguing and relevant future work. We believe that a large number of users is a relevant factor for this issue; exploring the grouping of users or a stratified sampling of users to reduce the complexity of the search or generating different models for different groups of users can be good alternatives for improving the method. Another relevant issue is that the statistical test tends to reduce the size of the Pareto set in the MO search, reducing the offer of choices for the numerous users during the decision-making process. Finding a way to expand the number of solutions without degrading the results in GD and IGD is also a good alternative for future work.

5.4 Concluding remarks

This chapter presents a preference-based MO recommender method to answer our third research question and improve the knowledge of this unique recommendation problem. We ended the search for answers to all our research questions and advanced knowledge on three issues involved with the MO recommendation. In the next chapter, we conclude this thesis with some final considerations.

Chapter 6

Conclusions and Future Work

In this thesis, we aimed to expand the knowledge of Multi-Objective Filtering (MOF) in the light of three new perspectives. We formulated three research questions and defined methods capable of helping us to achieve our desires. Firstly, we explicitly explored statistical measures capable of expressing some input data characteristics, defining the *meta-featured MOF*. Secondly, we advanced our journey in MO recommendation knowledge, aiming to improve the risk-sensitiveness of RSs to multiple evaluation criteria by defining the *Risk-sensitive MOF*. Finally, we have gone even further, introducing user expectations concerning the multiple optimized objectives by defining the *Preference-based MOF*. We have produced numerous experiments with competitive baseline methods and performed an in-depth analysis of results that allowed us to positively answer our three research questions, achieving our goals and reaching some contributions discussed in Section 6.1. Then, we present directions for future work in Section 6.2.

6.1 Contributions

RQ1: "Does explicitly incorporating meta-features contribute to improving the results of multi-objective recommendation?"

To answer this question, we combined hybrid and multi-objective filtering methods proposing the *Meta-featured MOF* strategy. This new recommendation strategy uses a set of general *meta-features* and *constituent algorithms* based on Content-Based Filtering and Collaborative Filtering. These resources are processed, combined using a hybridization strategy, and selected for the MO evolutionary search. The result of this search is a set of Pareto-based optimal solutions. Finally, optimal solutions are selected to make specific recommendations.

We performed an extensive computational experiment in four databases that provide different conditions of recommendation, with different volumes and nature of data. We compare results with solid and competitive baselines. In this way, the proposed strategy was validated, demonstrating that it can achieve its goals, producing valuable recommendations on various evaluation criteria, surpassing the baselines in some scenarios, and obtaining competitive results in others.

Therefore, we conclude with a positive answer to our first research question: *explicitly incorporating meta-features can contribute to improving multi-objective recommendation results.*

RQ2: "Does explicitly incorporating risk-sensitive measures contribute to improving the results of multi-objective recommendation?"

To answer this question, we improved the strategy initially proposed, proposing the *Risk-sensitive MOF*. This new recommendation strategy uses a risk-sensitive measure applied individually to the multiple optimized objectives. We kept the optimization focus on the primary objectives, but we started to guide the search through a new dominance relation called *Risk-sensitive dominance*. For this, we adapted and modeled the risk sensitivity computation and an evolutionary meta-heuristic for the recommendation context, specifically for the Multi-Objective Filtering.

Again, we performed extensive experiments using the same experimental procedure as before, including the best baselines and comparing them against different criteria. Thus, we validated our second strategy, demonstrating that it can achieve its goals, producing valuable recommendations on various evaluation criteria and still managing to reduce losses and degradation (poor results) while maintaining a good overall average result in many scenarios.

Therefore, we conclude with a positive answer to our second research question: *explicitly incorporating risk-sensitive measures can contribute to improving multi-objective recommendation results.*

RQ3: "Does explicitly incorporating individual preferences of users concerning the optimized objectives contribute to improving the results of multi-objective recommendation?"

To answer this question, we once again improved the strategy initially proposed, proposing the *Preference-based MOF*. This new recommendation strategy explores the users' preferences regarding the optimized objectives applied individually to each objective. We kept the optimization focus on the primary objectives, but we started to guide the search through a new dominance relation called *Individualized Extreme Dominance*. This new dominance relation considers three essential concepts, *Pareto dominance*, *Extreme Dominance*, and *Statistical Significance Tests*, to guide the MO search considering the importance of each objective individually and robustly. For this, we adapted and modeled the users' preferences computation and an evolutionary meta-heuristic for the recommendation context, specifically for the Multi-Objective Filtering.

Again, we performed extensive experiments using the same experimental procedure as before, including the best baselines and comparing them against different criteria. We validated our third strategy, demonstrating that it can achieve its goals, producing valuable recommendations on various evaluation criteria and approaching the individual expectations of users concerning the optimized objectives in many scenarios.

Therefore, we conclude with a positive answer to our third research question: *explicitly incorporating individual preferences of users concerning the optimized objectives can contribute to improving multi-objective recommendation results.*

Practical contributions

From a theoretical perspective, this thesis makes significant contributions to the state-the-of-art in the RS literature in the context of MOF systems by answering the previous three research questions. On the other hand, from a practical point of view, we implement and test our proposed concepts and strategies. Our extensive experiments demonstrate that our methods have a higher chance of better satisfying user needs than the current state-of-the-art concerning the different scopes and requirements involving risk-sensitiveness and preference-based optimization and using additional features to improve results.

However, the proposed strategies must be viable for real applications to take advantage of all this. Although there are many processes, most are independent and parallelizable processes that can run offline or online to make them viable. The computation of *meta-features* and the execution of the constituent algorithms can be done offline for all users and updated online whenever there are data updates. The same for the features building performed from this data. MO search methods can be trained offline whenever more significant data updates occur. Bearing in mind that after obtaining an initial Pareto set, it is already possible to make valuable recommendations, and this set can be improved at any time. Decision-making and the production of final recommendations are faster and simpler processes that can be performed online without compromising performance. However, more complex decision-making can also run offline if necessary.

6.2 Future work

Features engineering

We selected the same set of *meta-features* based on general-purpose statistical measures for all datasets without regard to the constituent algorithms. However, exploring specific measures based on particular characteristics of each dataset or defining measures related to the strategies used by the algorithms can contribute even more to better results. Additionally, the exploitation of new *meta-features* that involve other characteristics, such as performance estimators, risk sensitivity, and user preferences, can be very promising.

Regarding the *constituent algorithms*, we explored a wide range of Collaborative Filtering algorithms and only one Content-Based Filtering algorithm. Therefore, including more variety of Content-Based Filtering algorithms and inserting algorithms from other approaches also has excellent potential to improve results.

Regarding feature selection, we explored a simple strategy based on three filtering strategies without considering optimization objectives. Exploring more advanced feature engineering techniques to identify more complex and hidden patterns, considering the optimized objectives, can generate exciting results. Applying machine learning techniques to generate new features can also provide relevant results.

Finally, in analyzing our experiments, we did not specifically assess how and when *meta-features* contributed to the results. We focused on defining a strategy that would exploit these resources and answer our research question. In this way, deepening the analysis of results in a more profound way is essential to carry out the work suggested in the previous paragraphs and to understand the usefulness of the *meta-features* better.

Risk-sensitiveness and fairness

Our method explored only one risk-sensitive measure, the G_{RISK}, without evaluating the impact of the α parameter. Thus, deepening risk-sensitiveness experiments through different α values and other potential risk sensitive measures, would be interesting.

A topic that may be closely related to risk-sensitiveness, and which is currently gaining notoriety, is referred to as *Fairness*. Exploring fairness measures and assessing their relationship to risk sensitivity is also an intriguing opportunity for future work.

Finally, in this thesis, we use user preferences only concerning the primary objectives of accuracy, novelty, and diversity. It would also be exciting to explore users' preferences regarding risk-sensitive (and fairness) for each objective.

Optimization processes

For all the strategies proposed in this thesis, we use only the NSGA-II as an algorithm for the evolutionary search for Pareto solutions. We demonstrate that our strategies helped obtain good Pareto sets and answered our research questions. However, using other meta-heuristics or defining specific heuristics can provide even better and more expressive results.

Additionally, our strategies generate results for all users, creating a single Pareto set for all, but with the ability to generate individual choices for each user. Thus, defining strategies for the selection or grouping of users to reduce the complexity of the MO processes and increase the Pareto set obtained, providing more options for decision-making choices without lowering the quality of the obtained Pareto sets, can be very promising.

However, we observed an even more relevant issue. Despite providing better Pareto sets, the decision-making strategies were not able to properly benefit from this to make even better recommendations. Therefore, improving the decision-making methods to more efficiently explore the Pareto sets resulting from the MO search becomes very relevant for improving results.

Optimization objectives

We apply our strategies to just three objectives: accuracy, novelty, and diversity. We used only one measure for each criterion: NDCG, EPD, and EILD, respectively. Evaluating the behavior and capacity of our strategies for other criteria and other measures, also involving objectives for multiple stakeholders, would be interesting.

Bibliography

Eytan Adar, Jaime Teevan, Susan T. Dumais, and Jonathan L. Elsas. The web changes everything: Understanding the dynamics of web content. In *ACM WSDM*, 2009.

Gediminas Adomavicius and Alexander Tuzhilin. Toward the Next Generation of Recommender Systems: A Survey of the State-of-the-Art and Possible Extensions. *IEEE TKDE*, pages 734–749, June 2005.

Gediminas Adomavicius and Jingjing Zhang. Impact of Data Characteristics on Recommender Systems Performance. *ACM TMIS*, 3(1):3:1–3:17, 2012.

Charu C. Aggarwal. *Recommender Systems: The Textbook*. Springer International Publishing, Cham, 2016. ISBN 978-3-319-29657-9 978-3-319-29659-3.

Ricardo A. Baeza-Yates and Berthier Ribeiro-Neto. *Modern Information Retrieval*. Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1999.

Xinlong Bao, Lawrence Bergman, and Rich Thompson. Stacking Recommendation Engines with Additional Meta-features. In *ACM RecSys*, pages 109–116, New York, NY, USA, 2009. ISBN 978-1-60558-435-5.

Slim Bechikh, Marouane Kessentini, Lamjed Ben Said, and Khaled Ghédira. Preference Incorporation in Evolutionary Multiobjective Optimization: A Survey of the State-of-the-Art. In *Advances in Computers*, volume 98, pages 141 – 207. Elsevier, 2015.

Lamjed Ben Said, Slim Bechikh, and Khaled Ghedira. The r-Dominance: A New Dominance Relation for Interactive Evolutionary Multicriteria Decision Making. *IEEE TEVC*, pages 801–818, October 2010.

Michael Bendersky, W. Bruce Croft, and Yanlei Diao. Quality-biased ranking of web documents. In *WSDM*, 2011.

Djallel Bouneffouf. Contextual bandit algorithm for risk-aware recommender systems. In *IEEE CEC*, pages 4667–4674, Vancouver, BC, Canada, 2016. IEEE.

Djallel Bouneffouf, Amel Bouzeghoub, and Alda Lopes Ganarski. Risk-aware recommender systems. In *Neural Information Processing*, pages 57–65. Springer Berlin Heidelberg, 2013.

John S. Breese, David Heckerman, and Carl Kadie. Empirical Analysis of Predictive Algorithms for Collaborative Filtering. In *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, UAI'98, pages 43–52, San Francisco, CA, USA, 1998. Morgan Kaufmann Publishers Inc.

Robin Burke. Hybrid Recommender Systems: Survey and Experiments. *User Modeling and User-Adapted Interaction*, 12(4):331–370, November 2002. ISSN 0924-1868, 1573-1391.

Robin Burke. Hybrid Web Recommender Systems. In *The Adaptive Web*, number 4321 in Lecture Notes in Computer Science, pages 377–408. Springer Berlin Heidelberg, January 2007. ISBN 978-3-540-72078-2 978-3-540-72079-9.

X. Cai, Z. Hu, P. Zhao, W. Zhang, and J. Chen. A hybrid recommendation system with many-objective evolutionary algorithm. *Expert Systems with Applications,*, vol. 159, 2020.

Xiaoye Cheng, Jingjng Zhang, and Lu (Lucy) Yan. Understanding the Impact of Individual Users Rating Characteristics on Predictive Accuracy of Recommender Systems. SSRN Scholarly Paper ID 3132681, Social Science Research Network, Rochester, NY, 2018.

Paolo Cremonesi, Yehuda Koren, and Roberto Turrin. Performance of recommender algorithms on top-n recommendation tasks. In *ACM RecSys*, pages 39–46, 2010.

Tiago Cunha, Carlos Soares, and André C. P. L. F. de Carvalho. Selecting Collaborative Filtering Algorithms Using Metalearning. In Paolo Frasconi, Niels Landwehr, Giuseppe Manco, and Jilles Vreeken, editors, *Machine Learning and Knowledge Discovery in Databases*, Lecture Notes in Computer Science, pages 393–409. Springer International Publishing, 2016.

Tiago Cunha, Carlos Soares, and André C. P. L. F. de Carvalho. Algorithm Selection for Collaborative Filtering: the influence of graph metafeatures and multicriteria metatargets. *arXiv:1807.09097 [cs, stat]*, 2018a.

Tiago Cunha, Carlos Soares, and André C. P. L. F. de Carvalho. Metalearning and Recommender Systems: A literature review and empirical study on the algorithm selection problem for Collaborative Filtering. *Information Sciences*, 423:128–144, 2018b.

D. Cvetkovic and I.C. Parmee. Preferences and their application in evolutionary multiobjective optimization. *IEEE Transactions on Evolutionary Computation,*, vol. 6(1):pp. 42–57, February 2002.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *Journal of Transactions on Evolutionary Computation*, 6:182–197, 2002.

Yashar Deldjoo, Alejandro Bellogin, and Tommaso Di Noia. Explaining recommender systems fairness and accuracy through the lens of data characteristics. *Information Processing & Management*, 58(5):102662, September 2021. ISSN 0306-4573. doi: 10.1016/j.ipm.2021.102662.

B. Taner Dinçer, Craig Macdonald, and Iadh Ounis. Hypothesis testing for the risk-sensitive evaluation of retrieval systems. In *Proceedings of the 37th ACM SIGIR*, page 2332, New York, NY, USA, 2014. ACM.

B. Taner Dinçer, Craig Macdonald, and Iadh Ounis. Risk-sensitive evaluation and learning to rank using multiple baselines. In *Proceedings of the 39th ACM SIGIR*, page 483492, New York, NY, USA, 2016. ACM.

B. Taner Dinçer, Iadh Ounis, and Craig Macdonald. Tackling Biased Baselines in the Risk-Sensitive Evaluation of Retrieval Systems. *Proceeding of the 36th ECIR*, 8416: 26–38, 2014.

Juan J. Durillo and Antonio J. Nebro. jMetal: A Java framework for multi-objective optimization. *Advances in Engineering Software*, 42(10):760–771, 2011.

M. Ehrgott. A characterization of lexicographic max-ordering solutions. In *Workshop of the DGOR-Working Group Multicriteria Optimization and Decision Theory*, pages 193–202, 1997.

Matthias Ehrgott. *Multicriteria optimization*, volume 491. Springer Science & Business Media, 2005.

Michael D. Ekstrand. Lenskit for python: Next-generation software for recommender systems experiments. In *Proceedings of the 29th ACM International Conference on Information and Knowledge Management*, CIKM '20, page 29993006, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450368599. doi: 10.1145/3340531.3412778. URL `https://doi.org/10.1145/3340531.3412778`.

Michael D. Ekstrand, John T. Riedl, and Joseph A. Konstan. Collaborative Filtering Recommender Systems. *Found. Trends Hum.-Comput. Interact.*, 4(2):81–173, February 2011.

Ernestas Filatovas, Algirdas Laninskas, Olga Kurasova, and Julius ilinskas. A preference-based multi-objective evolutionary algorithm R-NSGA-II with stochastic local search.

Central European Journal of Operations Research,, volume 25(4):pp. 859–878, December 2017.

Carlos M. Fonseca and Peter J. Fleming. Genetic Algorithms for Multiobjective Optimization: Formulation, Discussion and Generalization. In *Proceedings of the 5th International Conference on Genetic Algorithms*, pages 416–423, June 1993.

Reinaldo Silva Fortes, Alan R. R. de Freitas, and Marcos André Gonçalves. A Multicriteria Evaluation of Hybrid Recommender Systems: On the Usefulness of Input Data Characteristics. volume 2, pages 623–633, May 2017. ISBN 978-989-758-248-6.

Reinaldo Silva Fortes, Anisio Lacerda, Alan Freitas, Carlos Bruckner, Dayanne Coelho, and Marcos Gonçalves. User-Oriented Objective Prioritization for Meta-Featured Multi-Objective Recommender Systems. In *Adjunct Publication of the 26th Conference on User Modeling, Adaptation and Personalization*, UMAP '18, pages 311–316, New York, NY, USA, 2018. ACM. ISBN 978-1-4503-5784-5. doi: 10.1145/3213586.3225243.

Reinaldo Silva Fortes, Daniel Xavier de Sousa, Dayanne G. Coelho, Anisio M. Lacerda, and Marcos A. Gonçalves. Individualized extreme dominance (IndED): A new preference-based method for multi-objective recommender systems. *Information Sciences*, 572:558–573, September 2021. ISSN 0020-0255. doi: 10.1016/j.ins.2021.05.037. URL https://www.sciencedirect.com/science/article/pii/S0020025521004977.

Zeno Gantner, Steffen Rendle, Christoph Freudenthaler, and Lars Schmidt-Thieme. MyMediaLite: A Free Recommender System Library. In *ACM RecSys*, pages 305–308, New York, NY, USA, 2011.

Yingqiang Ge, Shuyuan Xu, Shuchang Liu, Zuohui Fu, Fei Sun, and Yongfeng Zhang. Learning personalized risk preferences for recommendation. In *Proceedings of the 43rd International ACM SIGIR Conference on Research and Development in Information Retrieval*, SIGIR 20, page 409418, New York, NY, USA, 2020. Association for Computing Machinery. ISBN 9781450380164. doi: 10.1145/3397271.3401056. URL https://doi.org/10.1145/3397271.3401056.

Bingrui Geng, Lingling Li, Licheng Jiao, Maoguo Gong, Qing Cai, and Yue Wu. NNIA-RS: A multi-objective optimization based recommender system. *Physica A: Statistical Mechanics and its Applications*, 424:383–397, 2015.

Ken Goldberg, Theresa Roeder, Dhruv Gupta, and Chris Perkins. Eigentaste: A Constant Time Collaborative Filtering Algorithm. *Inf. Retr.*, pages 133–151, July 2001.

Asela Gunawardana and Guy Shani. A Survey of Accuracy Evaluation Metrics of Recommendation Tasks. *J. Mach. Learn. Res.*, pages 2935–2962, 2009.

Ruining He and Julian McAuley. Ups and downs: Modeling the visual evolution of fashion trends with one-class collaborative filtering. In *WWW'2016*, page 507517, Republic and Canton of Geneva, CHE, 2016. ACM.

Xiangnan He, Lizi Liao, Hanwang Zhang, Liqiang Nie, Xia Hu, and Tat-Seng Chua. Neural collaborative filtering. In *Proceedings of the 26th international conference on world wide web*, Republic and Canton of Geneva, CHE, 2017. ACM.

Jonathan L. Herlocker, Joseph A. Konstan, Al Borchers, and John Riedl. An Algorithmic Framework for Performing Collaborative Filtering. In *ACM SIGIR*, pages 230–237, 1999.

Jonathan L. Herlocker, Joseph A. Konstan, Loren G. Terveen, and John T. Riedl. Evaluating Collaborative Filtering Recommender Systems. *ACM TOIS*, pages 5–53, January 2004.

Jianjie Hu, Guo Yu, Jinhua Zheng, and Juan Zou. A preference-based multi-objective evolutionary algorithm using preference selection radius. *Soft Computing*, September 2017.

Niall Hurley and Scott Rickard. Comparing measures of sparsity. *IEEE Trans. Inf. Theor.*, 2009.

H. Ishibuchi, N. Tsukamoto, and Y. Nojima. Evolutionary many-objective optimization: A short review. In *IEEE Congress on Evolutionary Computation, 2008. CEC 2008.*, pages 2419–2426, June 2008.

Ankush Jain, Pramod Kumar Singh, and Joydip Dhar. Multi-objective item evaluation for diverse as well as novel item recommendations. *Expert Systems with Applications*, 139:112857, January 2020.

Dietmar Jannach, Markus Zanker, Alexander Felfernig, and Gerhard Friedrich. *Recommender Systems: An Introduction.* Cambridge University Press, New York, September 2010.

Dietmar Jannach, Lukas Lerche, Iman Kamehkhosh, and Michael Jugovac. What recommenders recommend: an analysis of recommendation biases and possible countermeasures. In *UMUAI*, 2015.

Olivier Jeunen. Revisiting offline evaluation for implicit-feedback recommender systems. In *RecSys'2019*, page 596600, New York, NY, USA, September 2019. ACM.

Michael Jugovac, Dietmar Jannach, and Lukas Lerche. Efficient optimization of multiple recommendation quality factors according to individual user tendencies. *Expert Systems with Applications*, pages 321–331, September 2017.

Komal Kapoor, Vikas Kumar, Loren Terveen, Joseph A. Konstan, and Paul Schrater. "I Like to Explore Sometimes": Adapting to Dynamic User Novelty Preferences. In *ACM RecSys*, 2015.

J. Kennedy and R. Eberhart. Particle swarm optimization. In *Proceedings of ICNN'95 - International Conference on Neural Networks*, pages 1942–1948, 1995.

Shah Khusro, Zafar Ali, and Irfan Ullah. Recommender systems: Issues, challenges, and research opportunities. In Kuinam J. Kim and Nikolai Joukov, editors, *Information Science and Applications (ICISA) 2016*, pages 1179–1189, Singapore, 2016. Springer Singapore.

Bart P. Knijnenburg, Martijn C. Willemsen, Zeno Gantner, Hakan Soncu, and Chris Newell. Explaining the user experience of recommender systems. *Journal of User Modeling and User-Adapted Interaction*, 22:441–504, 2012.

Bingdong Li, Jinlong Li, Ke Tang, and Xin Yao. Many-Objective Evolutionary Algorithms: A Survey. *ACM CSUR*, pages 13:1–13:35, September 2015.

Jinzhong Li, Guanjun Liu, Chungang Yan, and Jiang Changjun. Robust Learning to Rank Based on Portfolio Theory and AMOSA Algorithm. *Journal of IEEE Transactions on Systems, Man, and Cybernetics: Systems*, 47:1–12, 2016.

Xiaoyuan Liang, Jie Tian, Xiaoning Ding, and Guiling Wang. A risk and similarity aware application recommender system. *CIT*, 23:303–315, 2015.

Qiuzhen Lin, Xiaozhou Wang, Bishan Hu, Lijia Ma, Fei Chen, Jianqiang Li, and Carlos A. Coello Coello. Multiobjective Personalized Recommendation Algorithm Using Extreme Point Guided Evolutionary Computation, November 2018.

Tie-Yan Liu. *Learning To Rank For Information Retrieval.* Springer, New York, USA, 2011.

Zhengchao Liu, Shunsheng Guo, Lei Wang, Baigang Du, and Shibao Pang. A multi-objective service composition recommendation method for individualized customer: Hybrid MPA-GSO-DNN model. *Comp. & Industrial Engineering,*, vol. 128, February 2019.

Jarana Manotumruksa, Craig Macdonald, and Iadh Ounis. A contextual recurrent collaborative filtering framework for modelling sequences of venue checkins. *Information Processing and Management*, 56:102092, 2019.

Julian McAuley, Christopher Targett, Qinfeng Shi, and Anton van den Hengel. Image-based recommendations on styles and substitutes. In *SIGIR'2015*, page 4352, New York, NY, USA, 2015. ACM.

Michael McCandless, Erik Hatcher, and Otis Gospodnetic. *Lucene in Action, Second Edition: Covers Apache Lucene 3.0*. Manning Publications Co., Greenwich, CT, USA, 2010.

Sean M. McNee, John Riedl, and Joseph A. Konstan. Being Accurate is Not Enough: How Accuracy Metrics Have Hurt Recommender Systems. In *ACM CHI EA*, pages 1097–1101, New York, NY, USA, 2006.

Julián Molina, Luis V. Santana, Alfredo G. Hernández-Díaz, Carlos A. Coello Coello, and Rafael Caballero. g-dominance: Reference point based dominance for multiobjective metaheuristics. *EJOR*, pages 685–692, September 2009.

Antonio J. Nebro, Juan J. Durillo, and Matthieu Vergne. Redesigning the jMetal Multi-Objective Optimization Framework. In *Proceedings of the Companion Publication of the 2015 Annual Conference on Genetic and Evolutionary Computation*, GECCO Companion '15, pages 1093–1100, New York, NY, USA, 2015. ACM.

Fabian Pedregosa, Gaël Varoquaux, Alexandre Gramfort, Vincent Michel, Bertrand Thirion, Olivier Grisel, Mathieu Blondel, Gilles Louppe, Peter Prettenhofer, Ron Weiss, Vincent Dubourg, Jake Vanderplas, Alexandre Passos, David Cournapeau, Matthieu Brucher, Matthieu Perrot, and Edouard Duchesnay. Scikit-learn: Machine Learning in Python. *arXiv:1201.0490 [cs]*, January 2012. URL `http://arxiv.org/abs/1201.0490`. arXiv: 1201.0490.

Gustavo Penha and Rodrygo L. T. Santos. Exploiting Performance Estimates for Augmenting Recommendation Ensembles. In *Fourteenth ACM Conference on Recommender Systems*, RecSys '20, pages 111–119. ACM, 2020.

Fábio Pinto, Carlos Soares, and João Mendes-Moreira. Towards Automatic Generation of Metafeatures. In James Bailey, Latifur Khan, Takashi Washio, Gill Dobbie, Joshua Zhexue Huang, and Ruili Wang, editors, *Advances in Knowledge Discovery and Data Mining*, Lecture Notes in Computer Science, pages 215–226. Springer International Publishing, 2016.

Evaggelia Pitoura, Kostas Stefanidis, and Georgia Koutrika. Fairness in rankings and recommendations: an overview. *The VLDB Journal*, pages 651–654, October 2021.

Steffen Rendle, Christoph Freudenthaler, Zeno Gantner, and Lars Schmidt-Thieme. Bpr: Bayesian personalized ranking from implicit feedback, 2012. Proceedings of the Twenty-Fifth Conference on Uncertainty in Artificial Intelligence (UAI2009).

Marco Tulio Ribeiro, Anisio Lacerda, Adriano Veloso, and Nivio Ziviani. Pareto-efficient Hybridization for Multi-objective Recommender Systems. In *ACM RecSys*, pages 19–26, 2012.

Marco Tulio Ribeiro, Nivio Ziviani, Edleno Silva De Moura, Itamar Hata, Anisio Lacerda, and Adriano Veloso. Multiobjective Pareto-Efficient Approaches for Recommender Systems. *ACM TIST*, pages 53:1–53:20, 2014.

Francesco Ricci, Lior Rokach, and Bracha Shapira. Introduction to Recommender Systems Handbook. In *Recommender Systems Handbook*, pages 1–35. Springer US, 2011. ISBN 978-0-387-85819-7 978-0-387-85820-3.

Mario Rodriguez, Christian Posse, and Ethan Zhang. Multiple Objective Optimization in Recommender Systems. In *ACM RecSys*, pages 11–18, New York, NY, USA, 2012.

Joseph Sill, Gabor Takacs, Lester Mackey, and David Lin. Feature-Weighted Linear Stacking. *arXiv:0911.0460 [cs]*, November 2009.

Daniel Xavier Sousa, Sérgio Canuto, Marcos André Gonçalves, Thierson Couto Rosa, and Wellington Santos Martins. Risk-sensitive learning to rank with evolutionary multi-objective feature selection. *ACM TOIS*, February 2019.

Daniel Xavier De Sousa, Sérgio Daniel Canuto, Thierson Couto Rosa, Wellington Santos Martins, and Marcos André Gonçalves. Incorporating risk-sensitiveness into feature selection for learning to rank. In *Proceedings of the 25th ACM CIKM*, page 257266, New York, NY, USA, 2016. ACM.

M. Srinivas and Lalit M. Patnaik. Genetic Algorithms: A Survey. *Journal of Computer*, 27:17–26, 1994.

Xiaoyuan Su and Taghi M. Khoshgoftaar. A Survey of Collaborative Filtering Techniques. *Advances in Artificial Intelligence*, 2009, October 2009.

Saúl Vargas and Pablo Castells. Rank and Relevance in Novelty and Diversity Metrics for Recommender Systems. In *ACM RecSys*, pages 109–116, 2011.

Feng Wang, Yixuan Li, Heng Zhang, Ting Hu, and Xiao-Liang Shen. An adaptive weight vector guided evolutionary algorithm for preference-based multi-objective optimization. *Swarm and Evolutionary Computation*, 49:220–233, September 2019.

Handing Wang, Markus Olhofer, and Yaochu Jin. A mini-review on preference modeling and articulation in multi-objective optimization: current status and challenges. *Complex & Intelligent Systems,*, vol. 3:pp. 1–13, August 2017.

Lidan Wang, Paul N. Bennett, and Kevyn Collins-Thompson. Robust ranking models via risk-sensitive optimization. In *Proceedings of the 35th ACM SIGIR*, page 761770, New York, NY, USA, 2012. ACM.

Shanfeng Wang, Maoguo Gong, Haoliang Li, and Junwei Yang. Multi-objective optimization for long tail recommendation. *Knowledge-Based Systems*, 104:145–155, July 2016.

Luc Wismans, T. Brands, Berkum Erik, and Michiel Bliemer. Pruning and ranking the Pareto optimal set, application for the dynamic multi-objective network design problem. *Journal of Advanced Transportation*, pages 512– 525, 2011.

Lin Xiao, Zhang Min, Zhang Yongfeng, Gu Zhaoquan, Liu Yiqun, and Ma Shaoping. Fairness-aware group recommendation with pareto-efficiency. In *Proceedings of the Eleventh ACM Conference on Recommender Systems*, RecSys '17, page 107115, 2017. ISBN 9781450346528.

Logan Yliniemi and Kagan Tumer. Multi-objective multiagent credit assignment in reinforcement learning and nsga-ii. *Soft Computing*, 20:3869–3887, 2016.

Peng Zhang, Linxue Hao, Dawei Song, Jun Wang, Yuexian Hou, and Bin Hu. Generalized bias-variance evaluation of trec participated systems. In *Proceedings of the 23rd ACM CIKM*, page 19111914, New York, NY, USA, 2014. ACM.

Cai-Nicolas Ziegler, Sean M. McNee, Joseph A. Konstan, and Georg Lausen. Improving Recommendation Lists Through Topic Diversification. In *WWW*, pages 22–32, 2005.

E. Zitzler and L. Thiele. Multiobjective evolutionary algorithms: a comparative case study and the strength pareto approach. *IEEE Transactions on Evolutionary Computation*, 3(4):257–271, 1999. doi: 10.1109/4235.797969.

Eckart Zitzler, Marco Laumanns, and Lothar Thiele. SPEA2: Improving the strength pareto evolutionary algorithm. *Proceedings of Evolutionary Methods for Design, Optimization and Control with Applications to Industrial Problems - EUROGEN*, 3242: 12–19, 2001.

Yi Zuo, Maoguo Gong, Jiulin Zeng, Lijia Ma, and Licheng Jiao. Personalized Recommendation Based on Evolutionary Multi-Objective Optimization [Research Frontier]. *IEEE CIM*, 10(1):52–62, 2015.

Appendix A

Detailing meta-features

This appendix details the calculation of meta-features explored in this thesis. Considering the input data $\mathcal{D}$ a *meta-feature* for each element e, which can be a user or an item, is computed as:

$$\mathrm{MF}_{\textsc{Agg},\delta,\sigma,e} = \textsc{Agg}\left(\delta\left(\sigma_e\left(\mathcal{D}\right)\right)\right), \tag{A.1}$$

where: σ_e is a selection function applied on the input data considering the element e; δ is a measure calculated for the selected subset of the input data; and Agg is an aggregation function, which might be *count*, *sum*, *average*, or *log*, when σ_e results in a set of values.

The *meta-features* explored in this work are listed in Table A.1, categorized into two classes: *Content* and *Rating*. Additionally, *Rating meta-features* are classified regarding to usage aspects in: *number of ratings* and *rating values.* Details are given in the following subsections.

A.1 Content-Based measures

In the Content-Based Filtering approach, the input contains items' attributes representing their content. Thus, the *meta-features* would be related to measures applied to the item content and similarities calculated for pairs of items.

In this work, the content-based measures are computed only for items. We evaluate different combinations of attributes for all content-based *meta-features* in preliminary experiments and due to the correlation between their values we used all attributes in the final experiments.

The measures exploited in this work are defined as follows:

1. **Cosine**: quantifies the similarity between two elements using the Cosine Similarity [Baeza-Yates and Ribeiro-Neto, 1999], defined as:

$$\mathrm{Cosine}(e_i, e_j) = \frac{V(e_i) \cdot V(e_j)}{|V(e_i)| * |V(e_j)|}, \tag{A.2}$$

Type	Acronym	Description	
I	Cosine	Agg	Average
		δ	Cosine similarity
		σ	Content from items
	Dice	Agg	Average
		δ	Dice coefficient
		σ	Content from items
	Jaccard	Agg	Average
		δ	Jaccard index
		σ	Content from items
	Entropy	Agg	No action, results in the δ value itself
		δ	Entropy measure
		σ	Content from items
II	PCR	Agg	Percentage of elements in common over the number of elements
		δ	Number of elements in common
		σ	Elements in common
	PR	Agg	Percentage of the number of ratings over the number of elements
		δ	Number of ratings
		σ	Ratings from element e
III	Gini	Agg	No action, results in the δ value itself
		δ	Gini index
		σ	Ratings from element e in ascending order
	Pearson	Agg	No action, results in the δ value itself
		δ	Pearson's Coefficient of Variation
		σ	Ratings from element e
	PqMean	Agg	No action, results in the δ value itself
		δ	*pq*-mean
		σ	Ratings from element e
	SD	Agg	No action, results in the δ value itself
		δ	Standard Deviation
		σ	Ratings from element e

Table A.1: *Meta-features*. Types: (I) Content; (II) Number of ratings; (III) Rating values. The mathematical definitions consist of the description for each component from Equation A.1.

where: $V(e)$ is the weighted *words* vector for the element e (the weights are computed using the *tf-idf* concept from Information Retrieval [Baeza-Yates and Ribeiro-Neto, 1999]); $V(e_i) \cdot V(e_j)$ is the dot product of the elements' vectors; and $|V(e)|$ is the Euclidean norm for the element e.

2. **Dice**: the Dice Coefficient is originally used by Adar et al. [2009] to compute the differences between two versions of the same *document* over time. We use it to compute the differences between the content from two different *elements* without loss of generality. The Dice Coefficient between two elements is defined as:

$$\text{Dice}(e_i, e_j) = 2 * \frac{|W(e_i) \cap W(e_j)|}{|W(e_i)| + |W(e_j)|}, \tag{A.3}$$

where: $W(e)$ is the set of *words* for the element e.

3. **Jaccard**: is another measure which quantifies similarities, as well as Cosine and Dice. The difference is the use of the Jaccard Index [Baeza-Yates and Ribeiro-Neto,

1999] as similarity measure. The Jaccard Index between two elements is computed considering the union of each element attributes' content and is defined as:

$$\text{Jaccard}(e_i, e_j) = \frac{|W(e_i) \cap W(e_j)|}{|W(e_i) \cup W(e_j)|}, \tag{A.4}$$

where: $W(e)$ is the set of *words* for the element e.

4. **Entropy**: quantifies the cohesiveness of the element's content via the Entropy measure [Bendersky et al., 2011]. Low values for Entropy indicate a tendency for the content to cover a single topic. The Entropy for an element is defined as:

$$\text{Entropy}(e_i) = -\sum_{w \in W(e_i)} p_D(w) * log\; p_D(w), \tag{A.5}$$

where: $W(e)$ is the set of *words* for the element e; and $p_D(w) = \frac{\text{tf}_{w_a}}{\sum_{w_b \in W(e_i)} \text{tf}_{w_b}}$ is a maximum likelihood estimation, where tf_w is the term frequency of word w in $W(e)$.

For *meta-features* based on similarity measures (*i.e.*, Cosine, Dice, and Jaccard), the target item e_i is compared to several other items e_j, selected as follows: (a) **All**: all other items; (b) **Below**: the similarity value is less than or equal to 0.5; and (c) **Above**: the similarity value is grater than or equal to 0.5.

A.2 Collaborative Filtering measures

In the Collaborative Filtering approach, the input is a *rating matrix* mapping the user satisfaction about items. The satisfaction can be expressed as a boolean value meaning *like* or *dislike*, or as a numeric value meaning the degree of *satisfaction* or *dissatisfaction.* Thus, the characteristics from these input data that can be extracted would be related to the number of ratings involved and the distribution of their values.

In this work, the collaborative filtering measures are computed for users and for items by considering their ratings.

The measures exploited in this work are defined as follows:

1. **Proportion of Common Ratings** (**PCR**): captures a notion of the size of the neighborhood via the concept of *users in common* and *items in common. Users in common* are those who ranked the same items that a particular user ranked and *items in common* are those that were ranked by the same users who ranked a particular item.

2. **Proportion of Ratings** (**PR**): captures a notion of the amount of ratings available via the percentage of the number of ratings given by a user or received by an item.

3. **Gini**: captures a notion of the rating values distribution using the Gini Index [Hurley and Rickard, 2009]. Gini measures the inequality of the values, when the result is zero it expresses a perfect equality, but when the result is one it expresses the maximal inequality. It is computed for an element, and is defined as:

$$\text{Gini}(e_i) = 1 - 2 * \sum_{k=1}^{n} \frac{c_k}{\|\overrightarrow{r_{e_i}}\|_1} * \left(\frac{n - k + 0.5}{n}\right), \tag{A.6}$$

   where: $\overrightarrow{r_{e_i}}$ is an array of the rating values for the element e_i in ascending order; $n = |\overrightarrow{r_{e_i}}|$ is the number of ratings in the array; c_k is the *k-th* element from the array; and $\|\overrightarrow{r_{e_i}}\|_1 = \sum_{j=1}^{n} c_j$.

4. **Pearson**: also captures the rating values distribution, but using the Pearson's Coefficient of Variation (CV) computed for an element, and is defined as:

$$\text{CV}(e_i) = \frac{S_{e_i}}{\overline{X_{e_i}}}, \tag{A.7}$$

   where: S_{e_i} is the standard deviation of the rating values; and $\overline{X_{e_i}}$ is the arithmetic mean of the rating values.

5. **PqMean**: is another measure which captures the rating values distribution, but using the *pq*-mean metric [Hurley and Rickard, 2009] computed for an element, and is defined as:

$$pq\text{-mean}(e_i) = -\left(\frac{1}{n}\sum_{k=1}^{n} c_k^p\right)^{1/p} \left(\frac{1}{n}\sum_{k=1}^{n} c_k^q\right)^{1/q}, \tag{A.8}$$

   where: n is the number of ratings for the element e_i; c_k is the *k-th* rating for the element e_i; and p and q are input parameters (from the Hurley and Rickard [2009] experimental results we configured with $p = 1$ and $q = 3$).

6. **SD**: also captures the rating values distribution, but using the Standard Deviation measure.

Appendix B

Detailing experimental strategy

This appendix details the experimental strategy, a k-folded cross-validation procedure composed of the process shown in Algorithm 3, described below.

We are dealing with hybrid methods, where the results of constituent algorithms compose the inputs to the hybridization algorithms. Therefore, it is necessary to deal with the combination of folds strategically, as described below.

The first task (line 1) performs the selection of the k folds applying a stratified random sampling based on the number of users' ratings. In this thesis, we use $k = 5$. For example, if a user has 100 ratings, each fold will be composed of 20 ratings chosen

Algorithm 3 Experimental Procedure

```
    Input: data, k=5, p=0.2
    Output: evaluation metrics values
 1: samples ← kFoldSampling(data, k)
 2: scores ← {}; meta ← {}
 3: for t in {1..k} do
 4:     for v in {1..k}-{t} do
 5:         trainSet ← samples[{1..k}-{t, v}]
 6:         predictionSet ← samples[v]
 7:         scores.add(RunConstituents('ALL', trainSet, predictionSet))
 8:         meta.add(CalcMetaFeatures(trainSet))
 9:     end for
10: end for
11: tunedConst ← SelectBestConstituents(scores)
12: tuningHF ← PrepareTuningHF(p, scores, meta, tunedConst)
13: hfAlgs ← RunTuningHF(tuningHF)
14: scores ← {}
15: for t in {1..k} do
16:     trainSet ← samples[{1..k}-{t}]
17:     predictionSet ← samples[t]
18:     constScores ← RunConstituents(tunedConst, trainSet, predictionSet)
19:     meta ← CalcMetaFeatures(trainSet)
20:     features ← FeaturesBuilding(constScores, meta)
21:     hfScores ← RunHF(hfAlgs, features)
22:     scores.add(constScores)
23:     scores.add(hfScores)
24: end for
25: return EvaluationAndVisualization(scores)
```

at random. That is, each fold is composed of a percentage of the ratings of all users, randomly selected. In this way, we maintain the proportionality of the real dataset. Note that we are not considering temporal aspects, leavening this characteristic for future work.

The second task (lines 2 to 11) performs the tuning of the *constituent algorithms* and prepares data for the subsequent tuning of the hybrid algorithms. To do this, we execute all *constituent algorithms* (line 7) and calculate all *meta-features* (line 8) for all 3×1 fold combinations, leaving a fold for the final evaluation tests for each combination. Then, the `SelectBestConstituents` function evaluates the predicted scores with the Root Mean Squared Error (RMSE) measure and selects the best configuration for each algorithm (line 11). Table B.1 presents the parameters and values used for tuning the *constituent algorithms*.

Algorithm	**Parameter**	**Values**
CB-Lucene	Content attributes	All (concatenated)
ALS-BiasedMF	Number of features	20, 40, 80, 100
BP-SlopeOne	-	-
Bias	-	-
Biased-MF	Number of factors	20, 40, 80, 100
Biased-SVD	Number of features	20, 40, 80, 100
BPR	Number of factors	32, 64, 128
	Learning rate	0.01, 0.1
Implicity-MF	Number of features	20, 40, 80, 100
ItemKNN	Max. number of neighbors	10, 25, 50, 100
NCF	Number of factors	64
	Negative items	4
SlopeOne	-	-
SVDPlusPlus	Number of factors	20, 40, 80, 100
UserKNN	Max. number of neighbors	10, 25, 50, 100

Table B.1: *Constituent algorithms* tuning configurations. For parameters with a single value, preliminary tests were performed to define the final value. In these cases, actions were needed to reduce the tuning time.

The third task (lines 12 and 13) performs the tuning of the WHF and MOF algorithms using the previously computed scores and *meta-features*. A single data set aggregates the features built for all fold combinations. A percentage p is randomly selected from this data and used for tuning the algorithms. In addition to these algorithms being usually more time-consuming, there are several testing scenarios, such as considering all features or selecting features, the three different hybridization approaches (STREAM, FWLS, and HR), and other situations. Therefore, this strategy reduces the number of executions needed, reducing the time spent on tuning. Table B.2 presents the parameters and values used for tuning the WHF and MOF algorithms. The regression methods, Ridge, Random Forest, and Gradient Boosting, are implementations of the Scikit-Learn library [Pedregosa et al., 2012]. We used the three regressions for the HR and STREAM hybrid methods, and for the FWLS method, we used only the Ridge. To tune the regression methods, we use the built-in `RandomizedSearchCV` function provided by Scikit-Learn. The evolutionary methods, PSO (Particle Swarm Optimization) and NSGA-II (Nondominated Sorting Genetic Algorithm II) are implementations of the JMetal framework [Durillo and Nebro, 2011; Nebro et al., 2015]. The crossover, mutation, and selection operators are built-in classes also provided by the Jmetal framework.

Learning method	**Parameter**	**Values**
Ridge	Alpha	Random uniform in [0.1, 5.0]
	Fit independent term	True, False
Random Forest	Number of trees	Random integer in [10, 500]
	Max. depth	Random integer in [1, 50]
	Min. number of samples to split	Random integer in [2, 50]
	Min. number of samples to leaf node	Random integer in [1, 50]
	Min. weighted fraction to leaf node	Random uniform in [0.0, 0.5]
	Max. leaf nodes	Random integer in [2, 50]
Gradient Boosting	Number of boosting stages	Random integer in [10, 500]
	Learning rate	Random uniform in [0.05, 2.0]
	Max. depth	Random integer in [1, 50]
	Min. number of samples to split	Random integer in [2, 50]
	Min. number of samples to leaf node	Random integer in [1, 50]
	Min. weighted fraction to leaf node	Random uniform in [0.0, 0.5]
	Max. leaf nodes	Random integer in [2, 50]
	Fraction of samples	Random uniform in [0.0, 0.9]
	Alpha	Random uniform in [0.0, 1.0]
PSO	Swarm sizes	60, 120
	Number of particles	20, 40, 80
NSGA-II	Population size	60, 120
	Crossover	SBX, Two point
	Mutation	Uniform, Polynomial
	Selection	Binary tournament

Table B.2: WHF and MOF tuning configurations.

The fourth task (lines 14 to 24) performs the predictions for all 4 × 1 fold combinations. The `RunContituents` function trains all the best configurations of *constituent algorithms* and generates their predictions (line 18), while the `CalcMetaFeatures` function calculates all *meta-features* (line 19). With this data, the `FeaturesBulding` function processes all the features (line 20). Then, the `RunHF` function trains all the best configurations of the hybrid algorithms and generates their predictions (line 21). The predictions are stored for later evaluation (lines 22 and 23).

Finally, the `EvaluationAndVisualization` function performs all evaluation measures calculations, statistical calculations over these evaluation measures, and generates visualization tables and graphs (line 25).

Appendix C

Complementary overall recommendation result tables

This appendix presents the mean values of the measures of all ranking results tables of the Sections 3.4.2, 4.3.2, and 5.3.2. The captions refer to the summary tables presented in the body of the thesis.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Rank-FWLS-Sel-SUM	0.8358±0.0014	**0.6798±0.0017**	**0.6537±0.0019**	**0.5993±0.0027**	**0.5328±0.0031**	**0.5134±0.0031**
1	MO-Rank-HR-All-SUM	0.8333±0.0014	**0.6811±0.0017**	**0.6542±0.0019**	**0.5996±0.0022**	**0.5347±0.0030**	**0.5149±0.0031**
1	MO-Rank-STREAM-All-SUM	0.8342±0.0014	**0.6778±0.0017**	**0.6525±0.0019**	**0.5990±0.0031**	**0.5314±0.0033**	**0.5123±0.0027**
4	Biased-MF	0.8354±0.0014	0.6771±0.0018	**0.6540±0.0019**	0.5823±0.0012	0.5101±0.0015	0.4905±0.0014
5	STREAM-All	**0.8545±0.0013**	0.6697±0.0018	0.6496±0.0020	0.5894±0.0021	0.5016±0.0024	0.4840±0.0024
7	HR-All	**0.8549±0.0013**	0.6682±0.0018	0.6474±0.0020	0.5896±0.0015	0.5016±0.0023	0.4830±0.0025
9	SO-Rank-STREAM-All	0.8153±0.0015	0.6679±0.0017	0.6414±0.0019	0.5806±0.0135	0.5155±0.0106	0.4954±0.0112
10	SO-Rank-FWLS-All	0.8102±0.0015	0.6657±0.0018	0.6396±0.0019	0.5751±0.0130	0.5107±0.0124	0.4906±0.0118
10	SO-Rank-HR-All	0.8087±0.0015	0.6662±0.0017	0.6398±0.0019	0.5767±0.0062	0.5133±0.0038	0.4931±0.0044
12	ALS	0.8432±0.0013	0.6540±0.0019	0.6344±0.0021	0.5797±0.0017	0.4852±0.0026	0.4661±0.0024

Table C.1: Mean values of the measures of Table 3.7.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Rank-HR-Sel-SUM	**0.7852±0.0049**	**0.5491±0.0044**	**0.5288±0.0051**	**0.5898±0.0022**	**0.4965±0.0012**	**0.4733±0.0018**
2	MO-Rank-HR-All-SUM	0.7840±0.0049	**0.5495±0.0044**	**0.5317±0.0051**	**0.5882±0.0034**	**0.4958±0.0018**	**0.4745±0.0021**
3	MO-Rank-STREAM-All-SUM	0.7807±0.0049	**0.5490±0.0044**	**0.5313±0.0051**	0.5855±0.0010	**0.4946±0.0024**	**0.4729±0.0036**
3	MO-Rank-FWLS-All-SUM	0.7795±0.0049	**0.5490±0.0045**	**0.5309±0.0052**	0.5851±0.0043	**0.4941±0.0036**	**0.4705±0.0042**
4	FWLS-All	**0.7943±0.0048**	**0.5481±0.0044**	**0.5249±0.0051**	**0.5859±0.0026**	0.4891±0.0020	0.4614±0.0033
6	HR-Sel	**0.7914±0.0049**	0.5424±0.0044	0.5177±0.0052	0.5811±0.0023	0.4819±0.0015	0.4510±0.0035
7	ALS	0.7611±0.0053	**0.5530±0.0046**	**0.5301±0.0054**	0.5483±0.0043	0.4818±0.0036	0.4515±0.0043
8	Biased-SVD	0.7604±0.0054	**0.5528±0.0046**	**0.5306±0.0054**	0.5473±0.0040	0.4815±0.0033	0.4512±0.0039
9	SO-Rank-FWLS-All	0.7657±0.0050	0.5381±0.0044	0.5110±0.0051	0.5622±0.0080	0.4761±0.0078	0.4451±0.0074
10	SO-Rank-HR-All	0.7712±0.0050	0.5324±0.0044	0.4957±0.0050	0.5669±0.0075	0.4745±0.0067	0.4375±0.0083
11	SO-Rank-STREAM-Sel	0.7612±0.0051	0.5329±0.0045	0.5034±0.0052	0.5553±0.0047	0.4697±0.0030	0.4360±0.0060

Table C.2: Mean values of the measures of Table 3.8.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	HR-All	**0.7192±0.0018**	**0.6714±0.0019**	**0.6478±0.0020**	**0.5325±0.0018**	**0.5207±0.0017**	**0.5036±0.0019**
1	MO-Rank-HR-All-SUM	**0.7172±0.0018**	**0.6700±0.0019**	**0.6472±0.0020**	**0.5316±0.0033**	**0.5200±0.0031**	**0.5032±0.0034**
2	MO-Rank-STREAM-Sel-SUM	0.7137±0.0018	**0.6689±0.0019**	**0.6474±0.0021**	**0.5312±0.0022**	**0.5201±0.0019**	**0.5034±0.0021**
3	FWLS-All	0.7098±0.0018	0.6634±0.0019	0.6386±0.0020	**0.5270±0.0052**	**0.5158±0.0049**	**0.4984±0.0056**
4	Biased-SVD	0.7084±0.0018	0.6640±0.0019	0.6415±0.0021	0.5207±0.0011	0.5105±0.0009	0.4943±0.0007
4	UserKNN	0.7083±0.0018	0.6639±0.0019	0.6399±0.0020	0.5205±0.0015	0.5107±0.0014	0.4939±0.0012
5	MO-Rank-FWLS-All-SUM	0.7003±0.0018	0.6589±0.0019	0.6364±0.0021	0.5218±0.0060	0.5121±0.0051	0.4953±0.0046
8	SO-Rank-HR-All	0.6638±0.0019	0.6259±0.0019	0.5949±0.0021	0.4908±0.0088	0.4838±0.0080	0.4631±0.0098
11	SO-Rank-FWLS-All	0.6455±0.0019	0.6116±0.0020	0.5808±0.0021	0.4750±0.0077	0.4702±0.0067	0.4476±0.0060
13	SO-Rank-STREAM-Sel	0.6357±0.0019	0.6007±0.0020	0.5644±0.0021	0.4673±0.0120	0.4621±0.0111	0.4378±0.0122

Table C.3: Mean values of the measures of Table 3.9.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Rank-HR-Sel-SUM	**0.7970±0.0014**	**0.5950±0.0011**	0.5000±0.0011	**0.5373±0.0024**	**0.4759±0.0014**	**0.4358±0.0007**
1	MO-Rank-STREAM-Sel-SUM	**0.7971±0.0014**	**0.5955±0.0011**	**0.5027±0.0012**	0.5371±0.0011	**0.4760±0.0007**	**0.4368±0.0006**
2	MO-Rank-HR-All-SUM	**0.7982±0.0014**	0.5927±0.0011	0.4967±0.0011	**0.5387±0.0018**	**0.4758±0.0014**	**0.4354±0.0022**
3	MO-Rank-FWLS-All-SUM	0.7904±0.0014	0.5932±0.0012	**0.5033±0.0012**	0.5334±0.0033	**0.4738±0.0020**	**0.4357±0.0015**
4	FWLS-All	**0.7995±0.0014**	0.5924±0.0011	0.4928±0.0011	**0.5391±0.0008**	**0.4760±0.0005**	0.4342±0.0007
4	HR-All	**0.7984±0.0014**	0.5927±0.0011	0.4935±0.0011	**0.5385±0.0014**	**0.4759±0.0009**	0.4344±0.0010
8	ItemKNN	0.7886±0.0014	0.5866±0.0012	0.4829±0.0011	0.5321±0.0009	0.4709±0.0005	0.4270±0.0005
10	Biased-SVD	0.7572±0.0015	0.5620±0.0012	0.4553±0.0012	0.5109±0.0011	0.4535±0.0008	0.4067±0.0006
13	SO-Rank-FWLS-All	0.7257±0.0016	0.5427±0.0013	0.4480±0.0012	0.4958±0.0062	0.4422±0.0032	0.3977±0.0022
14	SO-Rank-HR-Sel	0.7008±0.0016	0.5318±0.0013	0.4545±0.0013	0.4778±0.0220	0.4306±0.0152	0.3914±0.0110
14	SO-Rank-STREAM-Sel	0.7046±0.0016	0.5343±0.0013	0.4500±0.0013	0.4812±0.0165	0.4331±0.0103	0.3917±0.0076

Table C.4: Mean values of the measures of Table 3.10.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Risk-HR-All-SUM	0.8359±0.0014	**0.6811±0.0017**	**0.6551±0.0019**	**0.6052±0.0027**	**0.5385±0.0020**	**0.5192±0.0018**
1	MO-Rank-FWLS-Sel-SUM	0.8358±0.0014	**0.6798±0.0017**	**0.6537±0.0019**	**0.6029±0.0031**	**0.5351±0.0034**	**0.5158±0.0032**
1	MO-Risk-HR-All-Risk	0.8355±0.0014	**0.6809±0.0017**	**0.6549±0.0019**	**0.6039±0.0020**	**0.5372±0.0028**	**0.5181±0.0029**
1	MO-Risk-FWLS-Sel-SUM	0.8349±0.0014	**0.6778±0.0017**	**0.6519±0.0019**	**0.5998±0.0031**	**0.5311±0.0057**	**0.5118±0.0061**
2	MO-Rank-HR-All-SUM	0.8333±0.0014	**0.6811±0.0017**	**0.6542±0.0019**	**0.6017±0.0034**	**0.5355±0.0037**	**0.5160±0.0034**
4	Biased-MF	0.8354±0.0014	0.6771±0.0018	**0.6540±0.0019**	0.5872±0.0010	0.5141±0.0010	0.4945±0.0014
5	HR-All	**0.8549±0.0013**	0.6682±0.0018	0.6474±0.0020	0.5956±0.0014	0.5063±0.0021	0.4877±0.0022
5	STREAM-All	**0.8545±0.0013**	0.6697±0.0018	0.6496±0.0020	0.5955±0.0024	0.5064±0.0025	0.4888±0.0027
7	SO-Rank-STREAM-All	0.8153±0.0015	0.6679±0.0017	0.6414±0.0019	0.5818±0.0164	0.5167±0.0130	0.4968±0.0136
8	ALS	0.8432±0.0013	0.6540±0.0019	0.6344±0.0021	0.5854±0.0019	0.4890±0.0021	0.4703±0.0021
9	SO-Rank-HR-All	0.8087±0.0015	0.6662±0.0017	0.6398±0.0019	0.5761±0.0069	0.5130±0.0038	0.4931±0.0043
10	SO-Risk-HR-All	0.8070±0.0015	0.6639±0.0017	0.6375±0.0019	0.5748±0.0089	0.5111±0.0084	0.4909±0.0095
10	SO-Risk-STREAM-All	0.8089±0.0015	0.6637±0.0018	0.6374±0.0019	0.5758±0.0087	0.5098±0.0089	0.4896±0.0088

Table C.5: Mean values of the measures of Table 4.1.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Rank-HR-Sel-SUM	**0.7850±0.0049**	**0.5488±0.0044**	**0.5280±0.0051**	**0.5938±0.0016**	**0.4996±0.0009**	**0.4774±0.0010**
1	MO-Risk-HR-Sel-Risk	**0.7858±0.0049**	**0.5498±0.0044**	**0.5305±0.0051**	**0.5941±0.0030**	**0.4999±0.0017**	**0.4787±0.0019**
4	MO-Rank-STREAM-All-SUM	0.7802±0.0050	**0.5499±0.0045**	**0.5332±0.0051**	0.5834±0.0037	0.4932±0.0038	0.4718±0.0046
4	MO-Risk-STREAM-All-Risk	0.7812±0.0049	**0.5486±0.0044**	**0.5306±0.0051**	0.5859±0.0050	0.4939±0.0043	0.4704±0.0056
5	FWLS-All	**0.7943±0.0048**	**0.5481±0.0044**	**0.5249±0.0051**	0.5868±0.0034	0.4900±0.0024	0.4628±0.0034
8	HR-Sel	**0.7914±0.0049**	0.5424±0.0044	0.5177±0.0052	0.5824±0.0016	0.4829±0.0017	0.4525±0.0035
9	ALS	0.7611±0.0053	**0.5530±0.0046**	**0.5301±0.0054**	0.5515±0.0046	0.4847±0.0038	0.4553±0.0051
9	Biased-SVD	0.7604±0.0054	**0.5528±0.0046**	**0.5306±0.0054**	0.5503±0.0044	0.4843±0.0036	0.4549±0.0051
10	SO-Risk-FWLS-All	0.7679±0.0050	0.5393±0.0044	0.5092±0.0050	0.5632±0.0020	0.4774±0.0018	0.4460±0.0018
11	SO-Rank-FWLS-All	0.7704±0.0050	0.5400±0.0044	0.5073±0.0050	0.5659±0.0015	0.4787±0.0013	0.4459±0.0021
11	SO-Risk-HR-All	0.7721±0.0050	0.5372±0.0044	0.5038±0.0050	0.5668±0.0068	0.4769±0.0075	0.4437±0.0082
12	SO-Rank-HR-All	0.7698±0.0050	0.5340±0.0044	0.5003±0.0050	0.5620±0.0063	0.4728±0.0043	0.4370±0.0051

Table C.6: Mean values of the measures of Table 4.2.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	HR-All	**0.7192±0.0018**	**0.6714±0.0019**	**0.6478±0.0020**	**0.5427±0.0016**	**0.5295±0.0016**	**0.5136±0.0016**
2	MO-Rank-HR-All-SUM	0.7146±0.0018	**0.6680±0.0019**	**0.6447±0.0020**	**0.5431±0.0020**	**0.5299±0.0020**	**0.5135±0.0025**
2	MO-Risk-HR-All-SUM	0.7151±0.0018	**0.6685±0.0019**	**0.6457±0.0020**	**0.5440±0.0025**	**0.5308±0.0024**	**0.5145±0.0028**
3	MO-Rank-STREAM-Sel-SUM	0.7114±0.0018	**0.6678±0.0019**	**0.6470±0.0021**	**0.5410±0.0038**	**0.5289±0.0032**	**0.5137±0.0031**
4	MO-Risk-STREAM-Sel-SUM	0.7093±0.0018	0.6659±0.0019	**0.6447±0.0021**	**0.5404±0.0023**	**0.5282±0.0021**	**0.5126±0.0022**
5	FWLS-All	0.7098±0.0018	0.6634±0.0019	0.6386±0.0020	**0.5358±0.0059**	**0.5234±0.0054**	**0.5071±0.0060**
6	Biased-SVD	0.7084±0.0018	0.6640±0.0019	0.6415±0.0021	0.5284±0.0010	0.5173±0.0009	0.5021±0.0009
8	UserKNN	0.7083±0.0018	0.6639±0.0019	0.6399±0.0020	0.5260±0.0013	0.5154±0.0012	0.4999±0.0011
10	SO-Rank-HR-All	0.6577±0.0019	0.6219±0.0019	0.5933±0.0021	0.4848±0.0204	0.4785±0.0185	0.4589±0.0211
11	SO-Rank-FWLS-All	0.6447±0.0019	0.6121±0.0020	0.5853±0.0021	0.4713±0.0104	0.4665±0.0103	0.4470±0.0107
11	SO-Risk-HR-All	0.6513±0.0019	0.6142±0.0020	0.5792±0.0021	0.4776±0.0128	0.4715±0.0122	0.4502±0.0126
12	SO-Risk-FWLS-All	0.6318±0.0019	0.5995±0.0020	0.5698±0.0021	0.4605±0.0129	0.4556±0.0121	0.4346±0.0126

Table C.7: Mean values of the measures of Table 4.3.

#	Method	Ranking Measures			G_{RISK} Measures		
		NDCG	EPD	EILD	NDCG	EPD	EILD
1	MO-Risk-STREAM-Sel-SUM	0.7941±0.0014	**0.5955±0.0012**	**0.5052±0.0012**	**0.5530±0.0013**	**0.4881±0.0011**	**0.4482±0.0007**
2	MO-Rank-HR-Sel-SUM	**0.7977±0.0014**	**0.5955±0.0011**	0.5006±0.0011	**0.5557±0.0016**	**0.4895±0.0010**	**0.4479±0.0008**
3	MO-Risk-HR-Sel-SUM	0.7967±0.0014	**0.5956±0.0011**	0.5017±0.0012	**0.5549±0.0012**	**0.4890±0.0004**	**0.4478±0.0007**
3	MO-Rank-STREAM-Sel-SUM	0.7961±0.0014	**0.5957±0.0011**	0.5033±0.0012	**0.5543±0.0007**	**0.4888±0.0003**	**0.4481±0.0004**
6	FWLS-All	**0.7995±0.0014**	0.5924±0.0011	0.4928±0.0011	**0.5547±0.0006**	0.4872±0.0004	0.4431±0.0007
6	HR-All	**0.7984±0.0014**	0.5927±0.0011	0.4935±0.0011	**0.5536±0.0010**	0.4868±0.0007	0.4431±0.0009
7	ItemKNN	0.7886±0.0014	0.5866±0.0012	0.4829±0.0011	0.5469±0.0010	0.4816±0.0008	0.4355±0.0005
10	Biased-SVD	0.7572±0.0015	0.5620±0.0012	0.4553±0.0012	0.5203±0.0008	0.4601±0.0008	0.4109±0.0004
11	SO-Rank-FWLS-All	0.7337±0.0016	0.5469±0.0012	0.4473±0.0012	0.5073±0.0093	0.4506±0.0060	0.4025±0.0040
12	SO-Risk-FWLS-All	0.7298±0.0016	0.5441±0.0012	0.4450±0.0012	0.5040±0.0064	0.4478±0.0053	0.3999±0.0065
13	SO-Rank-HR-Sel	0.7128±0.0016	0.5339±0.0013	0.4467±0.0013	0.4927±0.0225	0.4399±0.0168	0.3957±0.0148
14	SO-Risk-HR-Sel	0.7060±0.0016	0.5332±0.0013	0.4502±0.0013	0.4874±0.0099	0.4368±0.0076	0.3939±0.0082

Table C.8: Mean values of the measures of Table 4.4.

#	Method	Distance	Ranking measures			G_{RISK} measures		
			NDCG	EPD	EILD	NDCG	EPD	EILD
				All users				
1	PrefMO-FWLS-Sel-IndDIST	**0.1148±0.0011**	0.8355±0.0012	**0.6672±0.0015**	**0.6396±0.0017**	**0.5920±0.0338**	**0.5137±0.0150**	**0.4922±0.0165**
1	PrefMO-HR-All-SUM	**0.1149±0.0012**	0.8334±0.0013	**0.6651±0.0017**	**0.6368±0.0018**	**0.5942±0.0364**	**0.5176±0.0140**	**0.4949±0.0162**
5	MO-Rank-HR-All-SUM	**0.1145±0.0011**	0.8312±0.0012	**0.6644±0.0015**	0.6357±0.0017	**0.5911±0.0340**	**0.5135±0.0134**	**0.4908±0.0152**
10	Biased-MF	0.1188±0.0011	0.8329±0.0012	0.6601±0.0015	0.6350±0.0017	**0.5750±0.0380**	0.4885±0.0141	**0.4664±0.0158**
12	SO-Rank-STREAM-All	0.1168±0.0011	0.8146±0.0013	0.6520±0.0015	0.6237±0.0017	**0.5705±0.0422**	**0.4914±0.0194**	**0.4688±0.0214**
14	HR-All	0.1312±0.0012	**0.8515±0.0011**	0.6494±0.0016	0.6266±0.0017	**0.5840±0.0342**	0.4752±0.0105	0.4544±0.0114
14	STREAM-All	0.1293±0.0011	**0.8509±0.0011**	0.6513±0.0016	0.6294±0.0017	**0.5840±0.0339**	0.4766±0.0101	0.4575±0.0109
15	PrefSO-STREAM-Sel	**0.1148±0.0011**	0.8033±0.0013	0.6486±0.0015	0.6192±0.0017	**0.5608±0.0444**	0.4843±0.0200	**0.4614±0.0223**
				Tolerant users				
1	PrefMO-HR-All-SUM	**0.1864±0.0042**	**0.9713±0.0013**	**0.6420±0.0051**	**0.6083±0.0055**	**0.6638±0.0009**	**0.5266±0.0023**	**0.5051±0.0030**
3	MO-Rank-HR-All-SUM	**0.1847±0.0038**	0.9701±0.0012	**0.6440±0.0045**	**0.6106±0.0049**	**0.6648±0.0030**	0.5248±0.0049	**0.5034±0.0049**
7	SO-Rank-HR-All	**0.1802±0.0037**	0.9646±0.0013	**0.6458±0.0045**	**0.6142±0.0049**	**0.6598±0.0027**	0.5125±0.0038	0.4925±0.0036
7	SO-Rank-STREAM-All	**0.1845±0.0038**	0.9659±0.0013	**0.6413±0.0045**	**0.6095±0.0049**	**0.6609±0.0026**	0.5128±0.0040	0.4930±0.0038
9	PrefSO-HR-All	**0.1830±0.0038**	0.9630±0.0014	**0.6413±0.0045**	**0.6108±0.0049**	0.6593±0.0030	0.5086±0.0109	0.4891±0.0098
9	PrefSO-STREAM-Sel	**0.1796±0.0038**	0.9629±0.0013	**0.6470±0.0045**	**0.6148±0.0049**	0.6594±0.0023	0.5118±0.0053	0.4926±0.0045
10	Biased-MF	0.1880±0.0038	0.9690±0.0012	**0.6380±0.0046**	**0.6078±0.0050**	0.6588±0.0024	0.5023±0.0023	0.4817±0.0032
12	HR-All	0.2205±0.0041	**0.9733±0.0011**	0.5945±0.0048	0.5656±0.0053	0.6596±0.0024	0.4608±0.0044	0.4391±0.0051
12	STREAM-All	0.2155±0.0040	**0.9726±0.0011**	0.6003±0.0048	0.5734±0.0052	0.6590±0.0027	0.4634±0.0034	0.4451±0.0054
				High accuracy users				
1	PrefMO-FWLS-Sel-SUM	**0.1105±0.0023**	0.6747±0.0034	0.5899±0.0039	0.5605±0.0044	**0.5126±0.0050**	**0.4792±0.0042**	**0.4530±0.0030**
3	HR-All	**0.1116±0.0024**	**0.6949±0.0034**	**0.5975±0.0039**	**0.5711±0.0045**	0.5005±0.0032	0.4616±0.0027	0.4392±0.0037
3	STREAM-All	**0.1096±0.0023**	**0.6939±0.0034**	**0.5982±0.0039**	**0.5728±0.0044**	0.5013±0.0042	0.4630±0.0039	0.4413±0.0039
4	MO-Rank-FWLS-Sel-SUM	**0.1095±0.0022**	0.6640±0.0035	0.5851±0.0038	0.5527±0.0044	**0.5093±0.0079**	**0.4792±0.0055**	0.4503±0.0061
11	Biased-MF	0.1141±0.0024	0.6629±0.0035	0.5812±0.0039	0.5515±0.0045	0.4821±0.0026	0.4510±0.0031	0.4244±0.0035
14	SO-Rank-STREAM-All	0.1143±0.0023	0.6325±0.0036	0.5631±0.0038	0.5297±0.0044	0.4706±0.0302	0.4453±0.0264	0.4171±0.0274
16	PrefSO-HR-All	0.1147±0.0023	0.6178±0.0036	0.5515±0.0038	0.5186±0.0044	0.4572±0.0220	0.4341±0.0199	0.4059±0.0203

Table C.9: Mean values of the measures of Table 5.1.

#	Method	Distance	Ranking measures			G_{RISK} measures		
			NDCG	EPD	EILD	NDCG	EPD	EILD
All users								
1	PrefMO-HR-Sel-SUM	**0.0972±0.0022**	0.7900±0.0039	**0.5493±0.0037**	**0.5290±0.0043**	**0.5809±0.0315**	**0.4848±0.0240**	**0.4626±0.0256**
1	MO-Rank-STREAM-All-SUM	**0.0958±0.0022**	0.7878±0.0039	**0.5494±0.0037**	**0.5302±0.0043**	**0.5747±0.0327**	**0.4806±0.0246**	**0.4582±0.0258**
1	PrefMO-STREAM-All-IndSUM	**0.0953±0.0022**	0.7865±0.0040	**0.5501±0.0037**	**0.5321±0.0043**	**0.5695±0.0319**	**0.4774±0.0238**	**0.4567±0.0247**
1	PrefMO-STREAM-All-SUM	**0.0950±0.0022**	0.7862±0.0039	**0.5499±0.0037**	**0.5314±0.0043**	**0.5747±0.0326**	**0.4816±0.0246**	**0.4598±0.0260**
1	PrefMO-St-STREAM-All-IndSUM	**0.0956±0.0022**	0.7861±0.0040	**0.5492±0.0037**	**0.5293±0.0043**	**0.5671±0.0321**	**0.4764±0.0243**	**0.4531±0.0253**
3	Biased-SVD	**0.0977±0.0022**	0.7684±0.0043	**0.5559±0.0037**	**0.5317±0.0046**	**0.5285±0.0278**	**0.4672±0.0265**	**0.4359±0.0274**
6	FWLS-All	0.1022±0.0023	**0.7990±0.0038**	0.5462±0.0037	0.5213±0.0043	**0.5745±0.0300**	**0.4735±0.0205**	**0.4431±0.0208**
10	SO-Rank-FWLS-All	0.1002±0.0022	0.7712±0.0041	0.5373±0.0037	0.5089±0.0043	**0.5507±0.0356**	**0.4614±0.0256**	**0.4298±0.0254**
11	PrefSO-FWLS-All	0.1023±0.0022	0.7773±0.0041	0.5383±0.0037	0.5042±0.0043	**0.5543±0.0347**	**0.4636±0.0253**	**0.4292±0.0244**
Tolerant users								
1	MO-Rank-HR-Sel-SUM	**0.1188±0.0067**	**0.8979±0.0060**	**0.6086±0.0089**	**0.5889±0.0103**	**0.6472±0.0023**	**0.5319±0.0022**	**0.5105±0.0030**
1	PrefMO-HR-Sel-SUM	**0.1176±0.0067**	**0.8968±0.0061**	**0.6087±0.0089**	**0.5914±0.0102**	**0.6470±0.0021**	**0.5321±0.0019**	**0.5125±0.0030**
1	MO-Rank-STREAM-All-SUM	**0.1170±0.0067**	**0.8960±0.0060**	**0.6100±0.0089**	**0.5930±0.0103**	**0.6446±0.0025**	**0.5296±0.0026**	**0.5089±0.0045**
1	PrefMO-STREAM-All-SUM	**0.1155±0.0066**	**0.8965±0.0060**	**0.6112±0.0088**	**0.5959±0.0101**	**0.6440±0.0019**	**0.5304±0.0029**	**0.5112±0.0045**
1	PrefMO-St-STREAM-All-SUM	**0.1170±0.0067**	**0.8960±0.0060**	**0.6102±0.0088**	**0.5917±0.0103**	**0.6432±0.0035**	**0.5297±0.0039**	**0.5078±0.0055**
7	ALS	**0.1104±0.0059**	0.8655±0.0083	**0.6219±0.0084**	**0.5967±0.0106**	0.5800±0.0104	0.5181±0.0060	0.4849±0.0114
7	Biased-SVD	**0.1082±0.0057**	0.8669±0.0083	**0.6242±0.0083**	**0.6019±0.0105**	0.5806±0.0102	0.5195±0.0051	0.4883±0.0106
9	FWLS-All	0.1300±0.0077	**0.9028±0.0058**	0.5985±0.0094	0.5772±0.0111	0.6380±0.0039	0.5116±0.0028	0.4793±0.0049
10	SO-Rank-HR-All	0.1321±0.0072	**0.8923±0.0066**	0.5932±0.0090	0.5552±0.0105	0.6306±0.0057	0.5083±0.0057	0.4671±0.0048
11	PrefSO-FWLS-All	0.1242±0.0067	0.8909±0.0062	0.6013±0.0088	0.5673±0.0103	0.6291±0.0080	0.5148±0.0022	0.4774±0.0056
High accuracy users								
1	MO-Rank-HR-Sel-SUM	**0.1082±0.0066**	**0.6499±0.0125**	**0.4439±0.0105**	**0.4148±0.0126**	**0.5036±0.0068**	**0.4242±0.0043**	**0.3967±0.0032**
1	PrefMO-HR-Sel-SUM	**0.1072±0.0066**	**0.6517±0.0123**	**0.4458±0.0105**	**0.4178±0.0126**	**0.5018±0.0053**	**0.4228±0.0031**	**0.3966±0.0046**
1	PrefMO-St-HR-Sel-SUM	**0.1086±0.0065**	**0.6533±0.0124**	**0.4444±0.0105**	**0.4135±0.0125**	**0.5064±0.0031**	**0.4253±0.0010**	**0.3970±0.0065**
2	FWLS-All	**0.1072±0.0064**	**0.6669±0.0120**	**0.4487±0.0102**	**0.4125±0.0123**	**0.4998±0.0073**	0.4201±0.0048	**0.3885±0.0071**
7	ALS	**0.1129±0.0073**	0.6388±0.0126	**0.4508±0.0107**	**0.4181±0.0133**	0.4592±0.0077	0.4013±0.0051	0.3691±0.0078
7	Biased-SVD	**0.1132±0.0073**	0.6349±0.0127	**0.4472±0.0107**	**0.4135±0.0133**	0.4566±0.0066	0.3990±0.0040	0.3658±0.0082
8	SO-Rank-FWLS-All	**0.1102±0.0067**	0.6279±0.0122	**0.4314±0.0102**	0.3951±0.0122	0.4646±0.0164	0.3975±0.0131	0.3661±0.0114
8	PrefSO-FWLS-All	**0.1103±0.0067**	0.6365±0.0124	**0.4328±0.0102**	0.3934±0.0121	0.4687±0.0056	0.3989±0.0042	0.3668±0.0045

Table C.10: Mean values of the measures of Table 5.2.

#	Method	Distance	Ranking measures			G_{RISK} measures		
			NDCG	EPD	EILD	NDCG	EPD	EILD
				All users				
1	HR-All	**0.2213±0.0011**	**0.7457±0.0013**	**0.6945±0.0014**	**0.6692±0.0016**	**0.5344±0.0188**	**0.5197±0.0219**	**0.5030±0.0252**
1	MO-Rank-HR-All-SUM	**0.2212±0.0011**	**0.7439±0.0013**	**0.6933±0.0014**	**0.6688±0.0016**	**0.5358±0.0193**	**0.5210±0.0223**	**0.5039±0.0260**
1	PrefMO-HR-All-IndDIST	**0.2208±0.0011**	**0.7435±0.0013**	**0.6936±0.0014**	**0.6698±0.0016**	**0.5324±0.0201**	**0.5182±0.0229**	**0.5017±0.0264**
4	Biased-SVD	**0.2213±0.0011**	0.7360±0.0013	0.6883±0.0014	0.6640±0.0016	**0.5198±0.0215**	**0.5073±0.0240**	**0.4917±0.0275**
10	SO-Rank-HR-All	0.2244±0.0011	0.6960±0.0014	0.6551±0.0015	0.6217±0.0017	0.4829±0.0271	**0.4752±0.0290**	**0.4554±0.0316**
11	PrefSO-HR-All	0.2256±0.0011	0.6891±0.0014	0.6482±0.0015	0.6120±0.0017	0.4773±0.0276	**0.4699±0.0296**	**0.4487±0.0325**
				Tolerant users				
1	HR-All	**0.4168±0.0027**	**0.6574±0.0037**	**0.5783±0.0040**	**0.5394±0.0048**	**0.4883±0.0015**	**0.4658±0.0019**	**0.4409±0.0024**
1	MO-Rank-HR-All-SUM	**0.4176±0.0027**	**0.6550±0.0037**	**0.5768±0.0040**	**0.5379±0.0048**	**0.4889±0.0063**	**0.4664±0.0060**	**0.4400±0.0058**
1	PrefMO-HR-All-IndDIST	**0.4176±0.0027**	**0.6550±0.0037**	**0.5773±0.0040**	**0.5389±0.0049**	**0.4836±0.0055**	**0.4622±0.0054**	**0.4367±0.0044**
1	PrefMO-HR-All-SUM	**0.4172±0.0027**	**0.6533±0.0037**	**0.5756±0.0040**	**0.5366±0.0048**	**0.4871±0.0031**	**0.4648±0.0031**	**0.4388±0.0024**
1	PrefMO-St-HR-All-IndDIST	**0.4175±0.0027**	**0.6521±0.0037**	**0.5748±0.0040**	**0.5358±0.0048**	**0.4845±0.0031**	**0.4627±0.0022**	**0.4365±0.0020**
1	PrefMO-St-HR-All-SUM	**0.4178±0.0027**	**0.6531±0.0037**	**0.5752±0.0040**	**0.5354±0.0048**	**0.4870±0.0018**	**0.4647±0.0021**	**0.4380±0.0018**
1	MO-Rank-STREAM-Sel-SUM	**0.4174±0.0027**	**0.6507±0.0037**	**0.5747±0.0040**	**0.5363±0.0049**	**0.4868±0.0039**	**0.4648±0.0036**	**0.4383±0.0030**
4	UserKNN	**0.4171±0.0027**	0.6475±0.0037	**0.5717±0.0040**	**0.5317±0.0048**	0.4644±0.0007	0.4461±0.0007	0.4230±0.0020
9	SO-Rank-HR-All	**0.4218±0.0027**	0.6001±0.0038	0.5343±0.0040	0.4871±0.0048	0.4194±0.0169	0.4064±0.0150	0.3805±0.0161
10	PrefSO-HR-All	0.4248±0.0028	0.5897±0.0039	0.5247±0.0040	0.4748±0.0048	0.4129±0.0166	0.4000±0.0155	0.3720±0.0177
				High accuracy users				
1	HR-All	**0.1008±0.0012**	**0.8208±0.0028**	**0.7882±0.0030**	**0.7713±0.0032**	**0.5759±0.0023**	**0.5673±0.0025**	**0.5580±0.0030**
1	MO-Rank-HR-All-SUM	**0.1010±0.0012**	**0.8194±0.0028**	**0.7873±0.0030**	**0.7716±0.0032**	**0.5781±0.0023**	**0.5694±0.0024**	**0.5604±0.0026**
1	PrefMO-HR-All-IndDIST	**0.1006±0.0012**	**0.8181±0.0028**	**0.7863±0.0030**	**0.7699±0.0032**	**0.5768±0.0017**	**0.5682±0.0018**	**0.5593±0.0022**
1	PrefMO-HR-All-SUM	**0.1009±0.0012**	**0.8182±0.0028**	**0.7867±0.0030**	**0.7705±0.0032**	**0.5763±0.0028**	**0.5679±0.0027**	**0.5592±0.0029**
1	PrefMO-St-HR-All-IndDIST	**0.1009±0.0012**	**0.8174±0.0028**	**0.7861±0.0030**	**0.7701±0.0033**	**0.5754±0.0036**	**0.5668±0.0035**	**0.5581±0.0033**
1	PrefMO-St-HR-All-SUM	**0.1009±0.0012**	**0.8184±0.0028**	**0.7870±0.0030**	**0.7713±0.0032**	**0.5755±0.0036**	**0.5670±0.0034**	**0.5585±0.0033**
1	MO-Rank-STREAM-Sel-SUM	**0.1023±0.0012**	**0.8171±0.0028**	**0.7880±0.0030**	**0.7743±0.0033**	**0.5763±0.0024**	**0.5684±0.0025**	**0.5600±0.0031**
1	PrefMO-STREAM-Sel-IndDIST	**0.1022±0.0012**	**0.8156±0.0028**	**0.7864±0.0030**	**0.7717±0.0033**	**0.5762±0.0026**	**0.5683±0.0025**	**0.5594±0.0034**
3	Biased-SVD	**0.1021±0.0012**	0.8133±0.0028	**0.7840±0.0030**	**0.7696±0.0033**	0.5672±0.0019	0.5601±0.0019	0.5522±0.0017
9	SO-Rank-HR-All	0.1049±0.0013	0.7772±0.0032	0.7535±0.0033	0.7288±0.0036	0.5419±0.0069	0.5389±0.0059	0.5245±0.0079
10	PrefSO-HR-All	0.1041±0.0013	0.7726±0.0032	0.7483±0.0033	0.7204±0.0035	0.5375±0.0085	0.5350±0.0068	0.5197±0.0091

Table C.11: Mean values of the measures of Table 5.3.

#	Method	Distance	Ranking measures			G_{RISK} measures		
			NDCG	EPD	EILD	NDCG	EPD	EILD
				All users				
1	MO-Rank-STREAM-Sel-SUM	0.0764±0.0004	0.8020±0.0011	**0.5976±0.0009**	0.5038±0.0010	**0.5346±0.0124**	**0.4729±0.0110**	**0.4338±0.0165**
3	PrefMO-HR-Sel-SUM	0.0774±0.0004	**0.8025±0.0011**	**0.5971±0.0009**	0.5006±0.0010	**0.5353±0.0123**	**0.4732±0.0107**	**0.4329±0.0162**
4	PrefMO-HR-Sel-IndDist	**0.0735±0.0004**	0.7844±0.0011	0.5937±0.0010	**0.5096±0.0010**	**0.5243±0.0117**	**0.4670±0.0111**	**0.4300±0.0173**
7	HR-All	0.0781±0.0004	**0.8033±0.0011**	0.5947±0.0009	0.4946±0.0009	**0.5314±0.0132**	**0.4694±0.0113**	**0.4278±0.0163**
13	ItemKNN	0.0806±0.0004	0.7939±0.0011	0.5889±0.0010	0.4842±0.0009	**0.5258±0.0139**	**0.4650±0.0119**	**0.4213±0.0167**
19	PrefSO-FWLS-Sel	0.0799±0.0005	0.7247±0.0013	0.5460±0.0010	0.4644±0.0011	0.4793±0.0255	0.4302±0.0214	0.3906±0.0266
20	SO-Rank-FWLS-All	0.0847±0.0005	0.7337±0.0013	0.5473±0.0010	0.4511±0.0011	0.4836±0.0237	0.4320±0.0200	0.3873±0.0256
20	SO-Rank-HR-Sel	0.0824±0.0005	0.7119±0.0013	0.5388±0.0011	0.4598±0.0011	0.4687±0.0291	0.4224±0.0239	0.3832±0.0283
				Tolerant users				
1	MO-Rank-STREAM-Sel-SUM	0.0754±0.0009	**0.8795±0.0021**	**0.6515±0.0018**	0.5763±0.0020	**0.5620±0.0014**	**0.4964±0.0010**	**0.4697±0.0016**
2	PrefMO-STREAM-Sel-SUM	0.0748±0.0009	**0.8771±0.0022**	**0.6504±0.0019**	0.5769±0.0021	0.5608±0.0007	**0.4957±0.0008**	**0.4693±0.0018**
5	PrefMO-HR-Sel-IndDist	**0.0669±0.0009**	0.8626±0.0023	**0.6530±0.0020**	**0.5906±0.0022**	0.5517±0.0028	0.4923±0.0017	**0.4690±0.0016**
10	FWLS-All	0.0809±0.0009	**0.8812±0.0021**	0.6464±0.0018	0.5619±0.0020	**0.5614±0.0008**	0.4940±0.0007	0.4631±0.0007
15	ItemKNN	0.0820±0.0009	0.8718±0.0022	0.6422±0.0019	0.5528±0.0020	0.5564±0.0009	0.4908±0.0009	0.4578±0.0008
16	PrefSO-FWLS-Sel	0.0712±0.0010	0.8271±0.0026	0.6211±0.0021	0.5564±0.0022	0.5324±0.0118	0.4750±0.0073	0.4484±0.0056
17	SO-Rank-HR-Sel	0.0725±0.0010	0.8218±0.0026	0.6175±0.0021	0.5526±0.0023	0.5292±0.0113	0.4726±0.0067	0.4449±0.0036
				High accuracy users				
1	MO-Rank-HR-Sel-SUM	**0.0981±0.0014**	**0.7195±0.0034**	**0.5319±0.0030**	**0.4159±0.0028**	**0.5043±0.0025**	**0.4457±0.0022**	**0.3918±0.0019**
1	PrefMO-HR-Sel-SUM	**0.0980±0.0014**	**0.7202±0.0034**	**0.5323±0.0029**	**0.4160±0.0028**	**0.5054±0.0027**	**0.4466±0.0024**	**0.3928±0.0030**
1	MO-Rank-STREAM-Sel-SUM	**0.0976±0.0013**	**0.7198±0.0034**	**0.5322±0.0029**	**0.4179±0.0028**	**0.5042±0.0027**	**0.4457±0.0025**	**0.3929±0.0024**
3	FWLS-All	**0.0961±0.0013**	**0.7232±0.0034**	**0.5315±0.0029**	**0.4135±0.0028**	0.5006±0.0011	0.4422±0.0008	0.3881±0.0011
3	HR-All	**0.0960±0.0013**	**0.7216±0.0034**	**0.5308±0.0029**	**0.4131±0.0027**	0.4992±0.0024	0.4413±0.0013	0.3875±0.0019
9	ItemKNN	0.0988±0.0014	0.7112±0.0034	0.5241±0.0030	0.4029±0.0027	0.4920±0.0022	0.4355±0.0012	0.3801±0.0015
13	PrefSO-FWLS-Sel	0.1093±0.0016	0.6187±0.0037	0.4612±0.0031	0.3600±0.0030	0.4226±0.0237	0.3814±0.0186	0.3283±0.0170
14	SO-Rank-FWLS-All	0.1117±0.0016	0.6239±0.0037	0.4617±0.0031	0.3478±0.0029	0.4260±0.0069	0.3831±0.0040	0.3247±0.0022

Table C.12: Mean values of the measures of Table 5.4.