UNIVERSIDADE DE LISBOA

FACULDADE DE CIÊNCIAS

DEPARTAMENTO DE INFORMÁTICA

Development of a recommender system based on life and health sciences literature

Maria Teresa Hipólito da Cunha

Mestrado em Bioinformática e Biologia Computacional

Dissertação orientada por:
Prof. Doutora Márcia Cristina Afonso Barros

2022

Resumo

Os sistemas de recomendação têm evoluído rapidamente e transformado o nosso dia-a-dia ao usar grandes quantidades de informação para obter recomendações personalizadas em áreas como música, filmes ou vendas online. No entanto, nas ciências da vida e da saúde, apesar da necessidade de novas formas de explorar a crescente quantidade de informação digital, há um obstáculo que tem impedido esta evolução: a privacidade dos dados. É preciso ter acesso às preferências dos utilizadores para testar e evoluir os sistemas de recomendação em saúde.

O objetivo deste trabalho é criar um conjunto de dados de acesso aberto com preferências de utilizadores obtidas implicitamente a partir de literatura das ciências da vida e da saúde, e testá-lo utilizando sistemas de recomendação de filtragem colaborativa.

Utilizando a metodologia LIBRETTI, criámos um conjunto de dados (DisRM) a partir de artigos científicos do PubMed. O DisRM está no formato <*utilizador*, *item*, *classificação*> onde os utilizadores são autores de artigos e os itens são doenças, tendo um total de 2 309 190 classificações. Foram criados dois conjuntos de dados adicionais, DisRM10 e DisRM20, que incluem apenas os utilizadores que têm um número de classificações igual ou superior a 10 e 20, respetivamente. Ao aplicar um algoritmo de filtragem colaborativa k-vizinhos mais próximos baseado em memória aos conjuntos de dados DisRM10 e DisRM20, o objetivo era otimizar o *recall* e o ganho cumulativo com desconto normalizado (nDCG) para garantir que a maioria dos itens relevantes eram recomendados e apareciam primeiro na lista de recomendações. Os melhores resultados de recomendações foram alcançados utilizando a medida de similaridade PIP, obtendo um *recall* de 0.81 e um nDCG de 0.87 para o DisRM10. Comparando o DisRM com outros conjuntos de dados padronizados, este obteve resultados semelhantes ou melhores o que valida a qualidade do nosso conjunto de dados.

Palavras Chave: Sistemas de Recomendação, Saúde, Conjunto de Dados, Preferências Implícitas, Filtragem Colaborativa

Abstract

Recommender systems are quickly evolving and transforming our daily life by being used to explore large amounts of information and delivering personalized recommendations on several areas like streaming and e-commerce. But in the life and health sciences field, although there is a growing need of new ways to explore information due to the increase of digital information, there is one major issue that is preventing its evolution: the privacy of data. It is necessary to have data about users' preferences to test and evolve health recommender systems.

The main objective of this work is to create an open-source implicit feedback dataset based on life and health sciences literature and test it using a collaborative filtering recommender system.

Using the LIBRETTI methodology, we created the dataset, called DisRM, using research articles from PubMed. The dataset is in the format <*user*, *item*, *rating*> where the users are authors of research articles and the items are diseases, and it has 2 309 190 ratings. Two additional datasets were created, DisRM10 and DisRM20, including only the users who have a number of ratings equal to or greater than 10 and 20, respectively. When applying a memory-based CF K-Nearest Neighbors algorithm to DisRM10 and DisRM20 we had the goal of optimizing the recall and the normalized discounted cumulative gain (nDCG), to ensure that most of the relevant items are being recommended and ranked high. We achieved the best recommendation results using the similarity measure PIP, obtaining a recall of 0.81 and a nDCG of 0.87 for DisRM10. When comparing DisRM with other baseline datasets, it performed similarly or better for recall and nDCG. This validates the quality of our dataset.

Keywords: Recommender Systems, Health, Dataset, Implicit Feedback, Collaborative Filtering

Resumo Alargado

Os sistemas de recomendação são ferramentas que fornecem recomendações personalizadas de itens aos seus utilizadores com base nas suas preferências. O primeiro sistema de recomendação surgiu em 1992 com o objetivo de filtrar de forma personalizada o elevado número de mensagens de correio eletrónico recebidas. Estes sistemas rapidamente se estenderam a diferentes áreas que precisavam de explorar a grande quantidade de informação que existe online, de forma a alcançar melhores resultados do que os obtidos com o recurso aos tradicionais métodos de recuperação de informação. Atualmente, existem inúmeros exemplos da aplicação de sistemas de recomendação no nosso dia-a-dia tais como a recomendação de música no Spotify, de filmes na Netflix ou de produtos para comprar online na Amazon.

Também nas áreas das ciências da vida e da saúde e da medicina se tem assistido a um aumento da quantidade de dados digitais. Por um lado, existe cada vez mais investigação e publicações de saúde que beneficiariam muito da utilização de novas formas de correlacionar a informação existente. Por outro lado, existe também um aumento no número de ficheiros de saúde de utentes que se encontram disponíveis digitalmente e que poderiam ser usados tanto para saúde personalizada, como também para o desenvolvimento da investigação em saúde. Isto evidencia a necessidade de criação de novas formas de explorar esta informação para além dos tradicionais métodos de recuperação de informação. Os sistemas de recomendação em saúde surgiram em 2007 com esse mesmo propósito mas o seu estudo ainda é muito recente e pouco desenvolvido, por isso, o seu uso ainda é limitado.

Para o desenvolvimento e avaliação de sistemas de recomendação é necessária a existência de dados relativos às preferências dos utilizadores mas, apesar do aumento da digitalização de ficheiros de saúde de utentes, a grande maioria desta informação é privada e difícil de aceder. Isto dificulta o processo de desenvolvimento de bons sistemas de recomendação em saúde a menos que se tenha acesso a dados sobre as preferências dos utilizadores. Este é um grande problema nesta área que tem levado ao subdesenvolvimento de sistemas de recomendação em saúde.

O objetivo deste trabalho é a criação de um conjunto de dados de acesso aberto com preferências de utilizadores obtidas implicitamente a partir de literatura das ciências da vida e da saúde, validado com recurso a testes utilizando sistemas de recomendação de filtragem colaborativa. Desta forma, este trabalho fornece duas contribuições: um conjunto de dados de acesso aberto na área das ciências da saúde e um sistema de recomendação para recomendações das ciências da vida e da saúde. O conjunto de dados e o código utilizado para a sua criação estão disponíveis no GitHub: `https://github.com/teresacunha/DisRM`.

Para criar o conjunto de dados utilizámos a metodologia LIBRETTI, que foi desenvolvida para criar conjuntos de dados com preferências de utilizadores obtidas, implicitamente, a partir de literatura científica. Esta metodologia pode ser dividida em duas etapas: recolha de dados e criação do conjunto de dados. Para a etapa de

recolha de dados, começamos por criar uma lista de doenças utilizando termos retirados do *Medical Subject Headings*(MeSH). Em seguida, para cada doença da lista, recolhemos artigos científicos utilizando a API do PubMed e extraímos os autores de cada artigo. O resultado desta etapa é uma base de dados relacional com toda a informação recolhida. Na etapa seguinte, criámos uma matriz de classificações utilizador-item a partir da base de dados relacional obtida na etapa anterior. A matriz obtida é o conjunto de dados DisRM (*Disease Ratings Matrix*, que se traduz para matriz de classificações de doenças), em que os itens que serão recomendados são doenças, os utilizadores são autores de artigos científicos e as classificações correspondem ao número de artigos que um autor escreveu sobre uma doença. O resultado desta etapa é, então, o conjunto de dados DisRM: um ficheiro no formato .csv em que cada linha é um tripleto <*utilizador*, *item*, *classificação*> e que tem um total de 2 309 190 classificações. Para além deste conjunto de dados, criámos mais duas versões do DisRM: DisRM10 e DisRM20. Estes contêm apenas os utilizadores que têm um total de classificações igual ou superior a 10 e a 20, respetivamente, tendo sido criados para prevenir a partida a frio dos utilizadores e, também, para se assemelhar a outros conjuntos de dados como o Movielens, que só inclui utilizadores que tenham classificado 20 ou mais filmes. O tamanho destes conjuntos de dados filtrados é de 880 893 e de 537 444 classificações, respectivamente. Os conjuntos de dados DisRM permitem a recomendação de doenças a autores, de forma a ajudá-los a encontrar doenças que não conheçam mas que sejam relevantes para a sua investigação, que possam ser semelhantes à sua área de estudo, e que não seriam encontradas utilizando os métodos tradicionais de recuperação de informação. Nestes métodos tradicionais de recuperação de informação, quando diferentes utilizadores fazem uma pesquisa pela mesma doença, obtêm os mesmos resultados. Ao utilizarem a abordagem dos sistemas de recomendação, torna-se possível obter resultados personalizados com base nos interesses passados de cada utilizador e dos seus colegas investigadores. O DisRM é, também, extremamente útil para o treino e teste de sistemas de recomendação em saúde em situações em que a informação que alimentará o sistema de recomendação no futuro é privada de forma a impulsionar a investigação e o desenvolvimento dos sistemas de recomendação em saúde.

Após a criação dos conjunto de dados, aplicamos vários algoritmos de filtragem colaborativa para avaliar qual obtinha melhores resultados com os nossos dados e, também, validámos os conjuntos de dados comparando o seu desempenho com outros conjuntos de dados padronizados. Para isso, utilizámos uma biblioteca chamada Filtragem Colaborativa para Java (CF4J) que permite utilizar vários algoritmos e métricas de avaliação implementados. Utilizando esta biblioteca, aplicámos um algoritmo de filtragem colaborativa k-vizinhos mais próximos baseado em memória e testámos várias métricas de similaridade para calcular a semelhança entre os utilizadores (o coeficiente de correlação de Pearson (COR), a similaridade por cosseno (COS), o coeficiente de similaridade de Jaccard (JAC), a diferença quadrada média (MSD), a diferença quadrada média de Jaccard (JMSD) e a Proximidade-Impacto-Popularidade (PIP)). Para avaliar a qualidade das recomendações utilizaram-se as medidas erro médio absoluto (MAE), precisão, *recall*, medida f, ganho cumulativo com desconto normalizado (nDCG) e cobertura. Após a aplicação dos algoritmos mencionados anteriormente aos DisRMs, o algoritmo que obteve melhores resultados foi o PIP para o conjunto de dados DisRM10: obteve o *recall*(0.81) e nDCG(0.87) mais altos o que significa que é o algoritmo que recomenda a maior quantidade de itens relevantes e com melhor posição na lista de recomendações. Num contexto de investigação como este, em que o objectivo principal do sistema de recomendação é reduzir a quantidade de informação que um utilizador tem que ler, obter um elevado *recall* é de extrema importância, de forma a garantir que está a ser recomendado ao utilizador a maioria dos itens relevantes existentes, ou seja, que se o utilizador explorar apenas a lista de recomendação que foi obtida para si, terá acesso à maioria dos itens considerados relevantes para si. Para validar o nosso conjunto de dados, comparamos o DisRM10 e o DisRM20 a quatro conjuntos de dados padronizados: o Movielens1M, o SD4AI, o ARM10 e o ARM20. Os resultados obtidos com o DisRM foram semelhantes ao ARM e melhores que o SD4AI e que o Movielens1M tanto para o *recall* como para o nDCG, o que valida a qualidade do nosso conjunto de dados.

Acknowledgements

First, I would like to thank my adviser, Prof. Márcia Barros, for all the help and support throughout this time, without whom this work would not have been possible, and my former adviser Prof. Francisco Couto, for the guidance.

Secondly, I want to thank FCT for the fellowship that financially supported me during part of this work and also for supporting this thesis by funding of Deep Semantic Tagger (DeST) project (ref. PTDC/CCI-BIO/28685/2017) and LASIGE Research Unit (ref. UIDB/00408/2020 and ref. UIDP/00408/2020).

To my dear university (and life) friends, Margarida, Rita, and Sofia, for the motivation, friendship and laughter. I'm glad I got to do this with you.

Also, a big thank you to my family, especially my parents, sister, and grandmother, for literally everything.

Lastly, I want to thank Teodoro for the unconditional motivation, support, friendship, and love.

Contents

List of Figures

List of Tables

Acronyms

CB Content-based. 3–5

CF Collaborative Filtering. 3–7, 17, 29, 30

CF4J Collaborative Filtering for Java. 17, 18, 20–22, 30

COR Pearson Correlation Coefficient. 4, 17

COS Cosine Similarity. 4, 17

HRS Health Recommender Systems. 1, 2, 8, 9, 29

JAC Jaccard Similarity Coefficient. 17

JMSD Jaccard Mean Squared Differences. 4, 18

MAE Mean Absolute Error. 5

MSD Mean Squared Differences. 17, 18

nDCG Normalized Discounted Cumulative Gain. 18, 21, 22, 29

PHR Personal Health Record. 7, 8

PIP Proximity-Impact-Popularity. 4, 18, 20, 21, 29, 30

RS Recommender Systems. 1, 2, 6, 7, 14, 17, 29

Chapter 1

Introduction

Recommender Systems (RS) are tools that provide personalized recommendations of items to users, using their preferences to show them relevant items. The first recommender system appeared in 1992 as a new personalized way to filter the high number of emails received (Goldberg *et al.* (1992)). These systems rapidly extended to several different areas that needed a new way to explore the large amount of information that exists online, to reach better results than the traditional information retrieval systems. There are several applications of these systems in many fields in our day-to-day life. Some examples are the recommendation of music from Spotify[1], movies from Netflix[2], and online products to buy from Amazon[3].

1.1 Motivation

In the life and health sciences and medical fields, there is also an increasing amount of digital information. On the one hand, there is more and more health research and publications that can benefit from new ways to find and correlate information, and on the other hand, there is an increase in the number of health records that are available digitally and may be used to improve both personalized health systems and health research developments. That called for new ways of exploring this data in addition to the traditional information retrieval methods. Health Recommender Systems (HRS) appeared in 2007 for this purpose (Valdez *et al.* (2016)) but the study of health recommender systems is still very recent, so their use is still limited.

1.2 Problem

For the development and evaluation of recommender systems, it is necessary to have data about users' preferences. Although there has been an increase in digital health records, health data is generally private and difficult to access. That hinders the process of developing good health recommender systems unless you have access to data about users' preferences. This is a major problem in this area that has been leading to the underdevelopment of HRS. This slow development is also due to this being a recent application of RS.

[1] www.spotify.com

[2] www.netflix.com

[3] www.amazon.com

1.3 Objective

The main objective of this work is to create an implicit feedback dataset based on life and health sciences literature. To achieve this, we will test an existing methodology to create datasets suitable for testing and evaluating recommender systems, using the vastly available scientific literature for that purpose.

1. A dataset accessible to everyone with information about the preferences of users in the health field allowing the testing and evaluating of health recommender systems;
2. A new health recommender system, testing state-of-the-art recommender algorithms.

1.4 Methodology

The methodology of this work can be divided into three main parts:

- Creation of a dataset;
- Development of the recommender system;
- Validation of the recommender system.

1.5 Contributions

Thus, the following specific contributions can be enumerated as follows:

Contribuition 1: 1 Dataset in the field of health sciences;

Contribuition 2: 1 Recommender system for life and health sciences recommendations.

1.6 Overview

The overview of this document is as follows. Chapter 2 refers to the related work where we give a context on RS and explore the state of the art in HRS. Chapter 3 focuses on the creation of the health dataset: the methodologies and the description of the dataset created. In Chapter 4 we evaluate the dataset by testing it with several collaborative filtering algorithms and comparing it to several baseline datasets. Finally, Chapter 5 presents the final conclusions, as well as the future work.

Chapter 2

Background

This chapter describes basic concepts on Recommender Systems necessary to contextualize this work as well as the current state-of-the-art in Health Recommender Systems.

2.1 Recommender Systems

Recommender systems appeared in response to the need for new ways to explore the increasing amount of information on the web beyond the results of traditional information retrieval. A result of a traditional information retrieval system is a match for the user's query, and it is usually the same for every user that searches for that same keywords. The results of recommender systems are personalized recommendations for each user that go beyond the query. These recommendations incorporate information about the user's preferences to display items that they do not know and that will be relevant to them. The user's preferences are usually collected in the form of ratings. These ratings may come from explicit feedback that the user gave to the item by classifying it, by giving it a thumbs up for example, or from implicit feedback extrapolated from the user's actions, for example, clicking on an item they saw on an e-commerce platform. Combining all the known ratings from every user, it is possible to create a user-item ratings matrix that will be used by the recommender system algorithms to compute the recommendations. These matrices are typically very large and sparse. Figure 2.1 shows an example with the ratings some users gave to articles. For example, User 2 gave a rating of 5 to article 5, a rating of 1 to article 4, and 2 to article 7.

Table 2.1: Example of a user-item ratings matrix.

	Article 1	Article 2	Article 3	Article 4	Article 5	Article 6	Article 7
User 1		3	5			1	
User 2		5		1			2
User 3	1		2			5	4
User 4	2	1	2		4		

The main recommender system approaches are Collaborative Filtering (CF), which uses the similarity between the preferences of the users to provide the recommendation, Content-based (CB), which uses the similarity between the characteristics of the items to provide the recommendations and Hybrid, that combines CF and CB to provide recommendations.

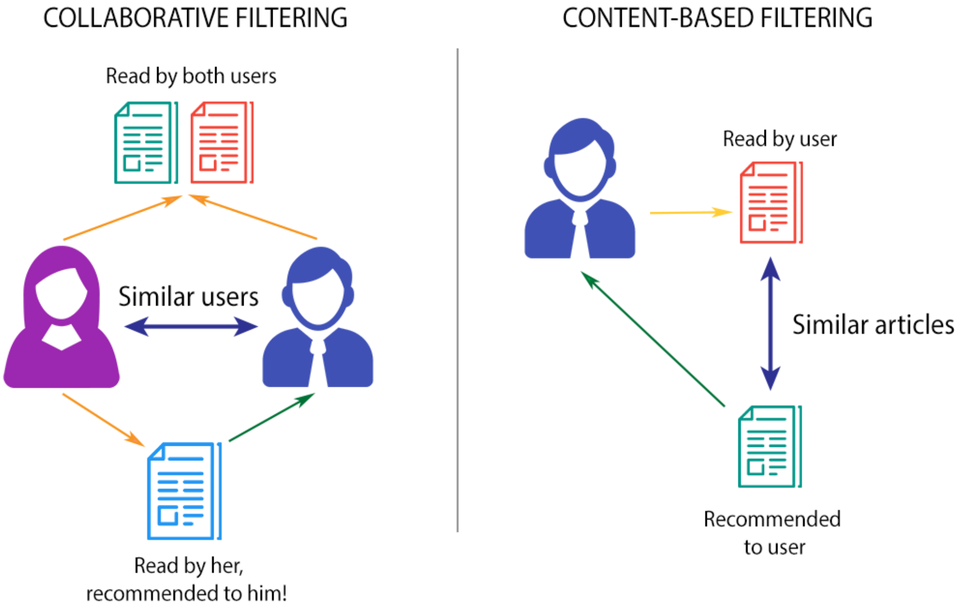


Figure 2.1: Representation of Collaborative Filtering and Content Based Algorithms. Adapted from Mohamed *et al.* (2019).

Using the example of article recommendation (figure 2.1), in CF two users can be similar if they have several read articles in common. If one of them read an article that the other is unaware of, this article is recommended to her/him. In CB, if a user has read several articles and there is one similar that he does not know, that article will be recommended to him.

The next section describes the three types of recommender algorithms in detail.

2.1.1 Recommender System Approaches

2.1.1.1 Collaborative Filtering

In this type of algorithms, the prediction of the missing ratings (items that the user did not rate yet) is based on the ratings given by other users and on the similarity between users. Thus, to predict the rating of user X to item A, the most similar users to user X are found, and their ratings to item A are used to make the missing prediction.

To compute the similarities, several types of functions can be used. There are the traditional statistics metrics like Pearson Correlation Coefficient (COR) and Cosine Similarity (COS) (Ortega *et al.*, 2018a) and other new similarity measures created for the improvement of Collaborative Filtering Algorithms, such as Proximity-Impact-Popularity (PIP) (Ahn, 2008) and Jaccard Mean Squared Differences (JMSD) (Bobadilla *et al.*, 2010).

Because of its nature, collaborative filtering recommender systems work better with more users. There are three main challenges when using this type of algorithm. The first is the cold start for new users and new items: if a user or item is new to the system, there is little or no information to base recommendations on. Secondly, we have the high sparsity of the user-item rating matrix that derives from the fact that each user only rates a small percentage of all existing items. The last challenge is the size of the dataset: the bigger it is, the less efficient and effective the algorithm will be.

2.1.1.2 Content Based

Content-based algorithms use the similarity between items, rather than users. There are several ways to calculate this similarity. The most common method is to characterize each item with several attributes and compare them based on that characterization. For example, if the items are movies, we can compare them based on the genre, the actors, and the directors. The ratings are then predicted based on the most similar items rated by the user. CB algorithms are commonly used in science when datasets have few users but well characterized items.

This type of recommender system also has a problem of cold start for new users but not for new items because the items do not need to be rated to be recommended, just characterized to be compared to other items. CB recommendations tend to lack diversity because they are always based on what the user already saw, not recommending different items.

2.1.1.3 Hybrid

In order to overcome the challenges of each algorithm, it is possible to combine both algorithms, CF and CB in a hybrid algorithm. Systems with hybrid algorithms can benefit from the strengths of each one without some of their disadvantages. For example, if we combine a CF and CB algorithms, the cold start problem for new items will be overcome.

2.1.2 Recommender System Evaluation

Recommender systems can be evaluated using offline or online methods (Aggarwal (2016)).

Offline evaluation is the most common for research purposes because it does not require interaction with real users, just a dataset with ratings. In offline methods, we divide the dataset into training and test set, and we use the former to predict the ratings of the latter. To evaluate the quality of the predictions it is common to calculate prediction error metrics (metrics that evaluate how much the predictions deviate from the real value) such as Mean Absolute Error (MAE) (measures the average deviation of the predictions from the real values), mean squared error (MSE) (measures the average of the squared deviation of the predictions from the real value, which allows giving more weight to outliers but the result is no longer in the same unit as the dataset) and root mean square error (RMSE) (represents the square root of the squared deviation of the predictions from the real values and allows to give more weight to the outliers while the result remains in the same unit as the dataset). For the evaluation of the quality of the set of recommendations generated, the most used metrics are precision (proportion of relevant recommended items in all recommend items), recall (proportion of relevant recommended items in all relevant items), and f-measure (accuracy measure that combines precision and recall). However, with offline evaluation, it is not possible to predict the impact of recommendations, hence the use of online evaluation (Bobadilla *et al.* (2013)).

Online evaluation requires the implementation of the recommender system into a platform thus real users may test it, and the results may take days or even weeks to be available. This method allows evaluating the satisfaction of the user by collecting (implicitly or explicitly) feedback from the user about the provided recommendations. This feedback can be collected through the click-rate (CTR) of the item only or sometimes along with A/B testing, which allows the implementation of two different systems to assess which one has a better performance (Aggarwal (2016)).

2.1.3 Recommender System Datasets

Recommender system datasets emerged from the need of data to test new recommender algorithms. Some of the most popular RS datasets are the Movielens datasets[1]. They were first released in 1998 and are the result of real users using the Movielens system of movie recommendations. Their popularity is the reflection of the growth of personalization and recommendation research, in which datasets such as these have substantial value in exploring and validating ideas and also attributed to the accessibility of movies as content domain, since movies are a theme that makes the output easy to discuss, due to its popularity. These datasets can be used as baseline because they have been highly tested. Movielens originated four CF datasets, with different amounts of ratings that we can see described in table 2.2. As the size of the dataset increases, they become more sparse. All four datasets have the same structure, having rating tuples of the form <user, item, rating, timestamp> and only including users with at least 20 ratings.

Table 2.2: Description of the four Movielens datasets, relatively to the date range of the ratings, the rating range and scale, the number of users, the number of items, the number of ratings and sparsity.

Name	Date Range	Rating Scale	Users	Movies	Ratings	Sparsity
ML 100K	09/1997 - 04/1998	[1-5] (stars)	943	1 682	100 000	93.7%
ML 1M	04/2000 - 02/2003	[1-5] (stars)	6 040	3 706	1 000 209	95.53%
ML 10M	01/1995 - 01/2009	[0.5-5] (half-stars)	69 878	10 681	10 000 054	98.66%
ML 20M	01/1995 - 03/2015	[0.5-5] (half-stars)	138 493	27 278	20 000 263	99.46%

Specifically in the scientific field, Ortega *et al.* (2018a) created the dataset SD4AI (Scientific Documentation for Artificial Intelligence), a dataset with the objective of providing the necessary resources so that the scientific documentation field can be enriched by the advances currently experienced by the RS field, the same way that this work intends to do with the health field. This dataset relates scientific papers about artificial intelligence with their main research topics. They retrieved articles from Scopus[2] (Elselvier's abstract and citation database) of the "Computer Science, Artificial Intelligence" area and with high rankings (from journals from the first third of the first quartile(Q1)), from 2016 to the first half of 2018. Like in our dataset, the feedback in SD4AI was obtained implicitly. It represents the number of times the topic appears in the paper and it is referred to as cardinality. It is structured similarly to Movielens except for the timestamp, it is constituted of tuples <paper, topic, cardinality>. SD4AI was tested against CF RS baselines and it showed similar behaviour, so it is validated to serve as baseline as well.

Table 2.3: Description of the SD4AI dataset, relatively to the cardinality range and scale, the number of papers, the number of topics, the number of ratings and sparsity.

Name	Cardinality Range	Papers	Topics	Ratings	Sparsity
SD4AI	[1-160] (step of 0.25)	14 143	18 502	1 389 094	99.47%

[1] https://grouplens.org/datasets/movielens/

[2] www.scopus.com

Barros *et al.* (2019) created a methodology for the development of scientific CF datasets based on implicit feedback from open repositories of scientific documentation with the objective of improving recommender system development in scientific contexts. This methodology consists on the creation of CF datasets that recommend scientific items to authors of articles (users). It takes a collection of scientific items, collects articles about each item using external sources of knowledge, and stores both in a database. The authors are then extracted to infer how many articles each author wrote about each item. With this information, the user/item ratings sparse matrix is created in which one rating represents the number of articles an author wrote about an item. This methodology was applied to two fields: astronomy, for recommending open clusters of stars, and chemistry, for recommending chemical compounds. In this thesis, we will only focus on one of them, the astronomy use case where the list of items is a list of open clusters of stars and the articles extracted for each cluster were retrieved from ADS[1] (Astrophysics Data System). This dataset is called ARM (Astro Ratings Matrix). Characteristics of this dataset are represented in 2.4.

Table 2.4: Description of the ARM dataset, relatively to the cardinality range and scale, the number of papers, the number of topics, the number of ratings and sparsity.

Name	Cardinality Range	Authors	Clusters	Ratings	Sparsity
ARM	[1-not fixed] (step of 1)	17 006	18 502	1 389 094	99.5%

2.2 State-of-the-art of Health Recommender Systems

The development of recommender systems led to its expansion to other areas such as life and health sciences. RS started being applied in several health contexts both in a clinical and in a research context.

One of the first articles about Health Recommender Systems in PubMed[2] is from 2011. Torkaman *et al.* (2011) proposed an automatic diagnosis recommender system for classifying leukemia based on cooperative game. They used real data collected from the Iran Blood Transfusion Organization (IBTO) that constitutes a dataset with 400 samples taken from human leukemic bone marrow and obtained an accuracy rate of 93.12% and values of precision, recall and f-measure of 93.12%, 99.3% and 96.1%, respectively. Due to the nature of the data, this dataset is not available publicly.

Wiesner & Pfeifer (2014) proposed the recommendation of health information artifacts (medical resources available online such as the ones present in PubMed - items) to be displayed in a Personal Health Record (PHR) (medical history of the patient accessible to the patient and the health professional), with the goal of lowering the effects of information overload which originates from the increasing amount of health related data. These recommendations were based on the personal data existing on the PHR and were obtained through content-based algorithms. They propose two different use cases. The first has the health professional as the end-user, where the PHR can be enriched with case-related information like related clinical guidelines or research articles that may help the physician with the clinical diagnosis. It can also be enriched with laymen-friendly information on how to live with a disease or how to change lifestyle habits, to be

[1] https://ui.adsabs.harvard.edu/

[2] https://www.ncbi.nlm.nih.gov/pubmed/

given to the patient when they have a medical consultation. The second use-case has the patient as the end-user and an active viewer of its own PHR without the direct support of a health professional. The goal is that the patient is empowered in terms of health information acquisition by being recommended high-ranked (only evidence-based, high quality) health documents and media content. Although the authors say this study is not completed, they compared their algorithm with simple Information Retrieval and obtained better recommendations. Nevertheless, data was a limitation to the study because they only had discharge letters (that acted as PHR) from only one specialty (cardiology) of only one hospital. With this limitation of data diversity, because of data privacy, it is harder to improve their algorithm.

Chen *et al.* (2015) also created an HRS that uses PHR but, in this case, their goal was to provide actionable decision support in the form of clinical order suggestions and, instead of content-based, it used collaborative filtering algorithms. Their motivation is the fact that, nowadays, there is great variability and inconsistency in medical practice, as well as a lack of medical experience that can be solved with a data-driven clinical decision support that reinforces consistency and compliance with best practices. The data was structured and collected from PHR of 18 thousand patients over the course of a year, from hospitalizations at Stanford University Hospital in 2011, constituting a private dataset. In this sample, there were more than 5.4 million instances of 17 thousand distinct clinical items (these items include medication, laboratory, and imaging items, among others). They obtained improvements in recommendations, in comparison with their benchmark (precision improved from 33% to 38%), but these values are still far away from giving confidence for the decisions to be made without a human decision maker.

Beyond clinical contexts, HRS can also be applied to other health areas.

Yang *et al.* (2017) proposed a personalized nutrient-based meal recommender system that aimed to meet individuals' nutritional expectations, dietary restrictions, and fine-grained food preferences. This system can provide healthy options, based on the users' food preferences and restrictions in order to promote healthy dietary habits and help the users prevent or manage health conditions such as obesity and diabetes. The authors used content-based algorithms to recommend food alternatives that agree both with their taste and their health, and their results outperformed other baselines by a significant margin. In this study, the authors used Food-101 (Bossard *et al.*, 2014), a big dataset for food images (101 000 images) to develop and test the tool that computes the similarity between two foods (based on their images). Additionally, to evaluate their recommender system, they used an online evaluation approach but the user data was not made publicly available.

Beyond clinical diagnosis and lifestyle recommendations, HRS are key to boost health research to the next level. Suphavilai *et al.* (2018) developed CaDRReS (Cancer Drug Response prediction using a Recommender System), a method that uses collaborative filtering algorithms to predict cancer drug responses for unseen cell-lines/patients. Additionally, it can help to understand drug mechanisms, identify cellular subtypes and further characterize drug-pathway associations. The data used to test their models was obtained from two public large-scale and one private inhouse datasets with drug-screening data for cancer cell-lines. The authors consider that the number of cell types and drugs in the available datasets is very limited so a method to develop new datasets with more complete data is important.

With this analysis, we can observe that, over the years, the interest in HRS has been increasing. However, most of the datasets are private and not available, making it difficult to replicate the studies and develop new algorithms. Also, to the best of our knowledge, there are neither studies nor datasets focused on recommending diseases of interest based on the past interests of the researchers/medical staff.

Chapter 3

Health Dataset

The goal of this work is to create an open source dataset of implicit feedback for the recommendation of diseases. This is a dataset of users, items, and ratings, where the users are authors from research articles, the items are diseases and the ratings are the number of articles an author wrote about a disease. The work developed on this chapter is available on GitHub: `https://github.com/teresacunha/DisRM`.

3.1 Methodology

We will follow the methodology LIBRETTI developed by Barros *et al.* (2019) to create implicit feedback datasets using scientific literature. Figure 3.1 represents an overview of that methodology applied to this work. It has two steps: Data Collection and Dataset Creation.

In the **Data Collection** step, we started by creating a list of diseases using terms from MeSH. Next, for each disease in the list, we retrieved research articles from PubMed using their API and extracted the authors from each article. The result of this step is a relational database with all the information collected.

In the **Dataset Creation** step, a user x item matrix is created based on the data from the relational database. This matrix is the dataset that we called **DisRM** (Diseases Ratings Matrix), in which the items to be recommended are the diseases, the users are the authors of the research articles and the ratings are the number of articles an author wrote about a disease. The result of this step is the dataset file in *.csv* format, in which each line is a triplet: <*author*, *disease*, *rating*>.

3.1.1 Data Sources

As explained in Chapter 2, one of the biggest difficulties for the development of HRS is the privacy of data. For DisRM, we chose open source data that can be accessible by everyone and used freely in the study of HRS. We chose PubMed as the source of life and health sciences literature because it gives access to a very large database of biomedical articles and allows us to retrieve a great amount of information to build DisRM using its API.

We obtained the list of diseases from MeSH[1] which is a medical controlled vocabulary organized in a hierarchical structure that has 16 top-level categories, one of which being "Diseases".

[1] https://www.nlm.nih.gov/mesh/meshhome.html

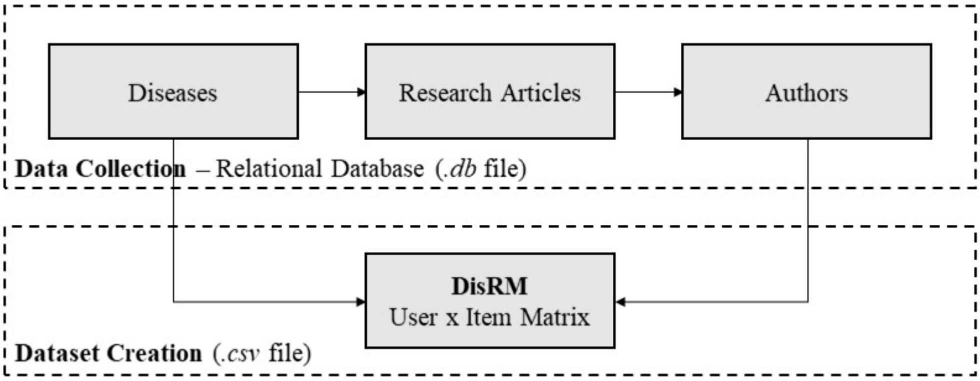

Diseases
Research Articles
Authors
Data Collection – Relational Database (.db file)
DisRM
User x Item Matrix
Dataset Creation (.csv file)

Figure 3.1: Methodology overview.

Thus, our list of items corresponds to the portion of MeSH terms that are classified as diseases. We chose MeSH as our source of items because PubMed allows query expansion using MeSH terms, ensuring that the articles retrieved were tagged with the intended disease and that articles that only mentioned the searched disease but were about others were not retrieved. In figure 6.1 (in the appendix) is a schematic representation of a portion of MeSH. We can see that, as we advance deeper in the hierarchy, the terms get more and more specific. For example, "Respiratory Tract Disease" is a much broader and encompassing term than "Asthma". As a consequence of this being a hierarchical structure, the list of diseases in DisRM will have terms with different levels of specificity. Another consequence of the hierarchical structure is that there are duplicate terms because the same term can have more than one different direct ancestral.

3.1.2 Data Collection

The first step in the data collection was the MeSH data[1] processing. We extracted the terms classified as diseases (C terms) and also excluded all the duplicated terms, resulting in a list of unique diseases.

Next, we used the Entrez Programming Utilities (E-Utilities) (NCBI, 2010) to retrieve articles from PubMed via API. The E-Utilities allows the data search and retrieval via API from 38 NCBI databases, namely PubMed. For each disease, we retrieved the hundred most relevant articles, relative to humans, from 1998 to 2019 by using the query below, replacing "disease" with the item to search:

```
term = 'disease[MESH] AND human[MESH]',
retmax = 100,
sort = 'relevant',
datetype = 'pdat',
mindate = '1998',
maxdate = '2019'
```

The search was limited to articles concerning humans because we are aiming to create a dataset for the recommendation of human diseases. For each article, we saved the title, the abstract, the authors and their affiliations, the publication year, the PMID (PubMed Unique

[1] https://www.nlm.nih.gov/mesh/2019/download/2019New_Mesh_Tree_Hierarchy.txt

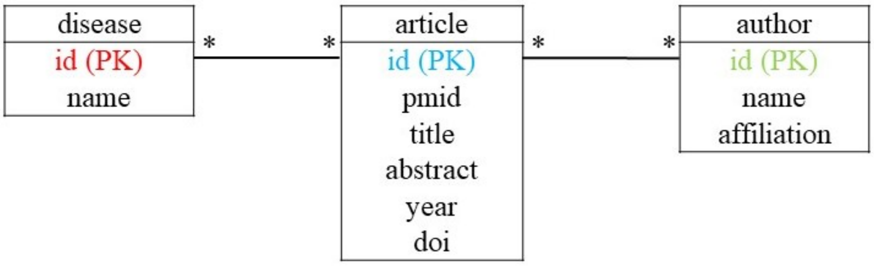


Figure 3.2: Structure of Relational Database.

Identifier), and the DOI (Digital Object Identifier). With all this information, we created an SQLite database with the structure presented in figure 3.2. The IDs presented in color were created specifically for this database to serve as keys.

3.1.3 Dataset Creation

Having all the information stored in a database, we proceeded with the creation of the dataset. In this step, we had to infer implicit feedback from the data in the database by assessing the number of articles each author wrote about each disease.

First, we processed the list of authors to remove the duplicate users. This merge was necessary to increase the number of ratings per author and reduce the cold start problem for users. We merged all the authors that had the same name.

Having a list of unique authors, we proceeded with the generation of the ratings. Figure 3.3 is a representation of that process. To achieve this, we iterated over the list of unique authors (represented by circles and the letter "A") and, for each author, retrieved the articles they wrote (represented by the rounded squares and the letters "Ar"). For those articles, we retrieved the associated diseases (represented by the squares and the letter "D"). After this process, we know how many times an author wrote/had interest in a disease.

In figure 3.3 is represented an example with four authors (A1, A2, A3, and A4). A1 wrote two articles about disease D1, so the rating for D1 from user A1 is 2. A2 only wrote an article about only one disease so the rating of that author to that disease is 1. A3 also wrote only one article but that article was about diseases D1 and D2 so author A3 has a rating of 1 for both diseases. Lastly, author A4 wrote three articles: Ar5 about diseases D3 and D4 and Ar6 and Ar7 about disease D4. Therefore their rating for disease D3 is 1 and for disease D4 is 3.

In addition to this dataset, we created two other versions of DisRM: DisRM10 and DisRM20, including only the users who have a number of ratings equal to or greater than 10 and 20, respectively. These filtered datasets were created to prevent the cold start for users and also to mimic other datasets, such as Movielens, which only includes users who have rated 20 or more movies.

3.2 Results and Discussion

In total, the database has 4 819 diseases, 280 569 articles, and 1 552 538 authors. Each article is related to, at least, one disease, but it can be related to more and has one or more authors. This data originated DisRM.

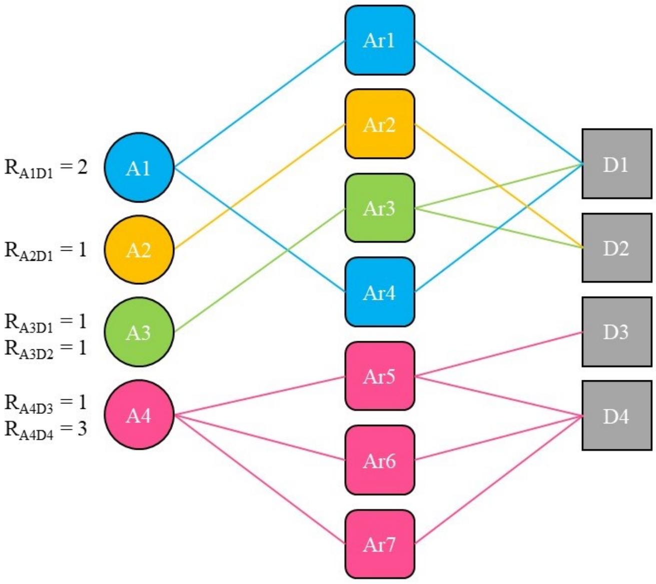


Figure 3.3: Implicit Feedback Inference Representation.

The DisRM dataset, with the information of the user item rating matrix, is structured in a similar way as the Movielens datasets, to facilitate its use: each line of the file is a tuple with three elements separated by a comma: *author*, *disease*, *rating*. In table 3.1, it is presented a real excerpt of the DisRM dataset, where it is possible to see that user 6 wrote articles about, at least, four diseases, having written 2 articles about disease 1673. As you can see in that table, the authors and diseases are represented with Ids. These Ids were created for the dataset only and have no meaning by themselves but, by consulting the database, it is possible to identify the authors and the diseases. In table 3.2 that conversion is presented: so, author Hoen PA wrote 2 articles about "Muscular Dystrophy, Oculopharyngeal".

Table 3.1: Excerpt of DisRM dataset.

Author	Disease	Rating
1	2024	1
2	3754	1
3	2377	1
5	472	1
6	456	1
6	1673	2
6	2575	1
6	4326	1

Table 3.2: DisRM excerpt from table 3.1 converted to show author and diseases names.

Author	Disease	Rating
Villa ALF	Submandibular Gland Diseases	1
Aho T	Thrombocytosis	1
Episcopo FL	MPTP Poisoning	1
Hoen EFM	Hepatitis C	1
Hoen PA	Ecthyma, Contagious	1
Hoen PA	Muscular Dystrophy, Oculopharyngeal	2
Hoen PA	Cluster Headache	1
Hoen PA	Muscular Dystrophy, Animal	1

As we can see in table 3.3, by grouping the authors by unique name, we reduced the number of authors from 1 552 538 to 674 859 unique authors. The inference feedback step originated a dataset with 2 309 190 ratings with a sparsity of 99.93%. The *.csv* file of the dataset has 2 309 190 lines, corresponding to the number of ratings.

As mentioned in section 3.1.3, we created two variants of the original dataset: DisRM10 and

DisRM20. The purpose of these variants is to eliminate the cold start problem for the users by ensuring that all the users rated, at least, 10 or 20 diseases. We decided to choose the threshold 20 because it is the minimum number of ratings per user in the MovieLens datasets (Harper & Konstan, 2016) but, as we can see in table 3.3, on the one hand, it reduced considerably the number of users and ratings but, on the other hand, it reduced the sparsity. Therefore, we also created the DisRM10 to have an option with more users and ratings than the DisRM20 and to have another dataset to test the performance with different algorithms.

In table 3.3 we also present the description of the four baseline datasets we chose to validate DisRM. We chose the Movielens1M from the set of the Movielens dataset because it was the one with the most similar size to the DisRM datasets variants. SD4AI also has around one million ratings. The ARM datasets are the only ones with a different order of magnitude, having around a hundred thousand ratings. Movilens1M is less sparse than all the DisRM datasets and the least sparse dataset of the four baselines. Both the ARM datasets are less sparse than all DisRMs and SD4AI has a level of sparsity similar to DisRM10.

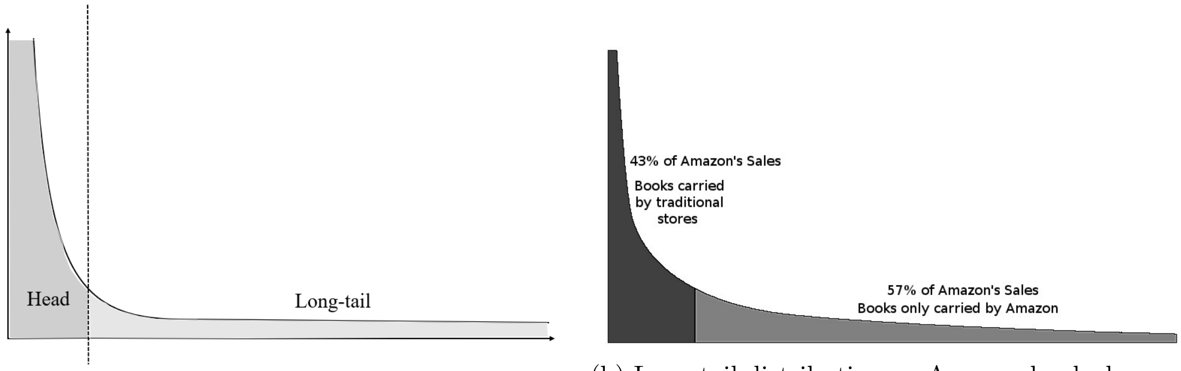
Table 3.3: Comparison of the three DisRM datasets created with the 4 baseline collaborative filetering datasets, relatively to the number of users, number of items, number of ratings and sparsity.

	Number of users	Number of items	Number of ratings	Sparsity
DisRM	674 859	4 819	2 309 190	99.93%
DisRM10	36 684	4 780	880 893	99.50%
DisRM20	10 131	4 764	537 444	98.89%
Movielens1M	6 040	3 706	1 000 209	95.53%
SD4AI	14 143	18 502	1 389 094	99.47%
ARM10	3 613	2 102	138 558	98.18%
ARM20	1 493	2 101	106 104	96.62%

Considering the ratio between the number of users and the number of items, the three DisRM datasets have more users per item than all the baseline datasets. This is due to the nature of the health research field. It is an enormous research area from which we took only a small sample so it is expected that, in this small sample, we are not able to find many articles from the same author, but many different authors. Although we followed the same methodology that was used to create the ARM datasets, this ratio in the ARM datasets is not as high as the DisRM datasets because the astronomy field is considerably smaller so it is possible to retrieve a bigger percentage of all the information that exists and have more articles (and consequently more items) per author. This difference is more significant in the original dataset DisRM for this reason and decreases as we increase the minimum number of rated items per user.

3.2.1 Long Tail Distribution

The popularity of items in an online retailer follows a long-tail distribution (see figure 3.4a), in which the head corresponds to the popular items (typically the ones that a physical retailer will have available) and the tail corresponds to all the remaining products that normally can't be in stock in a physical retailer because they are less popular, but are in an online one and represent the majority of the sales (Anderson, 2004). For example, the popularity of books available in Amazon follows this distribution and the products that are only sold online represent 57% of their sales (see figure 3.4b) (Patel, 2015). In the e-commerce context, RS help filter this huge amount of information in order to give customers ways to find less popular products that suit their particular needs and interests (Anderson, 2006).



(a) Representation of long-tail distribution.

(b) Long tail distribution on Amazon books by popularity.

Figure 3.4: Long tail distribution. a) shows an schematic example of a long tail distribution with representation of the head and tail and b) shows the distribution of books sold by amazon by popularity.

We can see in figure 3.5a that the popularity of the items (measured by analyzing the rating frequency of each disease) follows a long-tail distribution. We can observe that DisRM follows this typical data distribution commonly present in RS environments. In DisRM, this distribution highlights the purpose of this case study. In the head are the most studied diseases that most authors probably know and the tail corresponds to a great amount of diseases that the authors are less likely to know but that may also be of their interest. Thus, RS will help authors to easily find the most relevant diseases for their research.

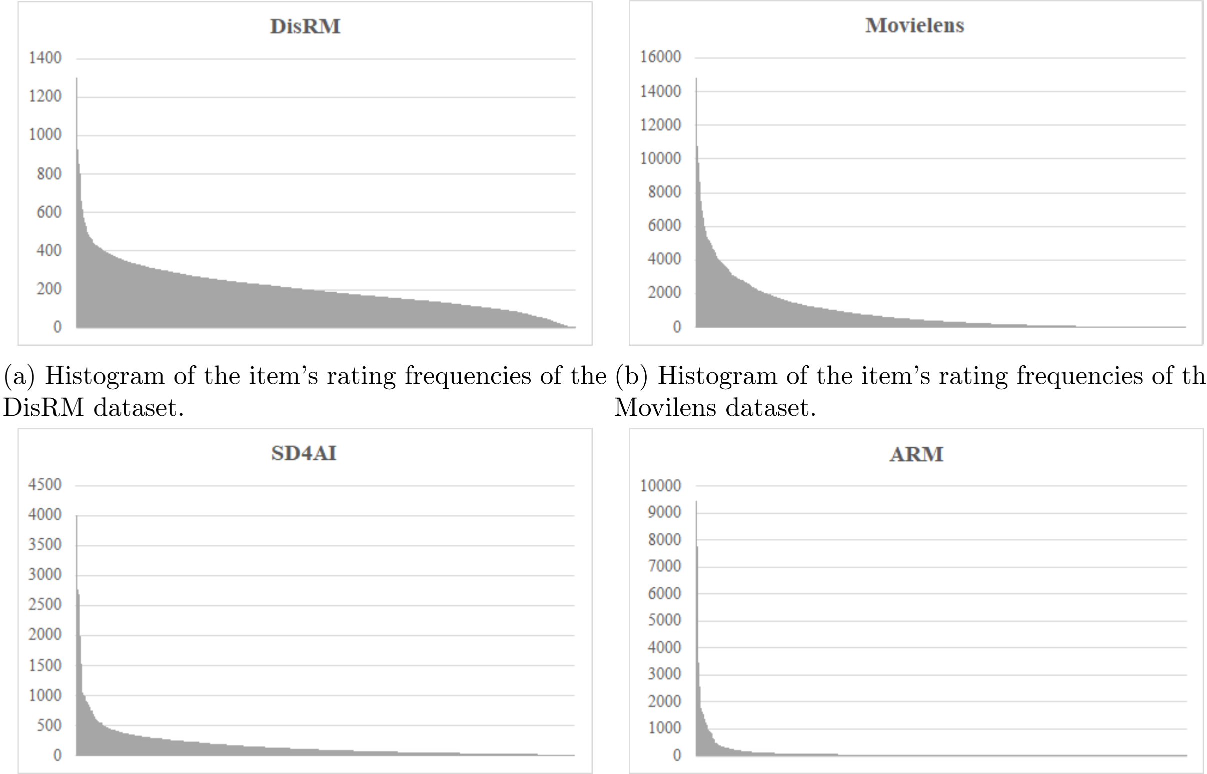
It is important to note that, because the items originated from a hierarchical vocabulary, there are terms like "Respiratory Tract Diseases" that will, naturally, be more popular for being less specific than the terms in the bottom of the hierarchy such as "Asthma".

Performing the same analysis in the three baseline datasets (see figure 3.5), we found that all of them follow this long tail distribution.

3.2.2 Ratings Distribution

As we mentioned earlier, each rating in DisRM represents the number of articles an author wrote about a disease. The minimum rating is 1 and the maximum rating is 58 (followed by 34), meaning that there is an author that wrote 58 articles about a disease. When consulting the dataset we can see that it was Guo X that wrote 58 articles about "Kashin-Beck Disease" and Davey G that wrote 34 articles about "Elephantiasis". In figure 3.6 we can see all the existing values of ratings and their distribution in the dataset, following a logarithmic scale. The most common rating is 1 and represents the majority of all ratings (93.51%), the second and third most common ratings are 2 and 3 and represent 5.02% and 0.91% of all ratings, respectively. Combined, the ratings of values 1, 2, and 3 represent 99.44% of all ratings.

There are several factors that can explain this distribution. First, we only retrieved the hundred most relevant articles for each disease, which means that we only have a small sample of all PubMed database. It is normal that, in a small sample, there are not many articles written



(a) Histogram of the item's rating frequencies of the DisRM dataset.

(b) Histogram of the item's rating frequencies of the Movilens dataset.

(c) Histogram of the item's rating frequencies of the SD4AI dataset.

(d) Histogram of the item's rating frequencies of the ARM dataset.

Figure 3.5: Comparison of distribution of items rating frequency in several datasets. a) shows DisRM, b) shows Movielens, c) shows SD4AI and d) shows ARM.

by the same author about the same disease. This explains why the smaller ratings represent most of the dataset. Second, the fact that we grouped authors by name without considering their affiliation combined with the existence of generic terms of diseases, may explain some of the higher values of rating.

Since we are using implicit feedback, all of the ratings in the dataset are positive, we don't have negative feedback. This can be an impediment to applying some recommender algorithms. We will explore this matter further in Chapter 4.

3.2.3 Dataset Applications

DisRM allows the recommendation of diseases to authors, in order to help them find unknown, but relevant, diseases to their research that may be similar to their field of study and that they wouldn't find using regular information retrieval methods. In a regular retrieval system, different users searching for the same disease will obtain the same results. When using RS approaches, we will be able to retrieve personalized information based on the past interests of each user and based on the past interests of their peers. It is also possible to apply this methodology to create other datasets of other scientific entities such as recommendation of genes or drugs.

DisRM is also useful for preliminary training and testing of health recommender systems if the type of information to be used in the future is private. As mentioned in chapter 2, there is

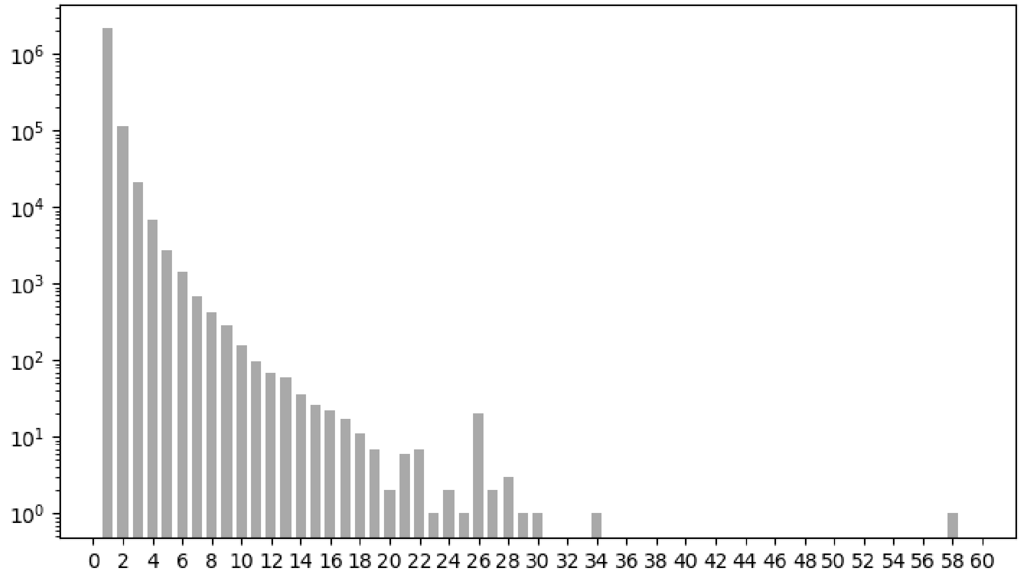


Figure 3.6: Distribution of the different values of ratings in the dataset.

a great lack of public health datasets which is slowing down the development of recommender systems in health. This dataset can help boost the investigation of health recommender systems.

Chapter 4

Collaborative Filtering Evaluation

Now that the new public health dataset was created, we applied several Collaborative Filtering (CF) algorithms to evaluate which one performed better with our data and also validated the dataset by comparing its performance using the same CF algorithms with several other open datasets.

4.1 Methodology

To evaluate the best CF for our dataset, we performed several experiments using a library called Collaborative Filtering for Java (CF4J) (Ortega *et al.*, 2018b). This library was created to support CF-based RS research experiments and it allows the user to use several implemented algorithms and evaluation metrics. Using this library, we applied a memory-based CF K-Nearest Neighbors algorithm and tested several similarity metrics to compute the similarity between the users:

- Pearson Correlation Coefficient (COR) - This is one of the most used methods. It measures the linear correlation between two vectors and the resulting value ranges from -1 to +1, where the higher its absolute value the greater the correlation. If the result is greater than zero, we have a positive correlation, if it is less than zero, we have a negative correlation and if it is zero, there is no correlation (Agarwal & Chauhan, 2017).
- Cosine Similarity (COS) - This is another commonly used method in collaborative filtering algorithms. It measures the cosine of the angle between two vectors. A con of this method is that it considers a null preference as negative (Agarwal & Chauhan, 2017).
- Jaccard Similarity Coefficient (JAC) - Jaccard represents the common ratings between two users so, the users will be more similar if they have more rated items in common. This measure doesn't consider the value of the rating, it only considers if the rating exists (Liu *et al.* (2014), Agarwal & Chauhan (2017)). Thus, if two users rated the same items but one of them liked and the other disliked all the items, they will be considered similar users, even though they probably are not.
- Mean Squared Differences (MSD) - This metric only uses the value of the ratings in the similarity calculation, it does not consider the number of common ratings. To compute the similarity between two users, the ratings considered are the ones that correspond to items rated by both users. Thus, if one user has rated 100 items and the other has rated

only 4, but these 4 were also rated by the first user, and if these 4 ratings are very similar, these two users can be considered similar even though they are not. (Sanchez *et al.* (2008), Agarwal & Chauhan (2017))

- Jaccard Mean Squared Differences (JMSD) - Bobadilla *et al.* (2010) created this similarity measure combining Jaccard and Mean Squared Differences to surpass their disadvantages and have a measure that considers both the non-numerical (Jaccard) and numerical (MSD) information.

- Proximity-Impact-Popularity (PIP) - PIP was created by Ahn (2008) as an answer to the cold start problem for new users and it is an heuristic measure composed of three factors: proximity, impact, and popularity. The proximity factor is based on the arithmetic difference between two ratings while also considering if they are in agreement (if they are in the same half of the rating scale), giving a penalty if they aren't. The impact factor will give more or less credibility to the similarity based on how strongly or mildly, respectively, an item is preferred or disliked by the user. Lastly, the popularity factor will give more value to a similarity if the ratings are further from the average rating of the co-rated item.

Beyond the different algorithms, we tested different number of neighbors for the prediction of ratings (10, 50, 100, 150, 200, 250, 300, 350, 400, 450, 500) and also different sizes of recommendation lists (1, 2, 3, 4, 5, 6, 7, 8, 9, 10) to try and find a balance between performance and quality.

We ran 5 rounds of cross-validation, with a train/test set division of the dataset corresponding to 80%/20%, respectively.

We aimed to evaluate the quality of a list of recommendations. For this, we used the already implemented metrics of CF4J:

- MAE to measure the quality of the predictions;

- Precision, Recall, and F-measure to evaluate if the algorithms are recommending the most relevant items;

- Normalized Discounted Cumulative Gain (nDCG) to study if the most relevant items of the recommended list are shown first;

- Coverage of the algorithm.

For the collection of these metrics, we defined the relevance threshold as 2, which means that we consider a disease as relevant for an author if they wrote at least two articles about it. This is necessary due to the nature of CF4J, which was designed to have a threshold for the relevant items. If we define the threshold as 1, all the recommended items to a user would be considered relevant. In the future, we need to test the dataset with other frameworks, however, for the purpose of comparing this work with existing works ((Ortega *et al.*, 2018a) and (Barros *et al.*, 2019)), we decided to use CF4J, even though this disadvantage.

4.1.1 Dataset Validation

It is important to validate the new dataset against others that have already been proven to have good recommendation results. We chose four baseline datasets to compare with DisRM: Movielens-1M, SD4AI, ARM10, and ARM20. Movielens-1M is one of the most used to test algorithms and has already shown to produce good results. It is also roughly the same size as DisRM10, as it has 1 000 209 ratings. Since the Movielens datasets are about movies, we wanted to choose one dataset within the scientific area. We chose SD4AI because, as DisRM, it has scientific articles as its main source of data. In terms of size, SD4AI is slightly bigger, having 1 389 094 ratings. Lastly, we chose two ARM datasets (ARM10 and ARM20) because they were created using the same methodology as DisRM but, instead of health items, they recommend astronomical entities. ARM20 is considerably smaller, having only approximately 100 000 ratings.

We compared the performance of the datasets by applying the same algorithms and evaluation measures as before, changing only the relevance threshold. For Movielens and SD4AI we chose the thresholds indicated by their creators: ratings 5 for Movielens and 3.75 for SD4AI. Since the ARM datasets have the same characteristics as DisRM, we also used the threshold 2.

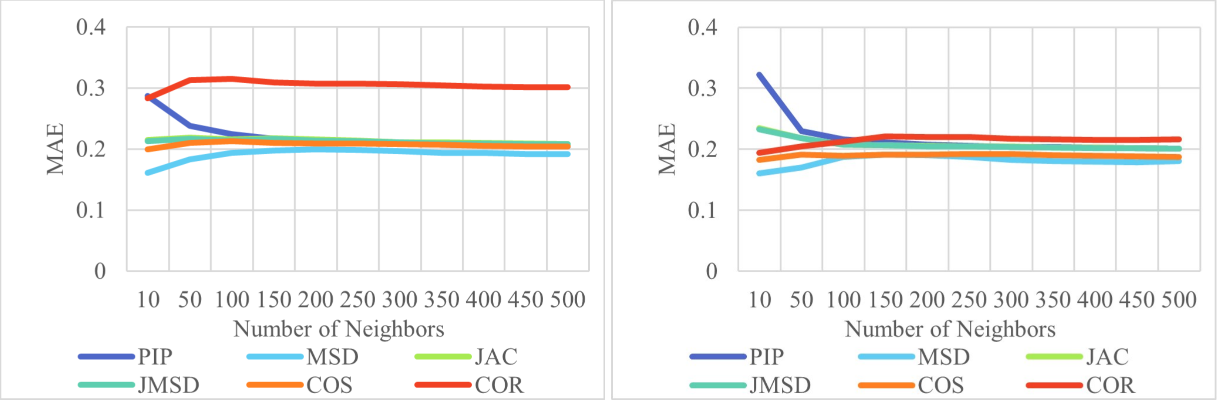
4.2 Results and Discussion

4.2.1 DisRM Evaluation

To assess which dataset performs better, and with which parameters, we evaluated both under six evaluation measures.

4.2.1.1 Mean Absolute Error

$$MAE = \frac{1}{n}\sum_{t=1}^{n}|e_t| \tag{4.1}$$



(a) MAE values obtained with DisRM10.

(b) MAE values obtained with DisRM20.

Figure 4.1: Visual analysis of MAE obtained by applying k-nearest neighbors algorithm to both DisRM datasets using six different similarity measures for the user similarity computation.

4.2.1.2 Coverage

The coverage is the capacity of the recommender system to recommend new items. More specifically, it calculates the percentage of situations in which at least one k-neighbor of each active user can rate an item that has not been rated yet by that active user. It is calculated as follows:

$$coverage = \frac{number\ of\ predicted\ items}{number\ of\ items\ not\ rated\ by\ the\ user} \tag{4.2}$$

Regardless of the dataset, PIP is the one that results in the best coverage. With 50 or more neighbors reaches a coverage of approximately 1, which means that in almost every situation at least one k-neighbor of each active user can rate an item that has not yet been rated by the active user.

4.2.1.3 Precision, Recall and F-measure

These three metrics are calculated to measure the quality of the set of recommendations generated. For a given number k of recommended items, precision (equation 4.3) is defined as the percentage of recommended items that are relevant for the user. The recall (equation 4.4) is defined as the percentage of the total relevant items for a user that have been recommended as positive for a list of size k. The F-measure (equation 4.5) is the harmonic mean of precision and recall, allowing the global evaluation of the recommender algorithms.

$$precision = \frac{relevantItems@k}{k} \tag{4.3}$$

$$recall = \frac{relevantItems@k}{totalRelevantItems} \tag{4.4}$$

$$F - Measure = 2 \times \frac{Precision \times Recall}{Precision + Recall} \tag{4.5}$$

The relevance threshold used for DisRM was 2. This threshold represents the value above which a rating is considered high enough that the item should be recommended to the user. Depending on the data, this value will be different. For example, Ortega *et al.* (2018a), the authors of SD4AI, considered the percentile 85 of the ratings range as the relevance threshold. In DisRM we only have positive implicit feedback, so any rating equal to or above 1 would be considered relevant because it would mean that the author wrote about that disease. Nevertheless, CF4J is not prepared for this type of implicit positive feedback, so the threshold can never be the lowest possible rating because that results in values of precision and recall of one, so we considered the relevance threshold as 2, the next possible value of rating. This problem has repercussions on the values of precision and recall because, as we observed in Chapter 3, 93.51% of the ratings have a value of 1 so, all the rating predictions that are bigger than 1 but smaller than two and should be considered relevant, are not.

In figures 4.3a and 4.3b, we can see the obtained results of precision for DisRM10 and DisRM20, respectively. All the algorithms resulted in a very low precision, between 0.05 and 0.15. On the contrary, we obtained higher values of recall. The best value obtained was with PIP applied to DisRM10 which reached 0.8. These values may indicate that we have few relevant items but that the majority of them is being recommended, which leads to a high recall (80%

Table 4.1: Overall comparison of DisRM10 and DisRM20 relatively to MAE, Coverage, Precision, Recall and nDCG.

	DisRM10	DISRM20
MAE	0.16047 (10 neighbors, PIP)	**0.16040** (10 neighbors, MSD)
Coverage	0.9964 (500 neighbors, PIP)	**0.9975** (500 neighbors, PIP)
Precision	0.137 (1 recommendation, JMSD)	**0.1513** (1 recommendation, JMSD)
Recall	**0.8101** (10 recommendations, PIP)	0.7267 (10 recommendations, PIP)
nDCG	**0.8732** (1 recommendation, PIP)	0.8269 (4 recommendation, PIP)

of the relevant are being recommended) and low values of precision (only about 10% of the recommended items are relevant because there were not many relevant items left to recommend).

As we analyze the f-measure values, and since we know that this metric is calculated based on precision and recall, we were expecting low values, but that is not the case: we have f-measure values of around 0.6. This incongruity is a consequence of the logic of CF4J. When both precision and recall have a value of zero, the f-measure is not processed and the algorithm does not consider that the element even exists in the final calculation of metrics.

4.2.1.4 Normalized Discounted Cumulative Gain

Normalized Discounted Cumulative Gain (nDCG) is a rank quality measure that quantifies the usefulness of the recommendations based on their position on the list assuming that highly relevant items are more useful when having a higher rank in the list and that highly relevant items are more useful than marginally relevant items, which are in turn more useful than non-relevant items. The result of this measure is in the interval 0 to 1, where 1 represents the perfect ranking algorithm. In both datasets, the algorithm that performed better was PIP, having a result of, approximately, 0.86 for DisRM10 and 0.81 for DisRM20.

As shown in table 4.1 and in figures 4.2, 4.3, 4.4, 4.5, 4.6, the algorithm that performed better overall was PIP. It delivered the highest recall and the highest nDCG which means that it is the algorithm that recommends the greatest amount of relevant items and with better ranking positions. In a context of research like this one, in which the most important task of a recommender system is to reduce the amount of information the user has to read, we could argue that the most important measure to optimize is the recall because a high recall is a guarantee that the users are being recommended most of the existing relevant items so that they can more easily find all the information that is important for their research. However, if we could improve precision, we would be able to reduce the number of irrelevant items that the user would have to go through to find the relevant ones. Comparing the performance of both datasets, we can see that, although they show very similar behaviour, DisRM10 performed better in both recall and nDCG.

4.2.2 DisRM Validation

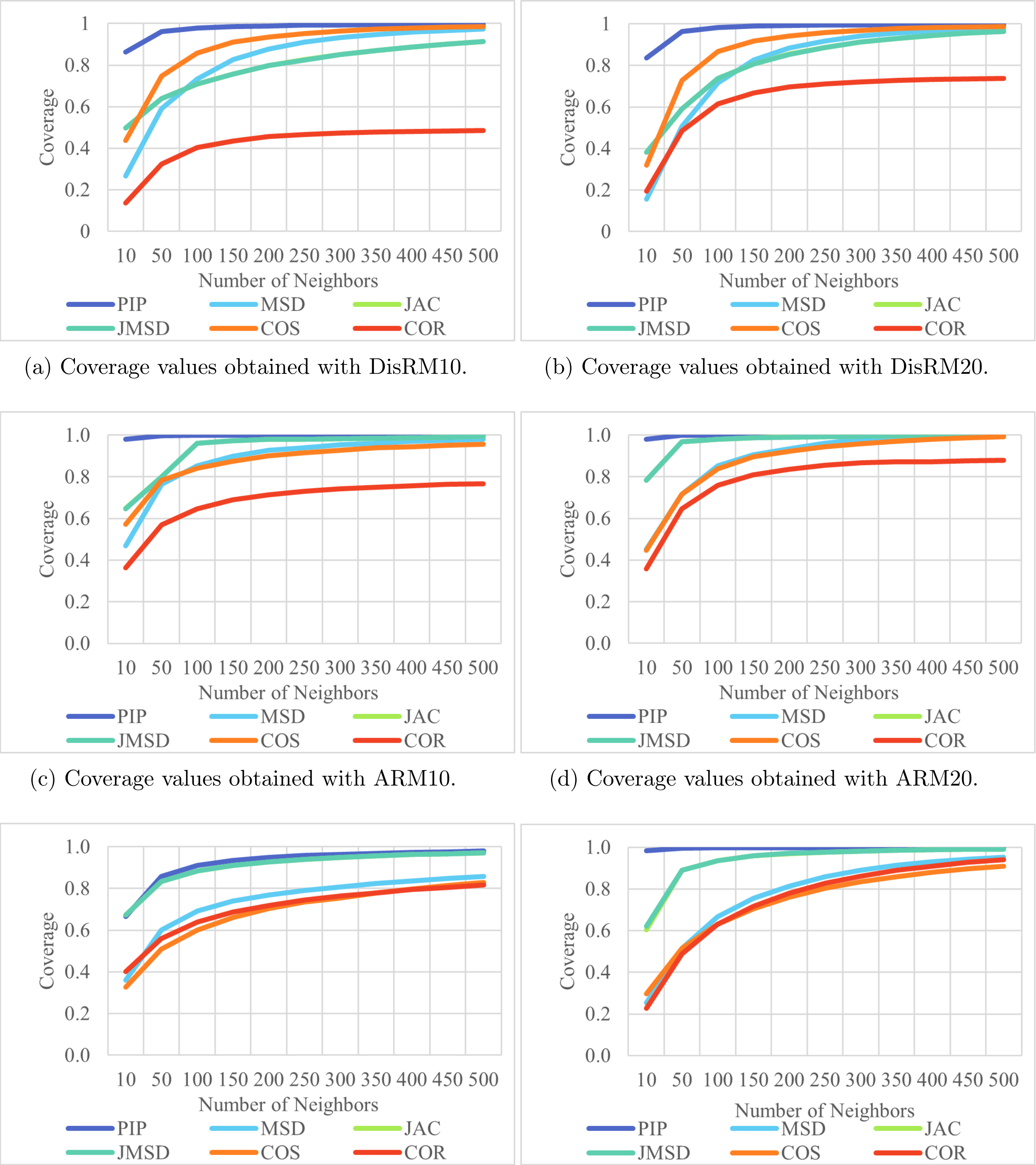
As mentioned earlier, to validate DisRM, we compared it to four baseline datasets: ML-1M, SD4AI, ARM10, and ARM20. We based the comparison on the same evaluation metrics used to evaluate the dataset, with the exception of MAE. Since MAE is in the same unit as the ratings, it is not comparable between datasets. In table 4.2, we can see the best result of each dataset for each metric, followed by the algorithm used to obtain it.

Table 4.2: Overall comparison of DisRM datasets with the 4 baseline datasets relatively to Coverage, Precision, Recall and nDCG.

	DisRM10	DISRM20	ARM10	ARM20	SD4AI	ML-1M
Coverage	0.996 (PIP)	0.998 (PIP)	1 (PIP)	1 (PIP)	0.979 (PIP)	0.999 (PIP)
Precision	0.137 (JMSD)	0.151 (JMSD)	0.420 (JAC)	0.726 (JAC)	0.790 (PIP)	0.985 (PIP)
Recall	0.810 (PIP)	0.727 (PIP)	0.914 (PIP)	0.909 (PIP)	0.600 (PIP)	0.598 (PIP)
nDCG	0.873 (PIP)	0.827 (PIP)	0.897 (PIP)	0.840 (PIP)	0.593 (PIP)	0.790 (PIP)

When analyzing table 4.2, we can see that, for precision, the DisRM results are lower than the baseline datasets but, as mentioned in the previous section, this is expected due to CF4J not being prepared for the implicit positive feedback of DisRM. For coverage, the DisRM results are very similar to the other datasets. For recall, our results were slightly lower than ARM but significantly higher than SD4AI and ML-1M. Finally, for nDCG, DisRM10 had better results that all the other datasets with the exception of ARM10.

These results validate that the DisRM dataset can be used for testing and evaluating recommender systems in the health field.



(a) Coverage values obtained with DisRM10.

(b) Coverage values obtained with DisRM20.

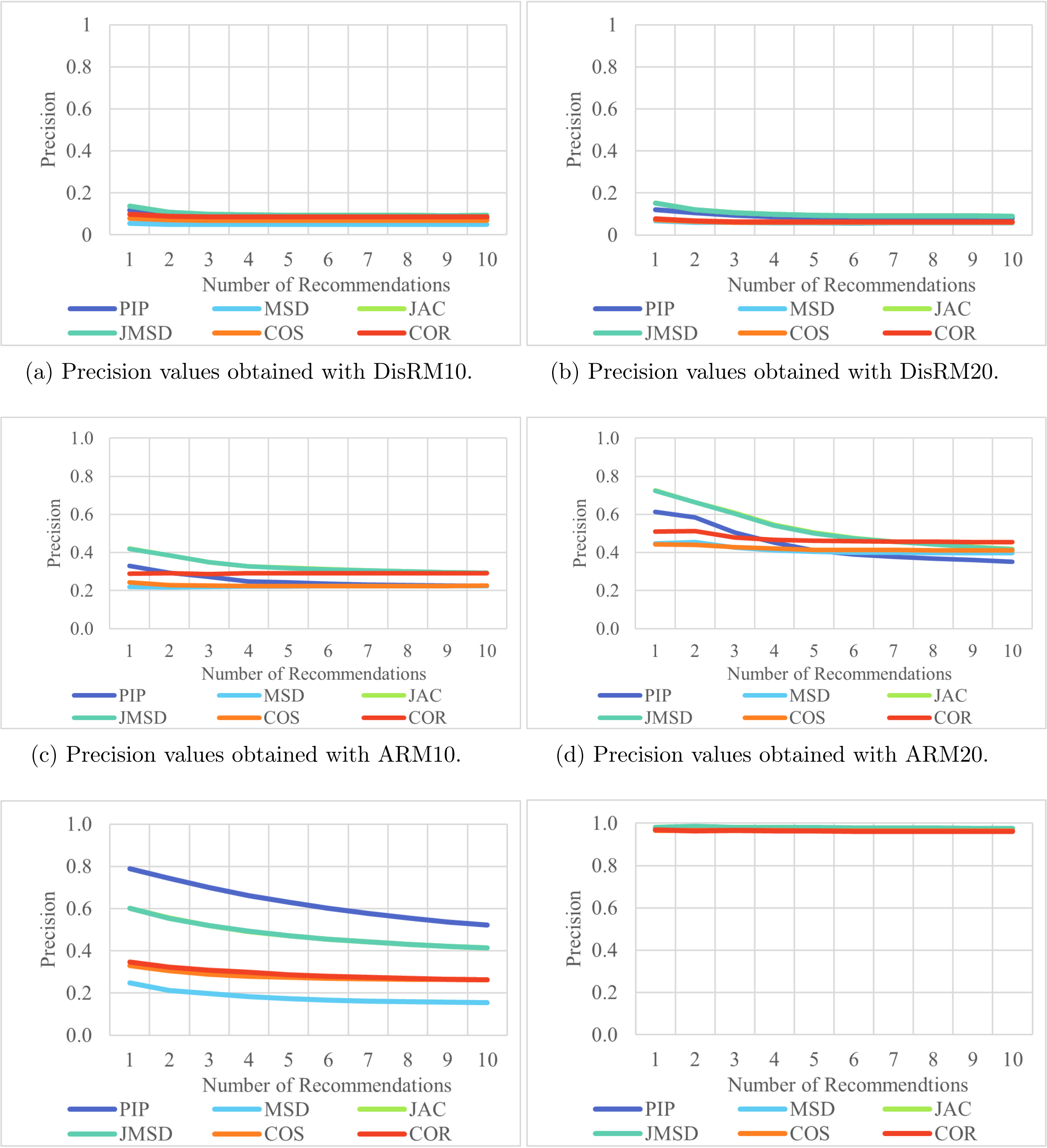
(c) Coverage values obtained with ARM10.

(d) Coverage values obtained with ARM20.

(e) Coverage values obtained with SD4AI.

(f) Coverage values obtained with ML-1M.

Figure 4.2: Visual analysis of coverage obtained by applying k-nearest neighbors algorithm to all datasets using six different similarity measures for the user similarity computation.



(a) Precision values obtained with DisRM10.

(b) Precision values obtained with DisRM20.

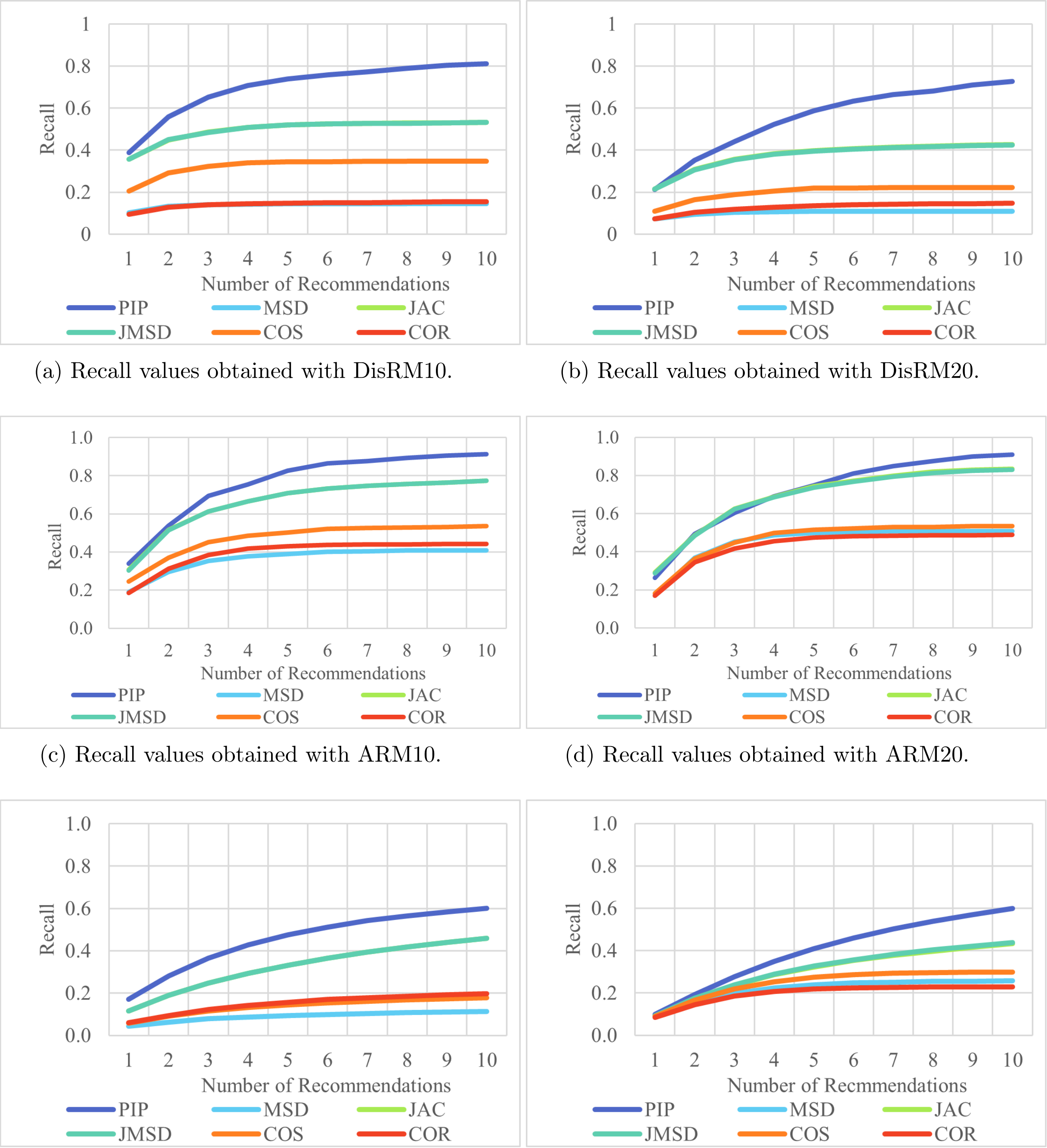
(c) Precision values obtained with ARM10.

(d) Precision values obtained with ARM20.

(e) Precision values obtained with SD4AI.

(f) Precision values obtained with ML-1M.

Figure 4.3: Visual analysis of precision obtained by applying k-nearest neighbors algorithm to all datasets using six different similarity measures for the user similarity computation.



(a) Recall values obtained with DisRM10.

(b) Recall values obtained with DisRM20.

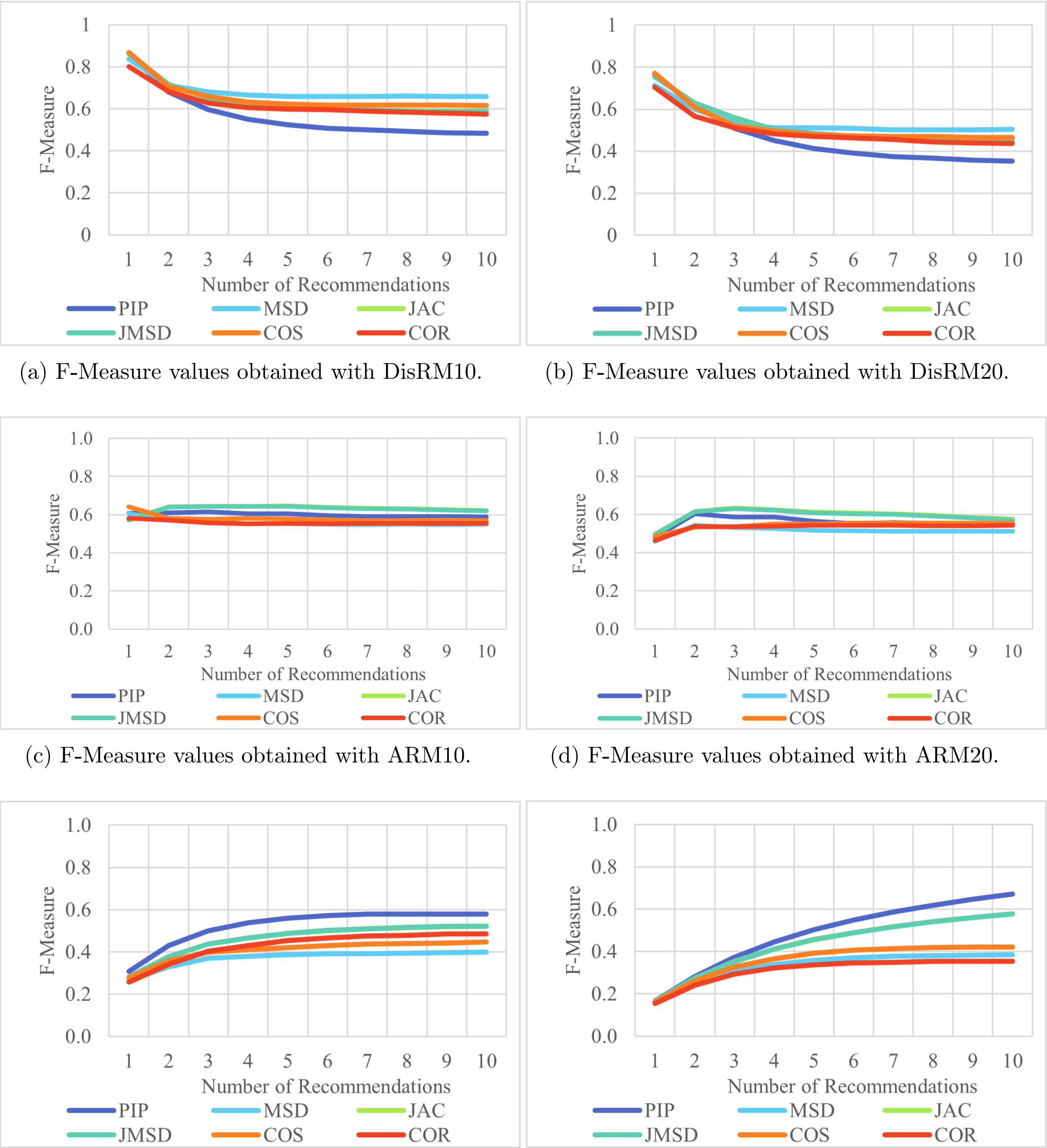
(c) Recall values obtained with ARM10.

(d) Recall values obtained with ARM20.

(e) Recall values obtained with SD4AI.

(f) Recall values obtained with ML-1M.

Figure 4.4: Visual analysis of recall obtained by applying k-nearest neighbors algorithm to all datasets using six different similarity measures for the user similarity computation.



(a) F-Measure values obtained with DisRM10.

(b) F-Measure values obtained with DisRM20.

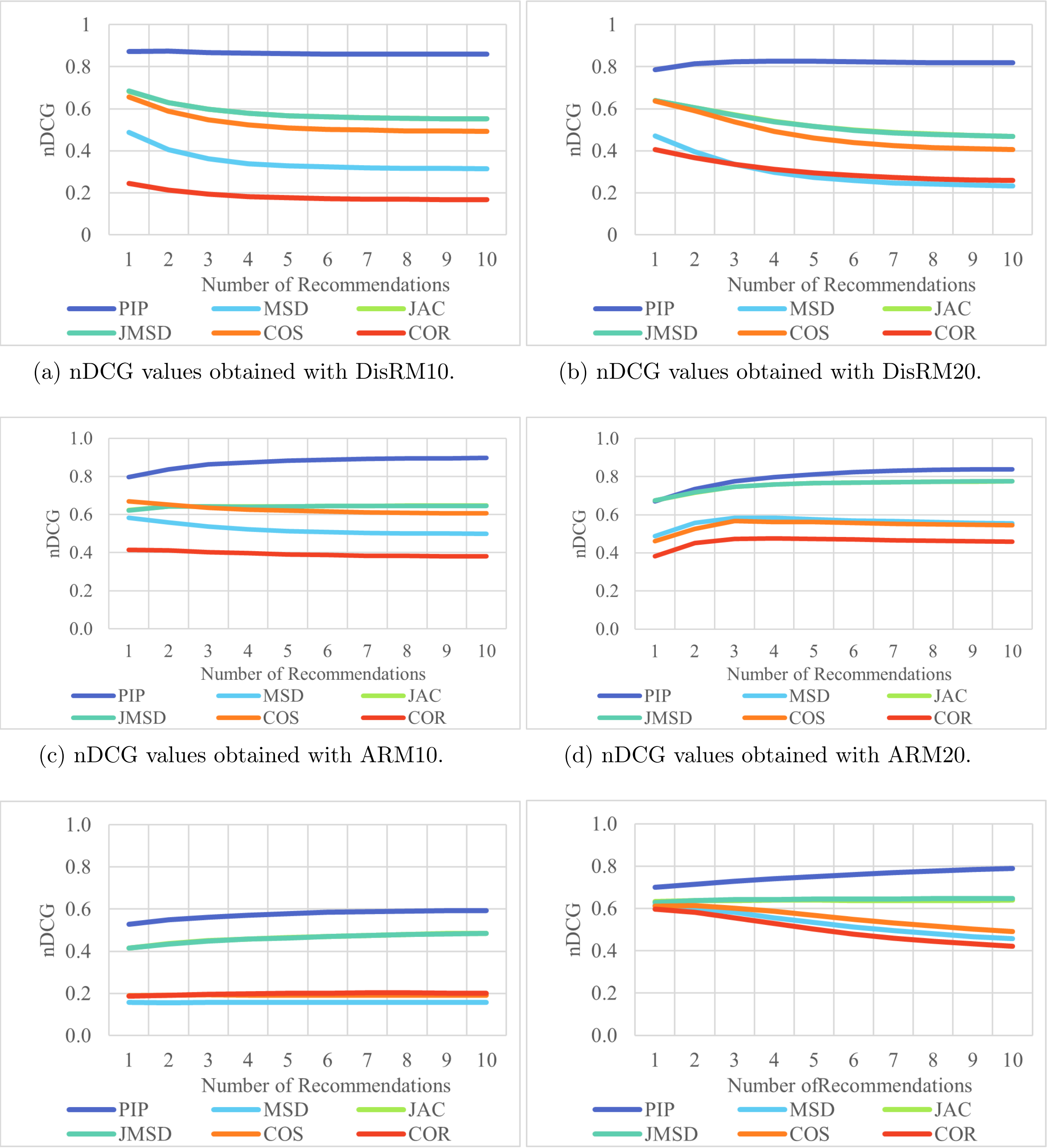
(c) F-Measure values obtained with ARM10.

(d) F-Measure values obtained with ARM20.

(e) F-Measure values obtained with SD4AI.

(f) F-Measure values obtained with ML-1M.

Figure 4.5: Visual analysis of f-measure obtained by applying k-nearest neighbors algorithm to all datasets using six different similarity measures for the user similarity computation.



(a) nDCG values obtained with DisRM10.

(b) nDCG values obtained with DisRM20.

(c) nDCG values obtained with ARM10.

(d) nDCG values obtained with ARM20.

(e) nDCG values obtained with SD4AI.

(f) nDCG values obtained with ML-1M.

Figure 4.6: Visual analysis of nDCG obtained by applying k-nearest neighbors algorithm to all datasets using six different similarity measures for the user similarity computation.

Chapter 5

Conclusion

RS are tools that provide personalized recommendations of items to users, using their preferences to show them relevant items. RS are an improvement of traditional information retrieval systems because they allow a new way of exploring large amounts of information. There are several applications of these systems in many fields in our day-to-day life like in streaming or e-commerce platforms. Also, in the health field, there is a need of applying new ways of exploring the growing amount of digital information. HRS appeared in 2007 but their development is still in its early stages, mostly due to the lack of open health datasets to test the recommender algorithms. Thus, this work had two main objectives: to create a dataset accessible to everyone with information about the preferences of users in the health field, allowing the testing and evaluation of health recommender systems, and to present a new health recommender system with the main final goal of contributing to the development of HRS.

To accomplish the first goal, we used the LIBRETTI methodology to create DisRM (Diseases Ratings Matrix), an open-source dataset of implicit feedback for the recommendation of diseases from scientific literature. In this dataset, the items to be recommended are the diseases, the users are the authors of the research articles and the ratings are the number of articles an author wrote about a disease. Additionally, we created two additional versions of the dataset to prevent the cold start problem for users, DisRM10 and DisRM20, that include only the users who have a number of ratings equal to or greater than 10 and 20, respectively. The DisRM datasets showed similar characteristics to the baseline datasets considered which shows they have the potential to be good RS datasets.

To achieve the second goal we applied a memory-based CF K-Nearest Neighbors algorithm to DisRM10 and DisRM20, testing several similarity metrics to evaluate which performed better. Considering our use case of recommending diseases to authors, we considered that the most important measures to optimize are recall and nDCG because, when we combine high results of both measures, they ensure that most of the relevant items are being recommended and ranked high. Thus, the algorithm that achieved the best recommendation results was PIP, obtaining a recall of 0.81 and a nDCG of 0.87 for DisRM10.

When comparing the performance of DisRM to the chosen baseline datasets for validation we observed that DisRM obtained comparable results to ARM and better results than SD4AI and ML-1M for recall and nDCG which validates the quality of our dataset.

On the one hand, these datasets allow the recommendation of diseases to authors, in order to help them find unknown, but relevant, diseases to their research that may be similar to their field of study and that they wouldn't find using regular information retrieval methods. On the other hand, DisRM is also useful for preliminary training and testing of health recommender systems if the type of information to be used in the future is private.

Additionally, we can conclude that a memory-based CF K-Nearest Neighbors algorithm using PIP proved to provide quality recommendations for health recommender systems.

5.1 Future Work

One improvement of this work would be to test DisRM with other frameworks other than CF4J to try to overcome the issue of CF4J not supporting only positive implicit feedback.

In the future, it would be interesting to explore Content Based algorithms for health recommender systems. Considering DisRM, this could be achieved by enriching the diseases with features extracted from the abstracts of the articles. For example, if an article about Asthma refers a specific gene in the abstract, we can say that that gene is a feature of Asthma. To search the features in the abstracts we could use MER (Couto *et al.* (2017)), which is a Named-Entity Recognition tool that, given any lexicon and any input text, returns all the terms of that lexicon that were recognized in the text. Different types of features could be used by using different ontologies. Some relevant examples are phenotypes from Human Phenotype Ontology (HPO) (Robinson & Mundlos (2010)), genes from Gene Ontology (GO) (Ashburner *et al.* (2000)), and Chemical Entities from Chemical Entities of Biological Interest (ChEBI) (Degtyarenko *et al.* (2007)).

Chapter 6

Appendices

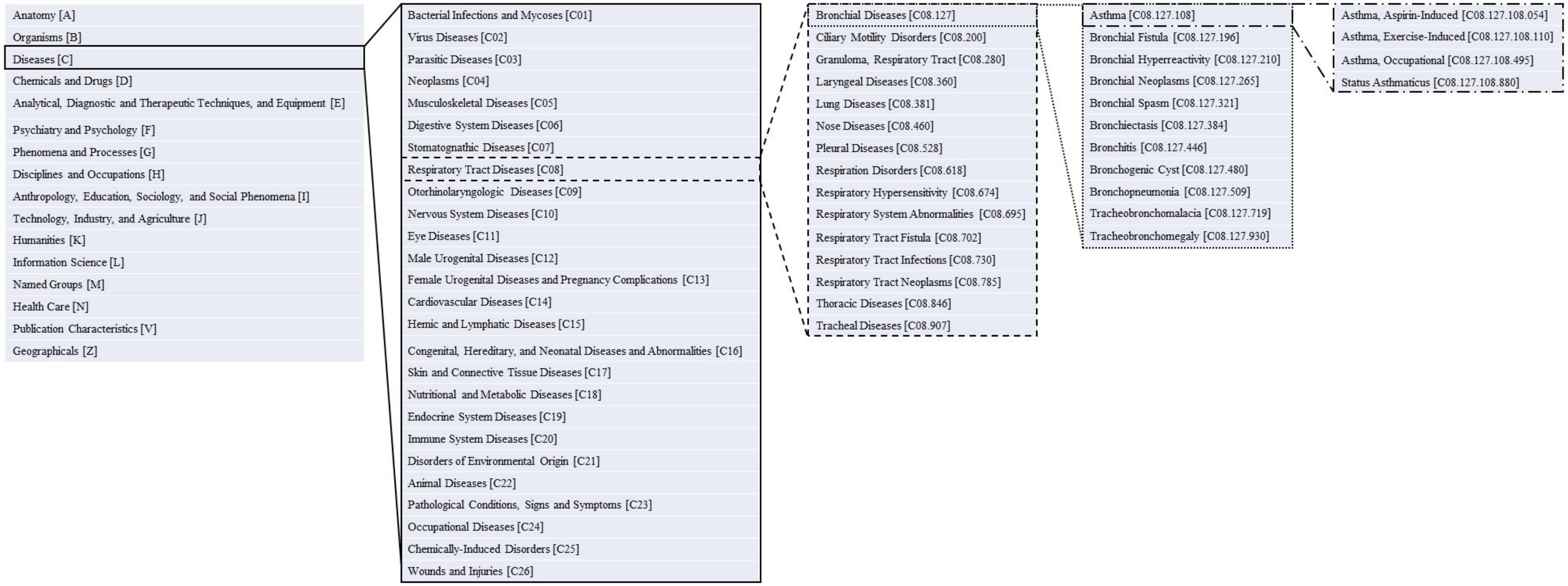

Anatomy [A]
Organisms [B]
Diseases [C]
Chemicals and Drugs [D]
Analytical, Diagnostic and Therapeutic Techniques, and Equipment [E]
Psychiatry and Psychology [F]
Phenomena and Processes [G]
Disciplines and Occupations [H]
Anthropology, Education, Sociology, and Social Phenomena [I]
Technology, Industry, and Agriculture [J]
Humanities [K]
Information Science [L]
Named Groups [M]
Health Care [N]
Publication Characteristics [V]
Geographicals [Z]
Bacterial Infections and Mycoses [C01]
Virus Diseases [C02]
Parasitic Diseases [C03]
Neoplasms [C04]
Musculoskeletal Diseases [C05]
Digestive System Diseases [C06]
Stomatognathic Diseases [C07]
Respiratory Tract Diseases [C08]
Otorhinolaryngologic Diseases [C09]
Nervous System Diseases [C10]
Eye Diseases [C11]
Male Urogenital Diseases [C12]
Female Urogenital Diseases and Pregnancy Complications [C13]
Cardiovascular Diseases [C14]
Hemic and Lymphatic Diseases [C15]
Congenital, Hereditary, and Neonatal Diseases and Abnormalities [C16]
Skin and Connective Tissue Diseases [C17]
Nutritional and Metabolic Diseases [C18]
Endocrine System Diseases [C19]
Immune System Diseases [C20]
Disorders of Environmental Origin [C21]
Animal Diseases [C22]
Pathological Conditions, Signs and Symptoms [C23]
Occupational Diseases [C24]
Chemically-Induced Disorders [C25]
Wounds and Injuries [C26]
Bronchial Diseases [C08.127]
Ciliary Motility Disorders [C08.200]
Granuloma, Respiratory Tract [C08.280]
Laryngeal Diseases [C08.360]
Lung Diseases [C08.381]
Nose Diseases [C08.460]
Pleural Diseases [C08.528]
Respiration Disorders [C08.618]
Respiratory Hypersensitivity [C08.674]
Respiratory System Abnormalities [C08.695]
Respiratory Tract Fistula [C08.702]
Respiratory Tract Infections [C08.730]
Respiratory Tract Neoplasms [C08.785]
Thoracic Diseases [C08.846]
Tracheal Diseases [C08.907]
Asthma [C08.127.108]
Bronchial Fistula [C08.127.196]
Bronchial Hyperreactivity [C08.127.210]
Bronchial Neoplasms [C08.127.265]
Bronchial Spasm [C08.127.321]
Bronchiectasis [C08.127.384]
Bronchitis [C08.127.446]
Bronchogenic Cyst [C08.127.480]
Bronchopneumonia [C08.127.509]
Tracheobronchomalacia [C08.127.719]
Tracheobronchomegaly [C08.127.930]
Asthma, Aspirin-Induced [C08.127.108.054]
Asthma, Exercise-Induced [C08.127.108.110]
Asthma, Occupational [C08.127.108.495]
Status Asthmaticus [C08.127.108.880]

Figure 6.1: Schematic representation of a portion of MeSH that shows its hierarchical structure.

References

Agarwal, A. & Chauhan, M. (2017). Similarity measures used in recommender systems: a study. *International Journal of Engineering Technology Science and Research IJETSR, ISSN*, 2394–3386. 17, 18

Aggarwal, C.C. (2016). Evaluating recommender systems. In *Recommender systems*, 225–254, Springer. 5

Ahn, H.J. (2008). A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem. *Information Sciences*, **178**, 37–51. 4, 18

The long tail. 13

Anderson, C. (2006). *The Long Tail - Why the Future of Business Is Selling Less of More*. Hyperion. 14

Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T. *et al.* (2000). Gene ontology: tool for the unification of biology. *Nature genetics*, **25**, 25–29. 30

Barros, M., Moitinho, A. & Couto, F.M. (2019). Using research literature to generate datasets of implicit feedback for recommending scientific items. *IEEE Access*, **7**, 176668–176680. 6, 9, 18

Bobadilla, J., Serradilla, F. & Bernal, J. (2010). A new collaborative filtering metric that improves the behavior of recommender systems. *Knowledge-Based Systems*, **23**, 520–528. 4, 18

Bobadilla, J., Ortega, F., Hernando, A. & Gutiérrez, A. (2013). Recommender systems survey. *Knowledge-based systems*, **46**, 109–132. 5

Bossard, L., Guillaumin, M. & Van Gool, L. (2014). Food-101–mining discriminative components with random forests. In *European conference on computer vision*, 446–461, Springer. 8

Chen, J.H., Podchiyska, T. & Altman, R.B. (2015). Orderrex: clinical order decision support and outcome predictions by data-mining electronic medical records. *Journal of the American Medical Informatics Association*, **23**, 339–348. 8

Couto, F.M., Campos, L.F. & Lamurias, A. (2017). Mer: a minimal named-entity recognition tagger and annotation server. *Proceedings of the BioCreative*, **5**, 130–137. 30

Degtyarenko, K., De Matos, P., Ennis, M., Hastings, J., Zbinden, M., McNaught, A., Alcántara, R., Darsow, M., Guedj, M. & Ashburner, M. (2007). Chebi: a database and ontology for chemical entities of biological interest. *Nucleic acids research*, **36**, D344–D350. 30

Goldberg, D., Nichols, D., Oki, B.M. & Terry, D. (1992). Using collaborative filtering to weave an information tapestry. *Communications of the ACM*, **35**, 61–70. 1

Harper, F.M. & Konstan, J.A. (2016). The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, **5**, 19. 13

Liu, H., Hu, Z., Mian, A., Tian, H. & Zhu, X. (2014). A new user similarity model to improve the accuracy of collaborative filtering. *Knowledge-Based Systems*, **56**, 156–166. 17

Mohamed, M.H., Khafagy, M.H. & Ibrahim, M.H. (2019). Recommender systems challenges and solutions survey. In *2019 International Conference on Innovative Trends in Computer Engineering (ITCE)*, 149–155, IEEE. xv, 4

NCBI (2010). *Entrez Programming Utilities Help [Internet]*. National Center for Biotechnology Information. 10

Ortega, F., Bobadilla, J., Gutiérrez, A., Hurtado, R. & Li, X. (2018a). Artificial intelligence scientific documentation dataset for recommender systems. *IEEE Access*, **6**, 48543–48555. 4, 6, 18, 20

Ortega, F., Zhu, B., Bobadilla, J. & Hernando, A. (2018b). Cf4j: Collaborative filtering for java. *Knowledge-Based Systems*, **152**, 94–99. 17

7 brilliant examples of brands driving long-tail organic traffic. 14

Robinson, P.N. & Mundlos, S. (2010). The human phenotype ontology. *Clinical genetics*, **77**, 525–534. 30

Sanchez, J., Serradilla, F., Martinez, E. & Bobadilla, J. (2008). Choice of metrics used in collaborative filtering and their impact on recommender systems. In *2008 2nd IEEE International Conference on Digital Ecosystems and Technologies*, 432–436, IEEE. 18

Suphavilai, C., Bertrand, D. & Nagarajan, N. (2018). Predicting cancer drug response using a recommender system. *Bioinformatics*, **34**, 3907–3914. 8

Torkaman, A., Charkari, N.M. & Aghaeipour, M. (2011). An approach for leukemia classification based on cooperative game theory. *Analytical Cellular Pathology*, **34**, 235–246. 7

Valdez, A.C., Ziefle, M., Verbert, K., Felfernig, A. & Holzinger, A. (2016). Recommender systems for health informatics: state-of-the-art and future perspectives. In *Machine Learning for Health Informatics*, 391–414, Springer. 1

Wiesner, M. & Pfeifer, D. (2014). Health recommender systems: concepts, requirements, technical basics and challenges. *International journal of environmental research and public health*, **11**, 2580–2607. 7

Yang, L., Hsieh, C.K., Yang, H., Pollak, J.P., Dell, N., Belongie, S., Cole, C. & Estrin, D. (2017). Yum-me: a personalized nutrient-based meal recommender system. *ACM Transactions on Information Systems (TOIS)*, **36**, 7. 8