PGCOMP - Programa de Pós-Graduação em Ciência da Computação
Universidade Federal da Bahia (UFBA)
Av. Adhemar de Barros, s/n - Ondina
Salvador, BA, Brasil, 40170-110

http://pgcomp.dcc.ufba.br
pgcomp@ufba.br

Sistemas de recomendação são ferramentas utilizadas para sugerir itens, que possivelmente sejam de interesse dos usuários. Estes sistemas baseiam-se no histórico de preferências do usuário para gerar uma lista de sugestões que possuam maior similaridade com o perfil do usuário, visando uma melhor precisão e um menor erro. É esperado que, ao ser recomendado um item, o usuário informe sua preferência ao sistema, indicando se gostou ou o quanto gostou do item recomendado. A interação do usuário com o sistema possibilita um melhor entendimento de seus gostos, que com o tempo, adiciona mais e mais itens a seu perfil de preferências. A recomendação baseada em similaridade do item com as preferências buscando a melhor precisão pode causar efeitos colaterais na lista como: superespecialização das recomendações em um determinado núcleo de itens, pouca diversidade de categorias e desbalanceamento de categoria ou gênero. Assim, esta dissertação tem como objetivo explorar a calibragem, que é um meio para produzir recomendações que sejam relevantes aos usuários e ao mesmo tempo considerar todas as áreas de suas preferências, buscando evitar a desproporção na lista de recomendação. Para isto, foram abordadas formas de ponderar o balanceamento entre a relevância das recomendações e a calibragem baseada em medidas de divergência, assim como um modelo de sistema calibrado e um protocolo de decisão. A hipótese é que a calibragem pode contribuir positivamente para recomendações mais justas de acordo com a preferência do usuário. A pesquisa foi realizada através de uma ampla abordagem propondo um modelo de sistema e um protocolo de decisão que contempla em seu experimento nove algoritmos de recomendação aplicados nos domínios de filme e música, analisando três medidas de divergência, dois pesos de balanceamento personalizado e dois balanceamentos entre relevância-calibragem. A avaliação foi analisada com métricas amplamente utilizadas, assim como métricas propostas neste trabalho. Os resultados indicam que a calibragem produz efeitos positivos tanto para a precisão da recomendação quanto para a justiça com as preferências do usuário, criando listas de recomendação que respeitem todas as áreas. Os resultados também indicam qual é a melhor combinação para obter um melhor desempenho ao aplicar as propostas de calibragem.

MSC | 116 | 2021

Explorando Calibragem Ponderada, Balanceamentos e Métricas para Justiça em Sistemas de Recomendação

Diego Corrêa da Silva

UFBA

Explorando Calibragem Ponderada, Balanceamentos e Métricas para Justiça em Sistemas de Recomendação

Diego Corrêa da Silva

Dissertação de Mestrado

Universidade Federal da Bahia

Programa de Pós-Graduação em
Ciência da Computação

Julho | 2021

Universidade Federal da Bahia
Instituto de Matemática e Estatística

Programa de Pós-Graduação em Ciência da Computação

EXPLORANDO CALIBRAGEM PONDERADA, BALANCEAMENTOS E MÉTRICAS PARA JUSTIÇA EM SISTEMAS DE RECOMENDAÇÃO

Diego Corrêa da Silva

DISSERTAÇÃO DE MESTRADO

Salvador
01 de Julho de 2021

DIEGO CORRÊA DA SILVA

EXPLORANDO CALIBRAGEM PONDERADA, BALANCEAMENTOS E MÉTRICAS PARA JUSTIÇA EM SISTEMAS DE RECOMENDAÇÃO

Esta Dissertação de Mestrado foi apresentada ao Programa de Pós-Graduação em Ciência da Computação da Universidade Federal da Bahia, como requisito parcial para obtenção do grau de Mestre em Ciência da Computação.

Orientador: Frederico Araújo Durão

Salvador
01 de Julho de 2021

Ficha catalográfica elaborada pela Biblioteca Universitária de Ciências e Tecnologias Prof. Omar Catunda, SIBI - UFBA.

S856 Silva, Diego Corrêa da

Explorando calibragem ponderada, balanceamentos e métricas para justiça em sistemas de recomendação/ Diego Corrêa da Silva – Salvador, 2021.

137 f.

Orientador: Prof. Dr. Frederico Araújo Durão.

Dissertação (Mestrado) – Universidade Federal da Bahia. Instituto de Matemática e Estatística, 2021.

1. Ciência da Computação. 2. Métrica. 3. Sistema Especialista. I. Durão, Frederico Araújo. II. Universidade Federal da Bahia. III Título.

CDU 004

Ata da Sessão Pública de Defesa de **Mestrado** nº. **116**
Colegiado do Programa de Pós-Graduação em Ciência da Computação

Ata da sessão pública do Colegiado do Programa de Pós-Graduação em Ciência da Computação, realizada em **01 de julho de 2021** para procedimento de defesa da **Dissertação** de **Mestrado** em Ciência da Computação nº. **116**, linha de pesquisa **Engenharia de Software**, d**o** candidat**o Diego Corrêa da Silva**, matrícula **2020104347**, intitulada "***Explorando Calibragem Ponderada, Balanceamentos e Métricas para Justiça em Sistemas de Recomendação***". Às **dez horas** do citado dia, via **webconferência**, foi aberta a sessão pelo presidente da banca examinadora **Profº. Drº. Frederico Araújo Durão(Orientador-UFBA)** que apresentou os outros membros da banca: **Prof. Dr. Leandro Balby Marinho (UFCG)**, e **Profª. Drª. Tatiane Nogueira Rios (DCC/UFBA)**. Em seguida foram esclarecidos os procedimentos pelo presidente que passou a palavra ao examinado para apresentação do trabalho de **Mestrado**. Ao final da apresentação, passou-se à arguição por parte da banca, a qual, em seguida, reuniu-se para a elaboração do parecer. No seu retorno, foi lido o parecer final a respeito do trabalho apresentado pel**o** candidat**o**, tendo a banca examinadora **aprovado** o trabalho apresentado, sendo esta aprovação um requisito parcial para a obtenção do grau de **Mestre**. Em seguida, nada mais havendo a tratar, foi encerrada a sessão pelo presidente da banca, tendo sido, logo a seguir, lavrada a presente ata, abaixo assinada por todos os membros da banca.

Salvador, 01 de julho de 2021

Profº. Drº. Frederico Araújo Durão(Orientador-UFBA)

Prof. Dr. Leandro Balby Marinho (UFCG)

Profª. Drª. Tatiane Nogueira Rios (DCC/UFBA)

Dedico esta monografia à minha mãe que sempre me permitiu meios para estudar, a minha companheira, amigos e professores que me deram todo o apoio necessário para chegar até aqui.

AGRADECIMENTOS

Agradeço primeiramente à minha mãe, Norma Lúcia Corrêa Santos, por todo o apoio ao longo da vida. Ela que é pedagoga de formação e criou desde cedo um cientista. Ao longo dos seus anos de ensino foi minha professora por diversas ocasiões e que em diversos momentos teve que separar o profissional do familiar.

Agradeço em especial ao orientador, Prof. Dr. Frederico Araújo Durão, que não poupou esforços para me apoiar nessa jornada. Esta dissertação que é o segundo trabalho acadêmico finalizado em parceria. Obrigado por fazer parte da construção desta nova etapa, por todo o conhecimento transmitido e pela confiança de acreditar nas ideias que foram postas em prática aqui.

Este trabalho não teria a qualidade que possui sem a dedicação de Danille M. H. L. da Gama, que leu parágrafo por parágrafo diversas vezes ao longo da construção da dissertação. Diversos erros de escrita seriam encontrados se não fosse a dedicação dela em se ater a cada vírgula. Agradeço não só pela correção da dissertação, mas também pela correção da gramática inglesa do artigo publicado a partir desta dissertação. Assim como agradeço pelos anos de relacionamento que possuímos.

Agradeço ao Prof. Dr. Marcelo Garcia Manzato, que contribuiu com diversas observações na construção desta dissertação. Observações estas que renderam uma parceria para a publicação do artigo.

Ao amigo Paulo Roberto de Souza, que possibilitou melhorias nesta dissertação, agradeço pelos comentários e contribuições no debate. Como também a todos que conheci através do Grupo de Pesquisa RecSys UFBA, do qual faço parte.

O presente trabalho foi realizado com apoio de diversos colaboradores, os quais possuem nossos agradecimentos e são reconhecidos a seguir. A Coordenação de Aperfeiçoamento de Pessoal de Nível Superior - Brasil (CAPES) - fomentou esta pesquisa sob o Código de Financiamento 88887.502736/2020-00. Os resultados encontrados, para a qualificação e publicação de artigo, possuíram o apoio do Núcleo de Biologia Computacional e Gestão de Informações Biotecnológicas - NBCGIB, com recursos FINEP/MCT, CNPQ e FAPESB e da Universidade Estadual de Santa Cruz - UESC, que nos cedeu acesso ao computador de alto desempenho. O Laboratório Nacional de Computação Científica (LNCC/MCTI, Brazil), URL: http://sdumont.lncc.br, nos concedeu acesso aos recursos do computador de alto desempenho chamado SDumont, que permitiu obter os resultados da pesquisa contidos nesta dissertação.

E por fim, mas não menos importante, ao Departamento de Ciência da Computação da Universidade Federal da Bahia. Este que me formou como egresso no curso de graduação em Ciência da Computação e com esta dissertação me forma como egresso do mestrado em Ciência da Computação.

Determine, rapaz. Onde vai ser seu curso de pós-graduação. Se oriente, rapaz. Pela rotação da Terra em torno do Sol.

—GILBERTO GIL (Oriente)

RESUMO

Sistemas de recomendação são ferramentas utilizadas para sugerir itens, que possivelmente sejam de interesse dos usuários. Estes sistemas baseiam-se no histórico de preferências do usuário para gerar uma lista de sugestões que possuam maior similaridade com o perfil do usuário, visando uma melhor precisão e um menor erro. É esperado que, ao ser recomendado um item, o usuário informe sua preferência ao sistema, indicando se gostou ou o quanto gostou do item recomendado. A interação do usuário com o sistema possibilita um melhor entendimento de seus gostos, que com o tempo, adiciona mais e mais itens a seu perfil de preferências. A recomendação baseada em similaridade do item com as preferências buscando a melhor precisão pode causar efeitos colaterais na lista como: superespecialização das recomendações em um determinado núcleo de itens, pouca diversidade de categorias e desbalanceamento de categoria ou gênero. Assim, esta dissertação tem como objetivo explorar a calibragem, que é um meio para produzir recomendações que sejam relevantes aos usuários e ao mesmo tempo considerar todas as áreas de suas preferências, buscando evitar a desproporção na lista de recomendação. Para isto, foram abordadas formas de ponderar o balanceamento entre a relevância das recomendações e a calibragem baseada em medidas de divergência, assim como um modelo de sistema calibrado e um protocolo de decisão. A hipótese é que a calibragem pode contribuir positivamente para recomendações mais justas de acordo com a preferência do usuário. A pesquisa foi realizada através de uma ampla abordagem propondo um modelo de sistema e um protocolo de decisão que contempla em seu experimento nove algoritmos de recomendação aplicados nos domínios de filme e música, analisando três medidas de divergência, dois pesos de balanceamento personalizado e dois balanceamentos entre relevância-calibragem. A avaliação foi analisada com métricas amplamente utilizadas, assim como métricas propostas neste trabalho. Os resultados indicam que a calibragem produz efeitos positivos tanto para a precisão da recomendação quanto para a justiça com as preferências do usuário, criando listas de recomendação que respeitem todas as áreas. Os resultados também indicam qual é a melhor combinação para obter um melhor desempenho ao aplicar as propostas de calibragem.

Palavras-chave: Calibragem. Justiça. Métricas. Personalização. Recomendação.

ABSTRACT

Recommendation Systems are tools used to suggest items, which possibly will be interesting to the users. These systems are based on the user's preference historic to generate a suggestion list that has higher similarity with the items in the user's historic, providing a better precision and minor error. It is expected that when an item is recommended, the user will send a feedback, indicating if he/she likes or dislikes the recommendation. The user interaction with the system provides a better way to understand the preferences of the user, who, over time, adds more and more itens in its profile. The recommendation based on item similarity with the user preference searching for better precision, can cause collateral effects in the recommendation list, such as: superspecialization of the items in the list with a little kernel of items, few diversity of categories and unbalanced genres. Thus, this dissertation aims to explore the calibration, which is a way to produce recommendations that will be relevant to the users and at the same time considering all genres in the preferences, trying to avoid a misproportion in the recommendation list. To this end, ways are addressed to weight the tradeoff between the relevance of the recommendations and the calibration based on divergence measures, as well as a calibrated system framework and decision protocol. The hypothesis are that calibration can contribute positively to provide more fairer recommendations. The research is accomplished with a large approach proposing a framework and a decision protocol that contemplates nine recommender algorithms applied on the domains of movie and music, analysing three divergence measures, two personalized tradeoff weights and two tradeoff formulations between relevance-calibration. The evaluation is analysed with well-known metrics, as well as proposed metrics. The results indicate that the calibration produces positive effects to the precision and to the fairness, creating a recommendation list that respects the genre distribution. The results also indicate which is the best system combination to obtain a better performance when applying the proposed calibration.

Keywords: Calibration. Fairness. Metrics. Personalization. Recommendation.

SUMÁRIO

LISTA DE FIGURAS

LISTA DE TABELAS

LISTA DE SIGLAS

CCE Coefficient of Calibration Error . . . 66
CMC Coefficient of MisCalibration . . . 66
IR Information Retrieval . . . 40
KL Kullback-Leibler . . . 100
KNN K Nearest Neighbors . . . 127
LDA Latent Dirichlet Allocation . . . 39
MACE Mean Average Calibration Error . . . 124
MAE Mean Absolute Error . . . 82
MAP Mean Average Precision . . . 124
MMR Maximum Marginal Relevance . . . 85
MRMC Mean Rank Miss Calibration . . . 124
MRR Mean Reciprocal Rank . . . 124
MSD Mean Squared Difference . . . 83
nDCG normalized Discounted Cumulative Gain . . . 54
NMF Non-negative Matrix Factorization . . . 89
OMS One Million Songs . . . 127
SPAD Subprofile-Aware Diversifcation . . . 40
SVD Singular Value Decomposition . . . 89
SVD++ Singular Value Decomposition Plus Plus . . . 127
xQuAD Query Aspect Diversifcation . . . 40

Capítulo 1

INTRODUÇÃO

Com o advento da Web 2.0 e seu rápido crescimento na quantidade de informação disponível, vários novos tipos de serviços surgiram, estes que, frequentemente, sobrecarregam o usuário (filmes, músicas, notícias, etc.), devido à grande quantidade de possibilidades. Com tantos serviços disponíveis para serem escolhidos, a busca por novos itens requer do usuário um esforço e atenção, podendo, o usuário, tomar decisões precipitadas, levando-o, muitas vezes, a encontrar itens que não o agradem. Assim, a disponibilidade de escolhas ao invés de ser um benefício acaba sendo um problema para este usuário (HIJIKATA; IWAHAMA; NISHIDA, 2006; RICCI; ROKACH; SHAPIRA, 2011). Portanto, para auxiliar os usuários na solução desse problema, surgem os Sistemas de Recomendação com o objetivo de filtrar informações relevantes apresentando uma lista ranqueada de itens de acordo com suas preferências, possibilitando assim a tomada de decisão.

De acordo com Ricci, Rokach e Shapira (2011), "Sistemas de Recomendação são ferramentas e técnicas que proveem sugestões de itens para os usuários. Estas sugestões são encontradas através de comparações entre itens e/ou usuários". Fields (2011) define Sistemas de Recomendação como "uma técnica ou método que apresenta a um usuário objetos sugeridos para consumo com base em seu comportamento passado".

Diversas são as técnicas utilizadas nos Sistemas de Recomendação, dentre as quais três se destacam: a Filtragem Colaborativa, a Filtragem Baseada em Conteúdo e a Filtragem Híbrida (RICCI; ROKACH; SHAPIRA, 2011). Na Filtragem Colaborativa o sistema utiliza as notas e/ou comportamentos dos usuários para gerar recomendações. Esta técnica é uma das mais utilizadas na área e possui vantagens como: a simplicidade na implementação, a justificabilidade no momento de entender como as recomendações foram encontradas, dentre outros; assim como possui desvantagens, por exemplo: o enviesamento por popularidade, a entrada de novo usuário no sistema (partida a frio), recomendação de novos itens cadastrados no sistema, dentre outras (DESROSIERS; KARYPIS, 2011; ISINKAYE; FOLAJIMI; OJOKOH, 2015). A Filtragem Baseada em Conteúdo utiliza as informações que descrevem os itens presentes nas preferências de um usuário para encontrar itens que são o máximo similares. Esta técnica é também uma das mais utilizadas e possui vantagens como: rápida adaptação a mudanças nas preferências do usuário, fácil

explanabilidade, dentre outras; mas também possui desvantagens como: a superespecialização nas preferências do usuário, total dependência dos metadados que descrevem os itens, dentre outras (LOPS; GEMMIS; SEMERARO, 2011; ISINKAYE; FOLAJIMI; OJOKOH, 2015). Uma terceira técnica popular é a Híbrida, a qual utiliza duas ou mais técnicas em conjunto para encontrar as recomendações (BURKE, 2002).

Na literatura, Sistemas de Recomendação têm sido alvo de pesquisas nas mais variadas vertentes: explicação de como as recomendações foram geradas (MITTELSTADT; RUSSELL; WACHTER, 2019); novo usuário e/ou item no sistema, problema conhecido como partida a frio (*Cold-Start*) (BARJASTEH et al., 2015); uso de dados ligados e abertos na Internet para aumentar as informações sobre os itens (PESKA; VOJTAS, 2013); diversidade dos itens que compõem as recomendações (CHENG et al., 2017); itens populares e enviesamento por popularidade (ABDOLLAHPOURI; BURKE; MOBASHER, 2017); justiça nas recomendações (STECK, 2018), dentre outros.

Em particular, neste trabalho é apresentado um estudo sobre justiça na recomendação através da calibragem, que visa gerar recomendações personalizadas a um usuário, garantindo um balanceamento mais justo dos itens por gêneros ou classes associadas a estes. Por exemplo, se as preferências de um usuário são compostas por 70% de filmes de sci-fi e 30% de filmes de ação, espera-se que a recomendação venha a ser calibrada em uma proporção que seja o máximo convergente com estes valores.

1.1 MOTIVAÇÃO

Sistemas de Recomendação têm como base predizer se um usuário irá preferir ou não um determinado item e qual o nível desta preferência (RICCI; ROKACH; SHAPIRA, 2011). Eles buscam ser precisos no acerto, tanto que as métricas de precisão e erro são as mais utilizadas para entender se o sistema está desempenhando corretamente sua tarefa (MCNEE; RIEDL; KONSTAN, 2006). Entretanto, os usuários não desejam somente precisão, se um usuário tem preferência sobre um determinado gênero de itens não significa que o sistema tenha que recomendar somente itens similares ao gênero de maior preferência e ignorar as demais preferências (STECK, 2018).

McNee, Riedl e Konstan (2006) apresentam em seu estudo um alerta sobre os sistemas de recomendação que focam apenas em precisão. Os autores abordam novos olhares sobre a recomendação, indicando alguns caminhos a serem seguidos. Após este alerta diversos estudos buscaram avaliar seus sistemas para além da precisão, incluindo perspectivas como: diversidade, surpresa, novidade, inesperado, dentre outros. Recentemente, diversas comunidades começaram a questionar se os sistemas computacionais são justos (ZLIOBAITE, 2015; HARDT; PRICE; SREBRO, 2016).

Ao focar em obter a melhor precisão e o menor erro na predição o sistema tende a recomendar itens que venham a abranger o gênero de maior dominância nas preferências dos usuários e em alguns sistemas, como os que usam a técnica de filtragem colaborativa, existe a possibilidade das recomendações sofrerem enviesamento por popularidade, encontrando recomendações que agradem o gênero de maior dominância do usuário de acordo com o que é mais popular dentre os outros usuários. Essa recomendação pode vir a ser precisa em vários momentos, entretanto causa um ciclo ao recomendar somente

o gênero dominante, tornando-o mais dominante ainda e baseando-se no que é popular entre os usuários, levando os itens a serem mais populares ainda.

Recentes e importantes estudos da literatura abordam o tema de justiça, focando em um aspecto em particular que busca gerar recomendações concisas com os gêneros/classes que compõem as preferências dos usuários, este que são: Steck (2018) que realiza uma investigação sobre como gerar recomendações justas. Ele apresenta uma formulação para a técnica de Filtragem Colaborativa, em que visa calibrar as recomendações seguindo a proporção dos gêneros que pertencem às preferências dos usuários; Kaya e Bridge (2019) que verificam o efeito da calibragem sobre a diversidade dos itens na lista de recomendação; e Abdollahpouri et al. (2020) que abordam a descalibragem entre o histórico de preferências do usuário e o que é recomendado pelos algoritmos comumente utilizados. Lin et al. (2020) analisam em diversos fatores o efeito da descalibragem da lista de recomendação, efeito que acontece para alguns usuários e para outros não, buscando observar as características do usuário e quais dos fatores influenciam na descalibragem.

Um exemplo motivacional é, supondo que as preferências de um usuário sejam compostas por músicas 60% de Rock, 20% de Samba e 20% de Mangue-Beat, assim temos que o gênero de maior preferência é o Rock. Ademais, supomos que a aplicação possua em sua base de dados músicas dos mais diferentes gêneros, mas que em sua maioria sejam pertencentes ao Rock. Assim, temos uma tendência das músicas mais populares serem do Rock e em acréscimo temos que a própria preferência do usuário é composta por maioria de Rock. Isto causa um efeito bolha entre o usuário e a aplicação (KAMISHIMA; AKAHO; ASOH, 2012), em que apenas um gênero é popular e constantemente recomendado.

1.2 PROBLEMA

O problema principal que esta dissertação investigou foi a utilização do aspecto de calibragem para promover justiça na lista de recomendação, tomando como base a ausência de mecanismos que personalizem e avaliem o grau de relevância e calibragem que a lista possui com as preferências do usuário. Para isso, o problema principal foi dividido em quatro problemas menores: o primeiro, foram as formulações de calibragem que atualmente não consideram o viés do usuário; o segundo, foram as formas de personalizar o grau de calibragem, o qual os trabalhos do estado da arte não exploram; o terceiro, foi a exploração das medidas de divergências que venham a obter melhores resultados quando comparadas com as utilizadas no estado da arte; e, por fim, a falta de métricas de avaliação para calibragem. Recomendações não calibradas tendem a recomendar o campo, gênero ou classe de maior domínio nas preferências, sobrepondo assim os gêneros com uma menor proporcionalidade nas preferências ou em alguns casos não incluindo-os na lista de recomendação. Isto pode implicar em que os usuários não se sintam representados ou respeitados pelo sistema, vendo seus desejos limitados a uma pequena parcela de classes, fazendo com que ao longo do tempo abandonem a aplicação ou não recomendem a adesão a amigos ou parentes.

A formulação da calibragem é dada como um balanceamento entre o ranqueamento por itens similares (mais relevantes) e a calibragem pelos gêneros que compõem as preferências. Esta formulação é seguida por diversos trabalhos na literatura (STECK, 2018;

KAYA; BRIDGE, 2019), entretanto, não leva em conta o viés das preferências do usuário, o que pode acarretar em um enviesamento pelos itens mais populares. Assim, outro problema que esta dissertação investigou, decorrente do não atendimento das expectativas do usuário baseando-se nas preferências individuais, foi o enviesamento por popularidade. Este que toma os itens mais populares entre os usuários como aqueles que devem ser recomendados, desrespeitando o usuário e as suas preferências.

Na literatura Steck (2018), Kaya e Bridge (2019) aprofundam o estudo sobre a calibragem com uma formulação que balanceia o grau do ranqueamento dos itens de acordo com a similaridade e o grau de calibragem para gerar a lista de recomendação. Entretanto ambos os estudos propõem apenas valores constantes do peso do balanceamento, não contemplando a personalização deste peso. Por exemplo, $0,7$ para o ranqueamento e $0,3$ para a calibragem, sendo que a soma dos pesos deve dar $1,0$. A não consideração de que cada usuário deseja um grau de calibragem diferente pode não contemplar o real desejo dos usuários. Se o peso constante é atribuído para dar maior importância ao ranqueamento, os usuários que desejam uma lista melhor calibrada não terão suas preferências atendidas. O mesmo acontece quando o peso constante é atribuído para dar maior importância à calibragem, os usuários que desejam uma lista mais ranqueada pela similaridade não terão suas preferências atendidas. Assim, a falta de personalização do peso gera um problema que foi estudado neste trabalho.

Steck (2018), Kaya e Bridge (2019) estudam a medida de divergência Kullback-Leibler para entender o quão calibrada a lista está e Abdollahpouri et al. (2020) estudam a medida Hellinger. Entretanto, cada medida gera uma lista diferente que contempla itens diferentes para comporem a lista de recomendação. A má escolha da medida de divergência pode gerar uma lista com uma precisão e calibragem com menor desempenho. Assim, este problema foi estudado nesta dissertação com adição de novas medidas de divergência que foram implementadas e comparadas para entender qual gera as melhores listas que satisfaçam aos usuários.

Ademais, a calibragem é um tema recente na literatura de Sistemas de Recomendação, contendo assim apenas uma métrica de avaliação especializada proposta por Steck (2018) que traduz os resultados obtidos. A variedade das métricas de avaliação ajuda a entender melhor o comportamento do sistema, assim a ausência de métricas que contemplem o entendimento do erro na calibragem e o nível de descalibragem ao longo das posições da lista gera um problema de real entendimento dos resultados obtidos pelo sistema. Assim, nesta dissertação também foi abordado novas métricas e coeficientes.

1.3 OBJETIVO

O objetivo desta dissertação foi desenvolver um protocolo de decisão para sistemas de recomendação calibrados. Protocolo este que visa indicar entre um conjunto de possíveis combinações de sistemas qual é o recomendado a ser implementado. Os sistemas gerados foram baseados em filtragem colaborativa para exemplificar o uso da técnica no protocolo. Cada sistema buscou produzir algum grau de justiça à distribuição dos gêneros que compõem as preferências de cada usuário, recomendando-os com listas de itens calibradas a estes gêneros.

Esta calibragem visa respeitar ao máximo o histórico das preferências dos usuários em suas devidas proporções. Por exemplo, se as preferências do usuário em uma aplicação do domínio de filmes são constituídas de 60% de Sci-fi e 40% de Aventura, então a aplicação deve respeitar a proporção e gerar uma lista com o mínimo de divergência.

Além do objetivo principal, este estudo também apresenta quatro objetivos específicos que foram alcançados. Estes objetivos foram:

OBE1: Formular um balanceamento entre a relevância e divergência que considere o viés do usuário;

OBE2: Propor maneira de encontrar o grau de calibragem de forma personalizada;

OBE3: Comparar e encontrar a melhor medida de divergência entre o estado da arte;

OBE4: Propor e comparar métricas e coeficientes de avaliação para calibragem.

1.4 QUESTÕES DE PESQUISA

Diante do problema e dos objetivos apresentados, as questões de pesquisa auxiliaram a condução do trabalho e as respostas são partes do problema resolvido. Essas questões conduziram e ajudaram a validar o que foi proposto como solução para o problema apresentado. Algumas das questões que esse trabalho respondeu foram:

QP1: Como encontrar o melhor sistema de recomendação calibrado?

QP2: A base de dados utilizadas influencia no comportamento ou desempenho do sistema calibrado?

QP3: É possível padronizar os sistemas calibrados de modo que auxiliem o desenvolvimento?

QP4: Quais são os efeitos que a calibragem produz nas listas de recomendação de cada algoritmo recomendador?

QP5: Utilizar pesos personalizados do balanceamento obtém melhorias ou mantém desempenho quando comparados com pesos constantes?

QP6: A medida de divergência utilizada influencia nas listas de recomendação?

QP7: Ao considerar o viés do usuário na formulação do balanceamento é possível obter melhoria no desempenho?

1.5 METODOLOGIA

Este trabalho investigou, propões e avaliou novas abordagens sobre o tema de calibragem em Sistemas de Recomendação que visam aprimorar os métodos do estado da arte. Assim, o desenvolvimento deste estudo deu-se em tópicos metodológicos apresentados a seguir:

- **Detecção de Oportunidade de Pesquisa**: A partir da revisão da literatura a técnica de filtragem colaborativa, que é a mais utilizada no estado da arte de Calibragem, foi escolhida. Em seguida, foram detectadas lacunas, as quais foram vistas como oportunidades de pesquisa e contribuição para a comunidade científica;

- **Implementação da Proposta**: A partir do estado da arte, os recomendadores foram implementados, seguindo a técnica escolhida, assim como os demais algoritmos usados pelos estudos. Após acompanhar o estado da arte foram implementados os algoritmos propostos e alguns ainda não abordados pela comunidade. Ao todo foram implementados: nove recomendadores, três medidas de divergência, três pesos do balanceamento, uma medida de relevância, duas formulações de balanceamento e um algoritmo de ranqueamento;

- **Execução de Experimentos**: Duas bases de dados foram escolhidas para a execução *offline* dos experimentos. Seis recomendadores foram executados em Grid Search para encontrar os melhores hiper parâmetros para cada base de dados. Após obter os parâmetros o sistema foi executado cuidadosamente, devido ao total de combinações de resultados que são 1404 possibilidades de sistema;

- **Avaliação**: Diversas métricas e coeficientes foram utilizados para analisar os resultados, estes que são: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR) e Mean Absolute Error (MAE); duas métricas propostas: Mean Average Calibration Error (MACE) e Mean Rank Miss Calibration (MRMC); e dois coeficientes Coefficient of Calibration Error (CCE) e Coefficient of MisCalibration (CMC). As avaliações observaram todas as posições das listas de recomendações geradas.

1.6 CONTRIBUIÇÕES OBTIDAS

A pesquisa que foi desenvolvida ao longo desta dissertação vem para contribuir em diversos aspectos com o estado da arte de Sistemas de Recomendação, principalmente com o aspecto de justiça e calibragem. A seguir são pautadas algumas contribuições obtidas:

- **Um protocolo decisório para sistemas de recomendação calibrado** é proposto no corpo desta dissertação, visando padronizar e descrever os componentes necessários para gerar um sistema de recomendação calibrado. Além de possibilitar a decisão de qual combinação de sistema calibrado se adéqua melhor aos dados utilizados;

- **Um algoritmo de calibragem** é apresentado durante esta dissertação, este que é de base genérica e pode ser adaptado a qualquer nova medida de divergência, medida de relevância, pesos de balanceamento, dentre outros algoritmos apresentados. Seguindo o definido pelo protocolo proposto;

- **Dois pesos do balanceamento relevância-divergência** são apresentados neste trabalho, destacando o ineditismo de pesos personalizados para uma calibragem mais respeitosa;

- Uma **formulação de balanceamento relevância-divergência** é apresentada durante esta dissertação, visando incrementar o desempenho do sistema a partir da influência do viés do usuário durante a criação da lista de recomendação;
- **Duas novas métricas de avaliação** para calcular o erro da calibragem são apresentadas neste trabalho, sendo esta pesquisa a primeira com uma proposta de nova classe de métricas, denominadas erro na calibragem;
- **Dois coeficientes decisórios** para auxiliar no entendimento do melhor resultado ao longo das combinações de sistemas.

1.7 ESTRUTURA

Neste capítulo introduziu-se o tema da dissertação, apresentou-se a motivação e o problema que baseiam este trabalho. Os próximos capítulos são organizados da seguinte forma: o Capítulo 2 apresenta uma revisão bibliográfica sobre Sistemas de Recomendação, demonstrando os conceitos, modelagens, técnicas e seus algoritmos, além das formas de avaliação e exemplos industriais; o Capítulo 3 descreve os trabalhos relacionados com a nossa pesquisa, debatendo a respeito de justiça e calibragem em Sistemas de Recomendação, além de apresentar os conceitos; o Capítulo 4 apresenta a proposta da pesquisa bem como as modelagens usadas, os algoritmos propostos, as etapas de funcionamento do sistema e toda a formalização da implementação; o Capítulo 5 detalha as bases de dados, a metodologia, as métricas utilizadas, além de apresentar os resultados obtidos; o Capítulo 6 apresenta a conclusão, contribuições obtidas e trabalhos futuros.

Capítulo 2

SISTEMAS DE RECOMENDAÇÃO

Neste capítulo é abordada uma revisão bibliográfica sobre Sistemas de Recomendação. Inicialmente é apresentada uma introdução, tarefas e conceitos. Em seguida são abordados os modelos de dados dos usuários. As técnicas de recomendação são descritas e exemplificadas, assim como seus processos, vantagens e desvantagens. Ademais nove algoritmos recomendadores utilizados neste trabalhado também são apresentados. Por fim, são apresentados, ainda, os protocolos de avaliação e as métricas de predição e ranque.

2.1 INTRODUÇÃO

Ricci, Rokach e Shapira (2011) afirmam que, "em sua forma mais simples, recomendações personalizadas são fornecidas como uma lista ordenada de itens candidatos". Para encontrar os itens que constituirão essa lista, Sistemas de Recomendação tentam predizer qual produto ou serviço mais agradará aos usuários, baseando-se no histórico das preferências.

Resnick e Varian (1997) em seu artigo nomeado "Sistemas de Recomendação" mostram que estes sistemas apoiam e aumentam o processo de recomendação. Basicamente, o que têm-se aqui são técnicas e ferramentas de programas para fornecer sugestões de itens que sejam úteis para um usuário. Nesses sistemas, "item" é um termo genérico utilizado para denotar o que o sistema sugere a um usuário, e o "usuário" é um termo genérico utilizado para denotar a quem o sistema irá oferecer os itens recomendados (RICCI; ROKACH; SHAPIRA, 2011).

2.2 TAREFAS E CONCEITOS

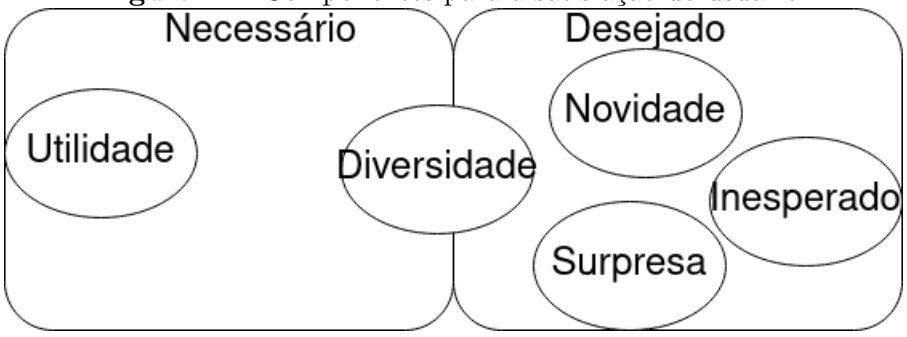
Os Sistemas de Recomendação possuem objetivos que as aplicações implementam em comum. Ricci, Rokach e Shapira (2011) sintetizam alguns desses objetivos, que são: aumentar o número de itens vendidos, vender mais itens diferentes, aumentar a satisfação dos usuários, aumentar a fidelidade dos membros e entender melhor o desejo dos nossos usuários. Estes objetivos são adaptados ou reformulados de acordo com o domínio da aplicação. Por exemplo, uma loja eletrônica deseja vender mais itens assim como vender

itens diferentes, entretanto um serviço de transmissão de filme não vende o item, mas deseja que seus usuários tenham contato com o maior número possível de itens disponíveis.

Herlocker et al. (2004) apresentam onze tarefas que os Sistemas de Recomendação devem implementar: encontrar alguns bons itens, encontrar todos os bons itens, entender o contexto, recomendação em sequência, recomendação em grupo, navegação entre os itens, recomendação confiável, melhorar o perfil do usuário, permitir a expressividade do usuário, ajudar os usuários e, por fim, influenciar os outros usuários.

Os objetivos citados por Ricci, Rokach e Shapira (2011) e as tarefas descritas por Herlocker et al. (2004) não são os únicos pontos que os Sistemas de Recomendação devem abordar. McNee, Riedl e Konstan (2006) alertam que apenas alcançar o objetivo de recomendar não é o bastante e que os Sistemas de Recomendação devem visar outros aspectos. Assim, as pesquisas posteriores trouxeram aspectos como: diversidade de itens na recomendação, surpresa ao usuário com um item não esperado, justiça com o histórico de preferências, itens que tragam novidade, utilidade do item para o usuário, cobertura, dentre outros. A Figura 2.1 demonstra o que é necessário e/ou desejado para a satisfação do usuário, de acordo com Silveira et al. (2017), que realizam um apanhado de como medir o quão bom o sistema de recomendação é.

Figura 2.1 Componentes para a satisfação do usuário.



Fonte: Tradução nossa de Silveira et al. (2017).

De acordo com Ricci, Rokach e Shapira (2011), os sistemas de recomendação utilizam basicamente três tipos de dados para suprir seu objetivo: itens, usuários e relações dos usuários com os itens, chamadas de transações.

- **Itens**: São os objetos a serem recomendados. Podem ser caracterizados por sua complexidade, por seu valor e/ou utilidade. O valor do item pode ser positivo caso ele seja útil para o usuário, ou negativo caso seja inapropriado. Itens podem ser representados utilizando várias abordagens de representação e informação. Em aplicações como o Netflix[1], o item é o filme que será recomendado; no Spotify[2] o item é uma música; no Coursera[3] o item é um curso; na Amazon[4] o item é um

[1]https://www.netflix.com
[2]https://www.spotify.com
[3]https://www.coursera.org/
[4]https://www.amazon.com.br/

produto. Ao longo deste trabalho, formalmente o conjunto de itens do sistema é representado pela letra I, onde um item qualquer é representado pela letra i ou j;

- **Usuários**: Possuem diversos objetivos e características. Para realizar a personalização das recomendações, os sistemas de recomendação utilizam um conjunto de dados sobre o usuário. Esses dados podem ser estruturados de várias maneiras e a seleção de quais dados utilizar depende da técnica de recomendação e da necessidade do domínio da aplicação. Ao longo deste trabalho, formalmente o conjunto dos usuários ativos do sistema é representado pela letra U, onde um usuário qualquer é representado pela letra u ou v;

- **Transações**: São registros das interações entre os usuários e os itens do sistema. São dados estruturados que armazenam informações importantes geradas durante a interação do usuário com o sistema, os quais após devidamente selecionados, tratados e modelados são usados pelos algoritmos recomendadores para encontrar recomendações úteis. Ao longo deste trabalho, formalmente o conjunto de transações do sistema é representado pela letra R, onde as transações de um usuário são representadas como $R(u)$ e as transações de um item são representadas como $R(i)$.

2.3 MODELO DO USUÁRIO

Durante a interação do usuário com o sistema, transações são registradas e servem como base de dados para a modelagem. De acordo com Ricci, Rokach e Shapira (2011) é possível adotar duas técnicas para coletar os dados dos usuários sobre os itens com os quais eles interagem, em grande parte das interações os usuários podem interagir positivamente com o item ou interagir negativamente com o item.

2.3.1 Feedback Explícito

Quando o sistema requer do usuário um retorno direto na avaliação a respeito de um item, questionando-o sobre a relevância deste, diz-se que este tipo de feedback é explícito. Este tipo de retorno do usuário auxilia o sistema a entender o quão interessante ou relevante o item foi para o usuário.

Celma (2009) e Ricci, Rokach e Shapira (2011) apresentam formas de obter o feedback explícito do usuário. O uso do feedback binário é uma das formas de entender a relevância do item para o usuário como positivo/negativo (*like/dislike*). O uso de feedback discreto com valores a partir de 0 abre uma maior possibilidade de avaliação sobre o interesse do usuário. Uma possibilidade de valores é a escala Likert (LUCIAN, 2016) de 0 à 5, como demonstrado na Figura 2.2(a) retirada do serviço de *streaming* Crunchyroll[5]. Uma outra forma de obter um retorno direto do usuário é requisitar a escrita de comentários ou opiniões sobre o item.

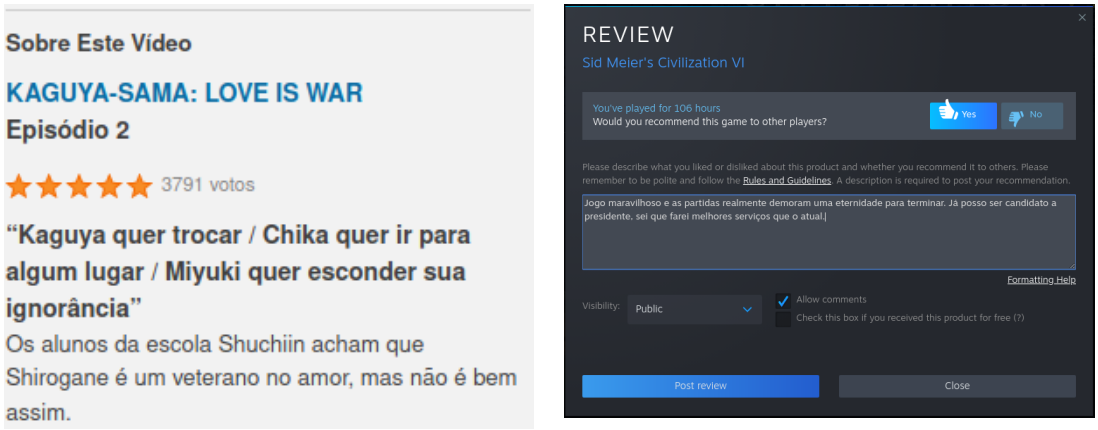
A aplicação da técnica de feedback explícito pode requerer do usuário mais de um tipo de interação, por exemplo, a Figura 2.2(b) demonstra que a loja de jogos eletrônicos

[5]https://www.crunchyroll.com/

Steam[6] requisita do usuário tanto o retorno binário quanto o retorno por comentário.

Figura 2.2 Feedback explícito escala likert, binário e por comentário.

(a) Escala Likert (b) Binário e Comentário



Fonte: Figuras capturadas pelo autor, Crunchyroll e Steam, 2020.

2.3.2 Feedback Implícito

De acordo com Celma (2009) "um sistema de recomendação pode inferir as preferências dos usuários a partir de monitoramento passivo das ações". Ricci, Rokach e Shapira (2011) afirmam que "o feedback implícito não requer qualquer ação que envolva o usuário, levando em conta que o feedback é derivado do monitoramento e análise das atividades dos usuários". Assim, o sistema deve a partir das transações do usuário selecionar e modelar os dados da interação a partir do monitoramento comportamental.

Figura 2.3 Feedback implícito a partir de uma busca textual.

Fonte: Figura capturada pelo autor, Google AdWords, 2020.

A Figura 2.3 demonstra a recomendação de roteadores para o usuário após uma realização de buscas por roteadores. Assim, o sistema de recomendação entendeu que durante as próximas pesquisas o usuário possui a intenção de buscar por mais roteadores.

O método do feedback implícito é baseado em atribuir um valor de relevância para uma ação do usuário sobre um item, como: salvar, descartar, imprimir, marcar, movimento do

[6]https://store.steampowered.com/

mouse, tempo em uma página, dentre outros. Assim, a quantidade de dados que pode ser coletada durante a interação do usuário com o sistema é grande. Entretanto, equívocos na interpretação dos dados podem vir a acontecer, por exemplo, ao usuário clicar sem querer em uma URL ou esquecer o reprodutor de música tocando um álbum que não lhe agrade. Nestes casos, o sistema pode interpretar que o usuário possui interesse sobre estes itens, cometendo assim um equívoco que será reproduzido nas recomendações.

2.3.3 Representação dos dados

Com os dados coletados e modelados, sendo o feedback explícito ou implícito, o sistema está apto a iniciar a fase de estudos dos dados para encontrar as recomendações. Na Tabela 2.1 é apresentada a representação formal dos dados estruturados. Na Seção 4.3 são apresentadas diversas modelagens dos dados que seguem a especificidade do domínio deste trabalho.

Tabela 2.1 Representação formal dos conjuntos de dados utilizados pelo sistema.

Símbolo	Descrição
U	Conjunto de todos os usuários
I	Conjunto de todos os itens
$\|U\|$	Número referente ao tamanho do conjunto U
R	Conjunto de todos os feedbacks válidos
$R(u)$	Todos os feedbacks do usuário u
$\hat{R}$	Conjunto de todos os feedbacks preditos
r_{ui}	Valor do feedback do usuário u ao item i
r_u	Valores de todos os feedbacks do usuário u
u, v	Um usuário qualquer
i, j	Um item qualquer
$\|I\|$	Número referente ao tamanho do conjunto I
$\|R\|$	Número referente ao tamanho do conjunto R
$R(i)$	Todos os feedbacks do item i
$\|\hat{R}\|$	Número referente ao tamanho do conjunto $\hat{R}$
$\hat{r}_{ui}$	Valor predito pelo recomendador do usuário u ao item i
r_i	Valores de todos os feedbacks do item i

Fonte: Tabela elaborada pelo autor.

2.4 TÉCNICAS DE RECOMENDAÇÃO

Como explanado por Ricci, Rokach e Shapira (2011) os Sistemas de Recomendação utilizam de técnicas para fornecer sugestões de itens que sejam úteis para um usuário. As principais técnicas de recomendação, as quais são descritas durante as próximas seções, são: Filtragem Baseada em Conteúdo, Filtragem Colaborativa e Filtragem Híbrida. Entretanto, estas não são as únicas técnicas existentes. Burke (2002) faz um levantamento de diversas técnicas de recomendação existentes além das três principais, como: baseada em

demografia, baseada em utilidade e baseada em conhecimento. A Tabela 2.2 apresentada no trabalho de Burke (2002) é um resumo sobre algumas técnicas de recomendação.

Tabela 2.2 Técnicas de recomendação, suas modelagens, entradas e processamento das principais técnicas de recomendação.

Técnicas	Modelagem	Entrada	Processo
Colaborativa	Feedbacks de todos os usuários	feedbacks do usuário em processamento sobre os itens de sua preferência	Identifica os usuários similares ao usuário em processamento e utiliza seus feedbacks para recomendar os itens.
Conteúdo	Dados dos itens	feedbacks do usuário em processamento sobre os itens de sua preferência	Molda um classificador adequado ao seu comportamento de classificação e o usa para selecionar os itens.
Demografia	Informação demográfica dos usuários e seus feedbacks	Informações demográficas do usuário em processamento	Identifica os usuários que são demograficamente similares ao usuário em processamento e utiliza seus feedbacks para recomendar os itens.
Utilidade	Dados dos itens	Uma função de utilidade sobre os itens que descreva as preferências do usuário em processamento	Aplica a função nos itens e determina quais itens farão parte do ranque.
Conhecimento	Dados dos itens. Conhecimento de como esses itens atendem às necessidades de um usuário	Descrição das necessidades ou dos interesses do usuário em processamento	Infere uma correspondência entre o item e o usuário em processamento.

Fonte: Tradução nossa de Burke (2002).

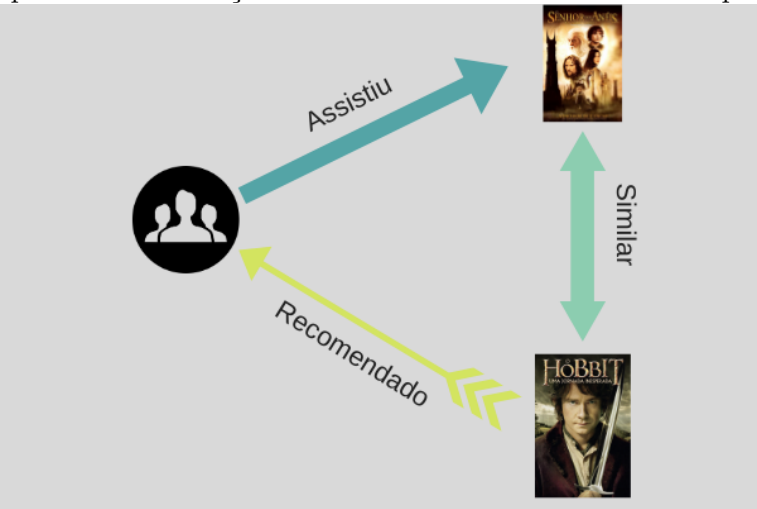
2.5 FILTRAGEM BASEADA EM CONTEÚDO

Uma das técnicas amplamente usadas é a Filtragem Baseada em Conteúdo. Esta técnica baseia-se nos dados que compõem os itens para encontrar itens similares com as preferências dos usuários. Adomavicius e Tuzhilin (2005) afirmam que o método baseado em conteúdo tem sua origem nas pesquisas de recuperação da informação. De acordo com Ricci, Rokach e Shapira (2011) "sistemas de recomendação que usam filtragem baseada em conteúdo tentam recomendar itens similares com aqueles que os usuários preferiram no passado". Isinkaye, Folajimi e Ojokoh (2015) definem que "a técnica baseada em conteúdo é um algoritmo dependente de domínio e enfatiza mais a análise dos atributos dos itens para gerar previsões".

Os sistemas que implementam a técnica de filtragem baseada em conteúdo analisam os metadados que constituem os itens, formando uma estrutura de informação que é utilizada durante o processo de recomendação. Este processo baseia-se no uso dos metadados dos itens das preferências de um usuário para então encontrar itens desconhecidos que possuem conteúdos similares e, por fim, ranqueia estes itens em uma lista de recomendação, levando em conta o quão similar este item desconhecido pelo usuário é com os itens nas preferências.

Como exemplo, na Figura 2.4 é possível verificar que o usuário assistiu ao filme "Senhor dos Anéis" e devido à similaridade das informações que descrevem os filmes é recomendado o "O Hobbit".

Figura 2.4 Exemplo de recomendação de filmes baseando-se no conteúdo que descreve os itens.



Fonte: Figura elaborada pelo autor.

2.5.1 Vantagens e Problemas

Os Sistemas de Recomendação que utilizam filtragem baseada em conteúdo apresentam vantagens e problemas como todas as outras técnicas. Os prós e contras do uso da técnica são apresentados nos estudos de Adomavicius e Tuzhilin (2005), Lops, Gemmis e Semeraro (2011) e Isinkaye, Folajimi e Ojokoh (2015).

As vantagens do uso desta técnica incluem:

- **Rápida adaptação**: Adaptação das recomendações caso as preferências do usuário passem por alterações, sendo essas alterações grandes ou pequenas;
- **Perfil pequeno**: Recomendação de novos itens mesmo que o usuário não possua muitos itens em seu perfil ou não contribua com uma quantidade significativa de feedbacks;
- **Explanabilidade**: Os passos executados pelos algoritmos para encontrar a lista de itens são fáceis de serem auditados e entendidos.

A técnica também possui desvantagens, estas que incluem:

- **Super especialização**: Os itens recomendados possuem o máximo de similaridade com os itens no perfil do usuário, portanto itens externos às preferências normalmente não entram na lista de recomendação, levando o usuário a viver em uma bolha de preferências;
- **Dependência de informação**: Os itens necessitam de metadados que os descrevem, quanto menos informações mais imprecisa a recomendação é, assim a descrição dos itens precisa ser bem estruturada e rica em dados;

- **Novo usuário**: Mesmo a técnica recomendando itens para perfis com poucos feedbacks quando um usuário não possui nenhum item em seu perfil, o sistema não consegue encontrar recomendações personalizadas. Este problema é conhecido como partida a frio.

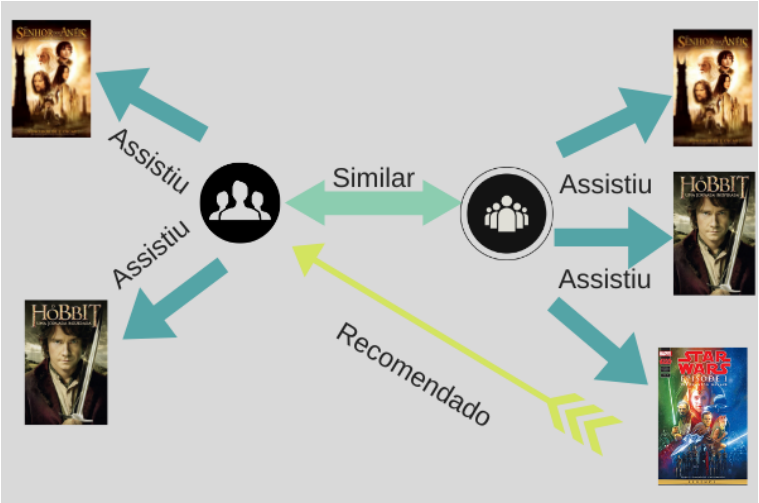
2.6 FILTRAGEM COLABORATIVA

Uma das técnicas mais utilizadas tanto em pesquisas quanto na indústria é a filtragem colaborativa. Ela toma como base os feedbacks de todos os usuários para recomendar novos itens para um outro usuário, levando em conta o histórico de preferências deste. A técnica leva em conta somente os feedbacks não dependendo assim do conteúdo dos itens ou informações extras dos usuários. Ricci, Rokach e Shapira (2011) afirmam que "esta abordagem recomenda para um usuário ativo os itens que outros usuários com gostos similares gostaram no passado". Burke (2002) indica que "recomendadores colaborativos agregam classificações ou objetos de recomendação, reconhecendo pontos em comum entre os usuários tomando como base as suas classificações, e gerando novas recomendações baseadas na comparação inter usuários".

"É provável que a classificação de u para um novo item i seja semelhante à de outro usuário v, se u e v classificaram outros itens de maneira semelhante. Da mesma forma, é provável que u classifique dois itens i e j de maneira semelhante, se outros usuários tiverem classificações semelhantes a esses dois itens" (DESROSIERS; KARYPIS, 2011).

Por exemplo, a Figura 2.5 demonstra uma possibilidade de recomendação utilizando a técnica de filtragem colaborativa. É possível verificar que os usuários à esquerda assistiram aos filmes de "Senhor dos Anéis" e "O Hobbit" e os usuários da direita assistiram aos filmes de "Senhor dos Anéis", "O Hobbit" e "Star Wars". Assim, os usuários da esquerda são recomendados com o filmes de "Star Wars".

Figura 2.5 Exemplo de recomendação de filmes baseando-se nas preferências dos usuários.



Fonte: Elaborada pelo autor.

Os algoritmos da técnica são divididos em dois grupos: método baseado em memória

e o método baseado em modelo (RICCI; ROKACH; SHAPIRA, 2011; ISINKAYE; FOLAJIMI; OJOKOH, 2015). Um terceiro método pode ser caracterizado, este que se atém a formas tradicionais de filtragem e ordenação dos dados, nas quais os algoritmos não são baseados em modelo ou memória. A seguir são descritas as características de cada método:

- **Método baseado em memória**: Em alguns estudos é chamada de método baseado em vizinhança ou método baseado em heurística. Usa diretamente as classificações dos usuários para encontrar outros vizinhos que possuam classificações semelhantes. Este método pode ser implementado de duas formas: i) baseando-se no usuário e ii) baseando-se no item. i) Ao implementar o caminho baseado no usuário o sistema calcula a similaridade das classificações entre um usuário u e os demais usuários, buscando usuários que possuam o maior grau de similaridade. Ao encontrar os vizinhos (usuários) mais similares, os itens das preferências destes são usados como base das recomendação a u. ii) Os algoritmos baseados no item buscam classificações similares entre os itens, a partir de um determinado item i nas preferências de u são encontrados itens que possuam similaridade nas classificações;

- **Método baseado em modelo**: Utiliza algoritmos de aprendizado de máquina para aprender sobre o perfil do usuário. O aprendizado do perfil é pré-computado e mantido no sistema, o que facilita o processo de recomendação, entretanto é necessário um período de aprendizado. O método baseado em modelo analisa a relação usuário-item para encontrar novos itens. Este método consegue trabalhar com esparsidade melhor que o método em memória. Os aprendizados utilizados podem ser desde os probabilísticos, fatorizadores de matrizes, redes neurais, dentre outros;

- **Método tradicional**: Utiliza formas simples de filtragem, seleção e ordenação dos itens. Alguns algoritmos não trabalham com personalização, outros apenas recomendam itens com os quais o usuário ainda não teve contato, tomando como base valores globais.

A Tabela 2.3 apresenta um resumo sobre os métodos, os grupos e os algoritmos utilizados nesta dissertação. Nas próximas seções apresenta-se os algoritmos da tabela, assim como as representações formais de cada um deles.

2.6.1 Vantagens e Problemas

Os sistemas de recomendação que implementam a técnica de filtragem colaborativa aceitam as vantagens e as desvantagens existentes na abordagem. Algumas vantagens são associadas aos métodos baseados em vizinhança e algumas são associadas aos baseados em modelos, assim como para as desvantagens.

As vantagens do uso da técnica colaborativa são:

1. **Simplicidade**: Os recomendadores baseados em memória são fáceis de entender, simples de implementar e adaptar (ISINKAYE; FOLAJIMI; OJOKOH, 2015; DESROSIERS; KARYPIS, 2011; SU; KHOSHGOFTAAR, 2009);

Tabela 2.3 Métodos, grupos e algoritmos da técnica de filtragem colaborativa.

Métodos	Grupos	Algoritmos
Tradicionais	Seleção	Popularidade Melhor nota
Baseado em memória	Usuário	User-KNN
	Item	Item-KNN
Baseado em modelo	Agrupamento	Co Clustering
	Fatoração de Matriz	SVD SVD++ NMF
	Slope One	Slope One

Fonte: Elaborada pelo autor.

2. **Justificabilidade**: As predições dos recomendadores baseados em memória possuem uma fácil e simples justificabilidade do porquê cada item está sendo recomendado (DESROSIERS; KARYPIS, 2011);

3. **Performance**: Os recomendadores baseados em modelos possuem uma alta precisão nas recomendações (SU; KHOSHGOFTAAR, 2009);

4. **Livre de conteúdo**: Diferentemente da técnica de filtragem baseada em conteúdo não se tornam necessários dados sobre os itens para encontrar recomendações, assim, em domínios onde os dados são raros esta técnica obtém melhores resultados (ISINKAYE; FOLAJIMI; OJOKOH, 2015; SU; KHOSHGOFTAAR, 2009).

As desvantagens do uso da técnica são:

1. **Enviesamento**: O uso de feedbacks pode influenciar a técnica a encontrar recomendações que não condizem com as preferências dos usuários. Um dos enviesamentos mais comuns é o por popularidade, onde as recomendações podem ser compostas pelo mesmo grupo de itens populares entre os usuários mais antigos do sistema. Outro enviesamento é a sobreposição, onde um gênero de maior preferência sobrepõe um gênero de menor preferência;

2. **Partida a frio**: Um dos principais problemas da técnica se refere a quando um usuário acaba de aderir à aplicação e não contribuiu com nenhum tipo de feedback. Nesse caso, a técnica não consegue encontrar recomendações para o usuário (ISINKAYE; FOLAJIMI; OJOKOH, 2015; SU; KHOSHGOFTAAR, 2009);

3. **Novo item**: Um novo item ao ser cadastrado na aplicação não possui nenhum feedback do usuário, assim a técnica não consegue encontrar o item e adicioná-lo na lista de recomendação de algum usuário que possa vir a preferir o item (ISINKAYE; FOLAJIMI; OJOKOH, 2015; SU; KHOSHGOFTAAR, 2009);

4. **Ovelha cinza**: Usuários que não se encaixam em nenhum grupo ou possuem preferências que não permitam encontrar vizinhos não recebem recomendações, devido à técnica não conseguir encontrar novos itens similares às preferências a partir de feedbacks de outrem (QIN, 2013; SU; KHOSHGOFTAAR, 2009);

5. **Ataque de xelim**: Um item concorrente nas recomendações pode deliberadamente alimentar o sistema com feedbacks negativos para punir um adversário, ganhando vantagens para si e não permitindo que usuários possam receber o item adversário como recomendação (QIN, 2013; SU; KHOSHGOFTAAR, 2009).

2.6.2 Tradicionais

A seguir são apresentados dois algoritmos tradicionais que não são baseados em memória ou modelo. Estes que são utilizados na filtragem, seleção e ordenação de itens e são implementáveis em poucas linhas de código.

2.6.2.1 Popularidade É um algoritmo bastante usado na ordenação de itens e é comumente encontrado nos sistemas. Este algoritmo baseia-se em contar em quantos perfis determinado item está presente, ou seja, quantos usuários preferiram este item. A Equação 2.1 demonstra formalmente a recomendação, para todos os itens $i \in I$ é contado quantas vezes esse item aparece nos feedbacks ($|R(i)|$).

$$pop(u) = \arg\max_{i \in I} |R(i)| \tag{2.1}$$

2.6.2.2 Melhor nota Resume os feedbacks dos usuários sobre um determinado item. Normalmente, é realizada uma média dos valores dos feedbacks. Assim, são recomendados os itens com os melhores feedbacks que o usuário ainda não conheça. A Equação 2.2 demonstra que para todo $i \in I$ uma média dos feedbacks μ_i é realizada, recomendando assim os itens com a melhor média dos valores dos feedbacks.

$$bs(u) = \arg\max_{i \in I} \mu_i \tag{2.2}$$

2.6.3 KNN

Os métodos de recomendação baseados em memória, também conhecidos como baseados em vizinhança, utilizam os feedbacks dos usuários, buscando encontrar o grau de similaridade entre os vizinhos. Quanto maior o nível de similaridade mais próximos esses vizinhos estão. Este método é dividido em dois grupos: o que verifica a similaridade entre os usuários e o grupo que verifica similaridade entre os itens. Um dos recomendadores baseados em vizinhança mais utilizados é o K Nearest Neighbors (KNN). Desrosiers e Karypis (2011) e Koren e Bell (2015) apresentam formulações do KNN para os usuários e para os itens chamados de User-KNN e Item-KNN.

2.6.3.1 User-KNN Esta implementação do KNN baseia-se em encontrar usuários com os maiores graus de similaridades. A relação de todos os usuários $u \in U$ com todos os itens $i \in I$ é modelada como uma matriz usuário-item, em que cada linha é um usuário e cada coluna é um item. Ao selecionar um usuário u para encontrar as recomendações é verificado quais outros usuários possuem similaridades nos feedbacks. O User-KNN mede o grau de similaridade entre um usuário u e outro usuário v através de qualquer medida de distância. A seguir são apresentadas algumas das medidas de distância que baseiam-se no cálculo de distância entre vetores.

$$sim(u,v) = PC(u,v) = \frac{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)(r_{vi} - \mu_v)}{\sqrt{\sum_{i \in I_{uv}} (r_{ui} - \mu_u)^2 \sum_{i \in I_{uv}} (r_{vi} - \mu_v)^2}} \tag{2.3}$$

A Equação 2.3 apresenta a correlação de Pearson, na qual dois usuários u e v são dados como entrada. Na equação, I_{uv} representa as posições dos vetores, que por sua vez são itens. r_{ui} representa o feedback do usuário u sobre o item i e r_{vi} do usuário v sobre o item i. μ_u representa a média dos valores dos feedbacks do usuário u, μ_v a média do usuário v.

$$sim(u,v) = cos(u,v) = \frac{\sum_{i \in I_{uv}} r_{ui} r_{vi}}{\sqrt{\sum_{i \in I_{uv}} r_{ui}^2 \sum_{i \in I_{uv}} r_{vi}^2}} \tag{2.4}$$

A Equação 2.4 é a representação formal da similaridade do Cosseno, em que dois usuários u e v são dados como entrada. Na equação, I_{uv} representa as posições dos vetores de u e v, que por sua vez são itens i. r_{ui} representa o valor do feedback do usuário u sobre o item i, r_{vi} do usuário v sobre o item i.

$$msd(u,v) = \frac{1}{I_{uv}} \cdot \sum_{i \in I_{uv}} (r_{ui} - r_{vi})^2 \tag{2.5}$$

$$sim(u,v) = msdsim(u,v) = \frac{1}{msd(u,v) + 1} \tag{2.6}$$

A Equação 2.5 computa a média da diferença quadrada (em inglês, *Mean Squared Difference - MSD*). O valor é dado como a soma da diferença entre os valores do feedback dos dois usuários ao quadrado, dividido pelo número de itens em I_{uv}. Na Equação 2.6 a representação formal da similaridade da média da diferença quadrada é apresentada, onde dois usuários u e v são dados como entrada. Quanto menor o valor provindo da Equação 2.5 maior a similaridade entre os usuários.

$$\hat{r}_{ui} = \frac{\sum_{v \in N_i^k} sim(u,v) \cdot r_{vi}}{\sum_{v \in N_i^k} sim(u,v)} \tag{2.7}$$

A Equação 2.7 representa a predição do User-KNN de usuário sobre um item $\hat{r}_{ui}$. O N_i^k representa os k vizinhos com o maior grau de similaridade com o usuário u. Com isso, o recomendador possui dois hiper parâmetros, o k e a medida de distância.

2.6.3.2 Item-KNN Esta implementação do KNN baseia-se em encontrar itens com os maiores graus de similaridades. A relação de todos os itens $i \in I$ por todos os itens $j \in I$ é modelada como uma matriz item-item, onde cada linha é um item e cada coluna também é um item. Ao selecionar um item i para encontrar as recomendações é verificado quais outros itens possuem similaridades nos feedbacks. O Item-KNN mede o grau de similaridade entre um item i e outro item j através de qualquer medida de distância. A seguir são apresentadas algumas das medidas de distância que baseiam-se no cálculo de distância entre vetores.

$$sim(i,j) = PC(i,j) = \frac{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)(r_{uj} - \mu_j)}{\sqrt{\sum_{u \in U_{ij}} (r_{ui} - \mu_i)^2 \sum_{u \in U_{ij}} (r_{uj} - \mu_j)^2}} \tag{2.8}$$

A Equação 2.8 apresenta a correlação de Pearson, na qual dois itens i e j são dados como entrada. Na equação, U_{ij} representa as posições dos vetores, que por sua vez são usuários. r_{ui} representa o feedback do usuário u sobre o item i, r_{uj} do usuário u sobre o item j. μ_i representa a média dos valores dos feedbacks do item i, μ_j a média do item j.

$$sim(i,j) = cos(i,j) = \frac{\sum_{u \in U_{ij}} r_{ui} r_{uj}}{\sqrt{\sum_{u \in U_{ij}} r_{ui}^2 \sum_{u \in U_{ij}} r_{uj}^2}} \tag{2.9}$$

A Equação 2.9 é a representação formal da similaridade do Cosseno, onde dois itens i e j são dados como entrada. Na equação, U_{ij} representa os usuários que possuem feedbacks dos itens i e j. r_{ui} representa o valor do feedback do usuário u sobre o item i, r_{uj} do usuário u sobre o item j.

$$msd(i,j) = \frac{1}{U_{ij}} \cdot \sum_{u \in U_{ij}} (r_{ui} - r_{uj})^2 \tag{2.10}$$

$$sim(i,j) = msdsim(i,j) = \frac{1}{msd(i,j) + 1} \tag{2.11}$$

A Equação 2.11 é a representação formal da similaridade média da diferença quadrada, entretanto nesta formulação utilizam-se os itens no lugar dos usuários, como previamente demonstrado na Equação 2.5. Assim, a similaridade é computada entre dois itens utilizando a média da diferença quadrada entre os feedbacks dos usuários sobre i e j.

$$\hat{r}_{ui} = \frac{\sum_{j \in N_u^k} sim(i,j) \cdot r_{uj}}{\sum_{j \in N_u^k} sim(i,j)} \tag{2.12}$$

A Equação 2.12 representa formalmente a predição do Item-KNN de usuário sobre um item $\hat{r}_{ui}$. O N_u^k representa os k vizinhos com o maior grau de similaridade com o item i. Com isso, o recomendador possui dois hiper parâmetros, o k e a medida de distância.

2.6.4 Fatoração de Matrizes

Dentro dos métodos baseados em modelos existe o grupo de algoritmos que trabalham com fatoração de matrizes. A Tabela 2.3 indica três possibilidades, as quais são apresentadas nesta seção, o Singular Value Decomposition (SVD), o Singular Value Decomposition Plus Plus (SVD++) e o Non-negative Matrix Factorization (NMF). Koren, Bell e Volinsky (2009) afirmam que os modelos baseados em fatoração de matrizes demonstram uma superioridade sobre os algoritmos baseados em vizinhança e afirmam também que estes algoritmos produzem uma recomendação com melhor precisão. Os recomendadores que trabalham com fatoração juntam em um fator latente de dimensionalidade f os interesses dos usuários sobre os itens.

2.6.4.1 SVD Famoso pelo seu desempenho durante o Netflix Prize[7]. O recomendador logo ganhou notoriedade e se tornou um dos algoritmos de fatoração de matrizes mais utilizados. A implementação descrita por Koren e Bell (2015) consideram ou não o viés do usuário e do item no preditor. As Equações 2.13, 2.14, 2.15, 2.16, 2.17 e 2.18 descrevem formalmente o SVD.

As Equações 2.13 e 2.14 representam formalmente o viés de um item b_i e um usuário b_u. Tem-se que μ é a média de todos os feedbacks dos usuários e r_{ui} é o valor do feedback do usuário u sobre um item i. Os valores de α e σ podem ser atribuídos pelo especialista ou encontrados a partir da distribuição normal, sendo que estes valores evitam a divisão por zero e são utilizados apenas no início do recomendador. R representa todos os feedbacks, sendo $R(u)$ todos os feedbacks de u e $R(i)$ todos os feedbacks do item i.

$$b_i = \frac{\sum_{u \in R(i)} (r_{ui} - \mu)}{\alpha + |R(i)|} \tag{2.13}$$

$$b_u = \frac{\sum_{i \in R(u)} (r_{ui} - \mu - b_i)}{\sigma + |R(u)|} \tag{2.14}$$

As Equações 2.15 e 2.16 demonstram formalmente como o SVD prediz o valor de um usuário sobre um item. q_i é um vetor de fatores latentes do item i de dimensionalidade f e p_u é um vetor com a mesma dimensionalidade, entretanto o vetor de p_u representa os fatores latentes do usuário u. A diferença entre as equações se dá devido à implementação escolhida considerar ou não o viés do usuário e do item, sendo a Equação 2.15 a que não utiliza o viés e a Equação 2.16 a que utiliza. Esta escolha fica a cargo do especialista do sistema e do domínio da aplicação.

$$\hat{r}_{ui} = q_i^T p_u \tag{2.15}$$

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T p_u \tag{2.16}$$

Para entender as variáveis do modelo (b_u, b_i, p_u e q_i), minimiza-se o erro quadrado regularizado, como é demonstrado na Equação 2.17 a seguir (KOREN; BELL, 2015).

[7] https://www.netflixprize.com/

$$min_{b*,q*,p*} \sum_{(u,i)\in K} (r_{ui} - \hat{r}_{ui})^2 + \lambda(b_i^2 + b_u^2 + ||q_i||^2 + ||p_u||^2) \tag{2.17}$$

A Equação 2.18 demonstra o cálculo da diferença entre o feedback real e o predito.

$$e_{ui} = r_{ui} - \hat{r}_{ui} \tag{2.18}$$

Abaixo é formalmente apresentada a regulação do recomendador, onde este itera em forma de "épocas" e a cada nova época adapta seus valores para obter uma menor taxa de erro. O valor γ representa a taxa de aprendizado que influencia o grau de adaptação.

- $b_u = b_u + \gamma \cdot (e_{ui} - \lambda \cdot b_u)$
- $b_i = b_i + \gamma \cdot (e_{ui} - \lambda \cdot b_i)$
- $q_i = q_i + \gamma \cdot (e_{ui} \cdot p_u - \lambda \cdot q_i)$
- $p_u = p_u + \gamma \cdot (e_{ui} \cdot q_i - \lambda \cdot p_u)$

O recomendador SVD possui ao todo seis hiper parâmetros. i) α e ii) σ, são valores obtidos através do especialista do sistema ou da distribuição normal e são utilizados apenas no início do processo. iii) γ, representa a taxa de aprendizado, onde a cada iteração o recomendador adiciona ou subtrai um valor regulado pelo aprendizado. iv) λ, representa o valor de regularização dos parâmetros b_u, b_i, p_u e q_i. v) f, representa a quantidade de fatores latentes. vi) "Épocas", representa quantas vezes o recomendador itera sobre os dados.

2.6.4.2 SVD++ Proposto para trabalhar também com feedback implícito e prover informações adicionais sobre as preferências dos usuários, o SVD++ é altamente útil em sistemas aos quais os usuários proveem mais feedbacks implícitos. Koren e Bell (2015) apresentam formalmente o recomendador que é uma melhoria do SVD.

A Equação 2.19 demonstra a primeira modificação que o SVD++ propõe sobre o SVD. O valor predito de um usuário u sobre um item i, leva em consideração um novo vetor de fatores latentes y_i também de dimensionalidade f. Este novo vetor representa o entendimento do feedback implícito a partir de uma análise do feedback explícito.

$$\hat{r}_{ui} = \mu + b_i + b_u + q_i^T(p_u + |R(u)|^{-\frac{1}{2}} \cdot \sum_{j\in R(u)} y_i) \tag{2.19}$$

A atualização das variáveis também é modificada no SVD++ em relação ao SVD. A formulação a seguir demonstra que na atualização dos fatores latentes q_i e p_u existe um novo hiper parâmetro δ, além de uma adição de expressões na atualização do q_i. A formulação também possui a atualização do novo vetor de fatores latentes y_i.

- $b_u = b_u + \gamma \cdot (e_{ui} - \lambda \cdot b_u)$
- $b_i = b_i + \gamma \cdot (e_{ui} - \lambda \cdot b_i)$

- $q_i = q_i + \gamma \cdot (e_{ui} \cdot (p_u + |R(u)|^{-\frac{1}{2}} \cdot \sum_{j \in R(u)} y_i) - \delta \cdot q_i)$
- $p_u = p_u + \gamma \cdot (e_{ui} \cdot q_i - \delta \cdot p_u)$
- $\forall j \in R(u)$:
  $y_i < -y_i + \gamma \cdot (e_{ui} \cdot |R(u)|^{-\frac{1}{2}} \cdot q_i - \delta \cdot y_i)$

O recomendador SVD++ possui sete hiper parâmetros, sendo os mesmos seis do SVD e um δ novo. Algumas implementações associam o valor de λ ao δ, Koren e Bell (2015) associam valores diferentes e também sugerem valores não otimizados para domínios.

2.6.4.3 NMF Um outro recomendador que pertence ao grupo de fatorizadores de matrizes é o NMF. Este algoritmo também possui diversas variações como debatido por Zhang et al. (2006) e por Luo et al. (2014). O NMF trabalha com uma matriz não negativa possibilitando encontrar fatores latentes não negativos. Luo et al. (2014) apresentam formalmente uma implementação do recomendador.

O NMF segue a predição do valor do feedback apresentado nas Equações 2.15 e 2.16 de modo semelhante às outras formulações dos recomendadores decompositores explanados acima, com exceção da atualização dos vetores dos fatores latentes q_i e p_u, estes que possuem uma nova formulação.

$$p_u = p_u \cdot \frac{\sum_{i \in I_u} q_i \cdot r_{ui}}{\sum_{i \in I_u} q_i \cdot \hat{r}_{ui} + \lambda_u |I_u| \cdot p_u} \tag{2.20}$$

$$q_i = q_i \cdot \frac{\sum_{u \in U_i} p_u \cdot r_{ui}}{\sum_{u \in U_i} p_u \cdot \hat{r}_{ui} + \lambda_i |U_i| \cdot q_i} \tag{2.21}$$

Onde I_u representa todos os feedbacks do usuário u e U_i representa todos os feedbacks recebidos pelo item i. O NMF possui ao todo oito hiper parâmetros, sendo os seis apresentados no SVD e SVD++, e os dois novos hiper parâmetros para controlar os vetores de fatores latentes, sendo o λ_i para controlar o q_i e o λ_u para controlar o p_u.

2.6.5 Slope One

Lemire e Maclachlan (2007) apresentam o Slope One, um recomendador construído com poucas linhas de código e suscetível a atualizações em tempo real. Os autores afirmam que o recomendador obtém precisão idêntica a recomendadores baseados em memória. O Slope One possui implementações que são baseadas em modelos e em memória como apresentado por Lemire e Maclachlan (2007). Assim, na Tabela 2.3 é apontado como um algoritmo baseado em modelo, o qual é apresentado a seguir.

$$dev(i,j) = \frac{1}{|U_{ij}|} \sum_{u \in U_{ij}} r_{ui} - r_{uj} \tag{2.22}$$

$$\hat{r}_{ui} = \mu_u + \frac{1}{|R_i(u)|} \sum_{j \in R_i(u)} dev(i,j) \tag{2.23}$$

Onde U_{ij} representa todos os usuários que possuem feedbacks sobre o item i e j, sendo r_{ui} o valor do feedback do usuário u sobre o item i e r_{uj} o valor sobre o item j. $R_i(u)$ representa o conjunto de itens j que possuem feedbacks de u que também possui pelo menos um usuário em comum com o item i. Por fim, μ_u representa a média dos valores dos feedbacks do usuário u.

2.6.6 Co-Clustering

Os algoritmos de agrupamento baseiam-se em mapear itens ou usuários que possuam similaridades sob um único grupo. George e Merugu (2005) apresentam uma implementação de um recomendador que utiliza tanto os grupos dos itens quanto os grupos dos usuários para encontrar recomendações. A seguir é apresentada a formulação.

$$\hat{r}_{ui} = \overline{C_{ui}} + (\mu_u - \overline{C_u}) + (\mu_i - \overline{C_i}) \tag{2.24}$$

A Equação 2.24 apresenta a formulação que o recomendador aplica para predizer um feedback do usuário u sobre um item i, que é representado por $\hat{r}_{ui}$. $\overline{C_{ui}}$ representa a média dos feedbacks do grupo correlacionado do usuário u com o item i. μ_u representa a média dos feedbacks do usuário u. μ_i representa a média dos feedbacks que o item i recebeu. $\overline{C_u}$ representa a média dos feedbacks do grupo ao qual o usuário u faz parte. $\overline{C_i}$ representa a média dos feedbacks do grupo ao qual o item i faz parte.

O recomendador possui três hiper parâmetros: i) "Épocas", que representa quantas vezes o recomendador itera sobre os dados; ii) γ, que representa o tamanho do grupo de itens; iii) ρ, que representa o tamanho do grupo de usuários.

2.7 FILTRAGEM HÍBRIDA

Burke (2002), em sua pesquisa, elucida que "um sistema de recomendação híbrido combina duas ou mais técnicas de recomendação para ganhar uma melhor performance com menos desvantagens que qualquer uma individual". Ricci, Rokach e Shapira (2011) explicam que "um sistema híbrido combina técnicas A e B tentando usar as vantagens de A para consertar as desvantagens de B".

Normalmente os sistemas de recomendação híbridos combinam as técnicas de filtragem colaborativa e filtragem baseada em conteúdo. Entretanto qualquer técnica pode ser combinada, desde que satisfaça os requisitos do domínio da aplicação. Ao combinar as técnicas as desvantagens são reduzidas a partir de uma vantagem de uma outra técnica, assim combinar técnicas que cubram as desvantagens umas das outras é o melhor para se obter uma melhor precisão (BURKE, 2002).

2.8 AVALIAÇÃO DE SISTEMAS DE RECOMENDAÇÃO

Nesta seção são abordadas as formas de avaliação dos Sistemas de Recomendação, como: os protocolos, as métricas e suas formulações.

2.8.1 Protocolos de avaliação

De acordo com Shani e Gunawardana (2011), para entender o comportamento dos Sistemas de Recomendação e compará-los, existem três diferentes protocolos de avaliação: *offiline*, estudos de casos dos usuários e *online*. Os protocolos podem ser aplicados a qualquer técnica de recomendação e seus recomendadores. Sistemas de Recomendação que serão utilizados com usuários reais normalmente aplicam os três protocolos, buscando primeiro entender o sistema com o protocolo *offline*, depois estudar o comportamento do sistema com alguns estudos de casos e por fim aplicar o protocolo *online* onde uma interação com usuários reais é exigida.

2.8.1.1 Offline Shani e Gunawardana (2011) afirmam que "o protocolo de avaliação *offline* é o mais fácil de ser utilizado e não requer interação do usuário com o sistema". Normalmente o protocolo é utilizado com um conjunto de dados pré-coletados dos usuários, itens e as transações. Esse conjunto de dados é utilizado para simular o comportamento dos usuários e o poder de predição dos algoritmos avaliados. Este protocolo nos permite filtrar comportamentos inapropriados dos algoritmos recomendadores, permitindo assim escolher o algoritmo mais adequado para o sistema. O protocolo *offline* permite que estudos de enviesamento possam ser melhor desenvolvidos, assim como uma avaliação do comportamento dos usuários. Além do estudo do enviesamento, o protocolo nos permite trabalhar com diferentes modelagens dos dados, dando vez assim para um estudo de uma modelagem mais complexa e eficiente do usuário.

2.8.1.2 Estudos de casos Sistemas de Recomendação necessitam da interação dos usuários, assim somente o protocolo *offline* não é suficiente para entender o comportamento dos mesmos. Entretanto pular direto para a disponibilização do sistema *online* também não é aconselhável. Neste caso, os estudos de casos são conduzidos. Este protocolo se baseia na escolha de um grupo limitado de usuários para interagir com o sistema, permitindo, assim, que diversos dados possam ser coletados, tanto qualitativos quanto quantitativos. Além destes dados é possível que o especialista do sistema observe determinados comportamentos que o protocolo *offline* não captura. Questionários podem ser aplicados durante todo o processo de interação do usuário.

2.8.1.3 Online De acordo com Shani e Gunawardana (2011), "Sistemas de Recomendação realistas são produzidos para influenciarem o comportamento dos usuários, assim como identificar possíveis mudanças de comportamento ao interagir com diferentes algoritmos recomendadores". O real desempenho de um Sistema de Recomendação ao interagir com usuários reais depende de uma variedade de fatores, sendo todos eles cruciais para a avaliação da efetividade.

2.8.2 Métricas preditivas

As métricas preditivas desejam comparar os valores preditos com os valores reais atribuídos pelos usuários (CELMA, 2009). Herlocker et al. (2004) afirmam que "as métricas prediti-

vas medem o quão perto os algoritmos recomendadores preveem os valores dos verdadeiros atribuídos pelos usuários". Eles também afirmam que "essas métricas são particularmente importantes para a avaliação das tarefas de predição que serão exibidas aos usuários". A seguir, é apresentada uma métrica comumente utilizada e aplicada neste trabalho.

2.8.2.1 MAE A métrica Mean Absolute Error (MAE) mede a média do desvio absoluto entre o valor predito e o valor original. MAE é uma métrica comumente usada e possui diversas variações. A Equação 2.25 é a representação formal da métrica.

$$MAE = \frac{1}{|\hat{R}|} \sum_{\hat{r}_{ui} \in \hat{R}} |\hat{r}_{ui} - r_{ui}| \tag{2.25}$$

Onde $\hat{R}$ representa o conjunto das predições (usuário-item), $\hat{r}_{ui}$ representa o valor predito para um feedback $\hat{r}_{ui} \in \hat{R}$ e r_{ui} representa o valor original do feedback. Assim, é aplicado um somatório da diferença entre o predito e o real, e ao final uma média é feita.

2.8.3 Métricas de ranqueamento

Normalmente os Sistemas de Recomendação entregam aos usuários uma lista de tamanho N contendo os top-N itens mais indicados para o determinado usuário ou grupo. Assim, para quantificar a qualidade da lista as métricas de ranque são aplicadas. A seguir, são apresentadas duas das mais usadas e aplicadas neste trabalho.

2.8.3.1 MAP A fim de obter uma métrica única que contribui para a precisão do método de recomendação ao longo de todo o conjunto de usuários, utiliza-se o Mean Average Precision (MAP) (PARRA; SAHEBI, 2013). O valor do *MAP* é obtido calculando a média sobre a precisão média da lista de recomendações de cada usuário.

$$MAP = \frac{1}{|U|} \sum_{u}^{U} AveP(u) \tag{2.26}$$

$$AveP(u) = \frac{1}{N} \sum_{i=1}^{N} p@i \tag{2.27}$$

$$p@i = \frac{r}{i} \tag{2.28}$$

Na Equação 2.26, $AveP(u)$ é a média da precisão para o usuário $u \in U$, isto é, a média dos valores da precisão obtidos para o conjunto de top-N recomendações depois que cada sugestão relevante é recuperada (MANNING; RAGHAVAN; SCHüTZE, 2008). As Equações 2.27 e 2.28 mostram os cálculos da média da precisão, que soma cada posição da lista $p@i$ onde r é a quantidade de músicas relevantes até a posição i.

2.8.3.2 MRR A métrica Mean Reciprocal Rank (MRR) avalia a qualidade das listas de recomendação, medindo o quão longe o primeiro item relevante está do topo da lista (QIN, 2013). Na Equação 2.29 é possível verificar a formulação do MRR onde $\forall u \in U$ a primeira ocorrência de uma recomendação relevante p_i é avaliada.

$$MRR = \frac{1}{|U|} \sum_{u}^{U} \frac{1}{p_i} \tag{2.29}$$

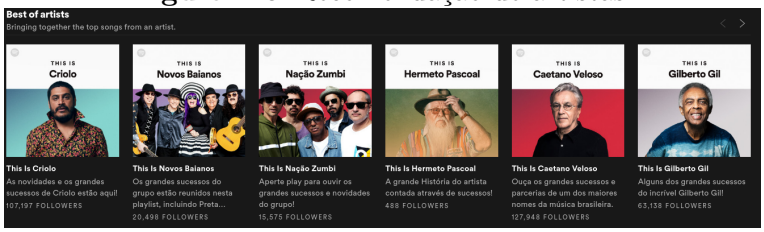
2.9 APLICAÇÕES INDUSTRIAIS DE SISTEMAS DE RECOMENDAÇÃO

Atualmente os Sistemas de Recomendação fazem parte dos nossos dias, estando presentes em: lojas eletrônicas indicando qual item comprar a partir de compras anteriores, transmissão de vídeo ou música onde o que se assistiu ou ouviu serve como base para recomendação de novos itens, cursos pela Internet que recomendam novas classes a partir de cursos previamente terminados. A seguir nomeamos alguns desses sistemas.

2.9.1 Spotify

O sistema de transmissão de música *Spotify*[8], criado em 2008 na Suécia, é um dos principais serviços do setor. Em março de 2020 possuía um catálogo com 50 milhões de faixas musicais e 286 milhões de usuários ativos. A Figura 2.6 demonstra uma recomendação baseada no artista, estes que são recomendados a partir dos artistas que compõem as preferências do usuário. A Figura 2.7 demonstra uma lista de recomendação totalmente personalizada com 30 músicas, que é criada semanalmente, tomando como base as preferências do usuário.

Figura 2.6 Recomendação de artistas.



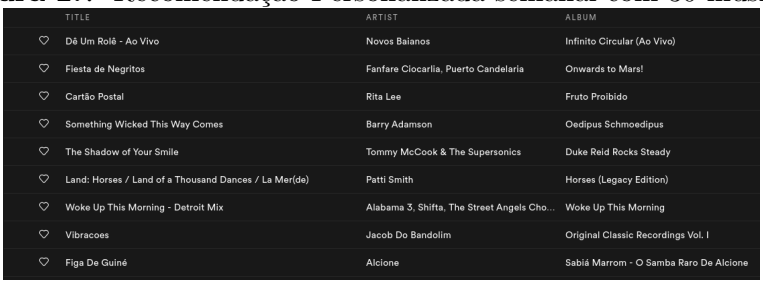
Fonte: Figura capturada pelo autor, Spotify, 2020.

2.9.2 Netflix

Empresa fundada em 1997 nos Estados Unidos com o objetivo de alugar filmes por correio. A Netflix[9] passou por mudanças ao longo de sua história e hoje utiliza a Internet para realizar a transmissão em tempo real do seu catálogo de filmes. Até junho de 2020 a empresa contava com mais de 183 milhões de usuários. A Netflix em 2009 desenvolveu

[8]https://www.spotify.com
[9]https://www.netflix.com

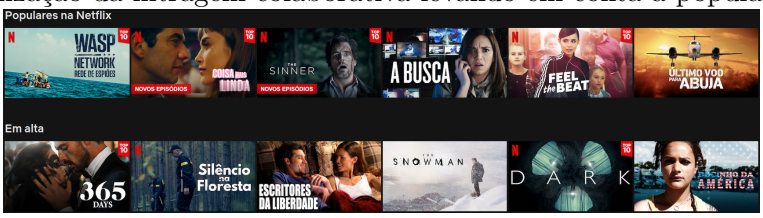
Figura 2.7 Recomendação Personalizada semanal com 30 músicas.



Fonte: Figura capturada pelo autor, Spotify, 2020.

um projeto chamado Netflix Prize[10], no qual ofereceu um prêmio de 1 milhão de dólares para a equipe que melhorasse em 10% o seu poder de recomendação. Atualmente o sistema de recomendação da empresa utiliza diversas técnicas. A Figura 2.8 demonstra a recomendação de filmes, levando em conta a popularidade entre todos os usuários e a popularidade momentânea.

Figura 2.8 Utilização da filtragem colaborativa levando em conta a popularidade dos itens.



Fonte: Figura capturada pelo autor, Netflix, 2020.

2.10 SUMÁRIO

Neste capítulo foram apresentados os principais conceitos e tarefas de um sistema de recomendação. Também foi explanado como modelar os dados dos usuário. Em sequência foram abordadas as técnicas de recomendação, debatendo detalhadamente cada técnica, abordando seus conceitos, algoritmos, vantagens e desvantagens. Também foram apresentadas formas de avaliar os sistemas de recomendação, em seus protocolos e métricas. Por fim, algumas aplicações industriais de sucesso foram apresentadas. No próximo capítulo será abordado o estado da arte sobre o tema que este trabalho visa debater.

[10]https://www.netflixprize.com/

Capítulo 3

JUSTIÇA E CALIBRAGEM

Como visto, Sistemas de Recomendação utilizam técnicas para analisar as preferências dos usuários e encontrar listas de itens que venham a possuir o máximo de relevância para os usuários. As técnicas são focadas em encontrar itens que sejam relevantes para o usuário, buscando a similaridade no texto, nas preferências ou misturando ambas. Entretanto, a busca por relevância não contempla outros aspectos importantes para os usuários, como: novidade, diversidade, surpresa ou justiça. Assim, o sistema necessita observar outros aspectos além da relevância para melhor agradar aos usuários e entendê-los.

A justiça nas recomendações vem para analisar as preferências dos usuários e fazer jus com sua variedade ou não de preferências. Os estudos de justiça para sistemas de recomendação abordam diversos temas como: justiça para múltiplos lados (BURKE, 2017; ABDOLLAHPOURI; BURKE, 2019), justiça no enviesamento por popularidade (ABDOLLAHPOURI; BURKE; MOBASHER, 2017; MEHROTRA et al., 2018) e, recentemente, estudos sobre recomendações calibradas (STECK, 2018; KAYA; BRIDGE, 2019; ABDOLLAHPOURI et al., 2019b, 2020; LIN et al., 2020). Calibragem não define justiça, mas sim, é um meio de prover algum grau de justiça para os usuários, gerando uma lista de recomendação baseada nas classes das preferências.

Neste capítulo é apresentado o contexto do nosso estudo que visa ir além da relevância, abordando os conceitos que envolvem justiça e calibragem de recomendações, além de apresentar trabalhos relacionados, pontuando prós e contras de cada trabalho.

3.1 JUSTIÇA

Justiça nas decisões algorítmicas significa que o sistema não deve discriminar características individuais como: cor, raça, localidade, idade, gosto cinematográfico ou musical. Como pontuado por Zliobaite (2015), tal conceito vem crescendo consideravelmente na área de aprendizado de máquina. Este conceito também vem ganhando destaque na área de sistemas de recomendação (BURKE, 2017; STECK, 2018). Ao aplicar o conceito de justiça em sistemas de recomendação deve-se adicionar o critério de personalização, considerando que os itens mais relevantes ao usuário u podem ser diferentes para o usuário

v. Sendo assim, a personalização é o pilar de suporte dos sistemas de recomendação principalmente em contextos como: música, filmes e livros; em que as preferências do usuário guiam o sistema para a criação da lista de recomendação.

Como exemplo de justiça e personalização considere um sistema de recomendação de vagas de trabalho. Este sistema para ser justo deve considerar que homens brancos e mulheres negras com o mesmo nível de qualificação profissional recebam oportunidades iguais de recomendação de trabalho mantendo a similaridade no ranque e salários. Entretanto se existem diferenças em seus currículos profissionais, o sistema deve se adaptar a essas tendências profissionais e personalizar a lista. Outras formas de personalização também podem alterar o resultado da lista de sugestões. Por exemplo, no sistema de recomendação de vagas de trabalho, alguns usuários podem preferir um salário menor desde que ganhem contrapartidas como: horário flexível, menor tempo de traslado ou benefícios melhores. Assim, o algoritmo deve ser construído para adaptar-se a essas personalizações requeridas pelo usuário.

3.1.1 Divisão dos atores

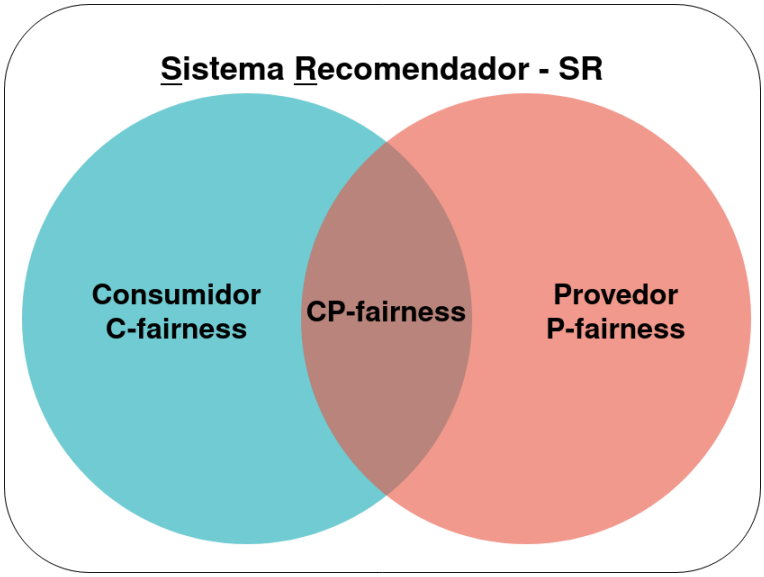
Burke (2017) apresenta o conceito de justiça aplicado a um sistema de recomendação *multistakeholder* que considera mais do que as preferências do usuário no momento de formar a lista de recomendação. O autor propõe uma divisão dos *stakeholders* em três categorias: Consumidores - C, Provedores - P e o Sistema de Recomendação - SR.

- **Consumidores - C:** São os que recebem as recomendações. Em geral são os usuários do sistema, podem ser pessoas, empresas, países ou grupos;
- **Provedores - P:** São aqueles que suportam ou estão por trás das recomendações e ganham algo quando escolhidos pelos consumidores, normalmente sendo caracterizados como itens, mas em alguns domínios podem ser empresas ou grupos de pessoas, por exemplo gravadoras e bandas;
- **Sistema de Recomendação - SR:** Este que ganha sempre que um provedor é apresentado a um consumidor de forma que lhe agrade ou ao prover uma interação positiva entre ambos.

Sistemas de Recomendação *Multistakeholders* adotam medidas que possibilitam a todos os lados tratamento de forma justa. Para isso, critérios relacionados à justiça necessitam ser definidos indo além da relevância durante a avaliação. Burke (2017) apresenta três classes de Sistemas de Recomendação diferenciados pela necessidade da aplicação de justiça (Figura 3.1): justiça nos consumidores (C-fairness), justiça nos provedores (P-fairness) e justiça para ambos (CP-fairness). Posteriormente, o estudo de Burke (2017) foi estendido no trabalho de Abdollahpouri e Burke (2019).

- **C-fairness:** Sistemas de Recomendação *C-fairness* devem considerar o impacto de classes protegidas no consumidor na hora de encontrar a lista de sugestão. Por exemplo, no caso de um banco que recomenda linhas de crédito para os seus clientes, o cliente é o foco da recomendação e dados individuais como cor, bairro e idade não

Figura 3.1 Diagrama contendo os múltiplos lados envolvidos na recomendação.



Fonte: Figura elaborada pelo autor.

devem ser considerados como informação para gerar a recomendação da linha de crédito. No caso de filmes e músicas os sistemas normalmente são construídos para obter a máxima precisão na sugestão usando as preferências de cada usuário como base do processo;

- **P-fairness:** Um sistema de recomendação *P-fairness* preserva a necessidade de justiça no provedor, apenas. Este sistema não garante nenhum tipo de justiça ao consumidor. Comparados aos consumidores, os provedores são mais passivos nos sistemas, devido a não estarem na busca por recomendações e sim por uma oportunidade para serem recomendados aos consumidores. Por exemplo, o Kiva[1], um site de patrocínio coletivo, visa que todos os seus projetos, nas 8 regiões e 80 países que atendem, possam receber a mesma oportunidade de serem financiados;

- **CP-fairness:** Plataformas que desejam aplicar justiça tanto para o consumidor na hora de receber as recomendações quanto para o provedor que estará nas recomendações constituem a última classificação, chamada *CP-fairness*. Por exemplo, um sistema de recomendação de emprego é caracterizado como um *CP-fairness* em que o trabalhador deseja o melhor emprego e a empresa deseja o melhor trabalhador para o cargo, levando em consideração a necessidade por justiça em ambas as partes. Um outro exemplo é o estudo de Mehrotra et al. (2018), que aborda um sistema de recomendação de música com justiça para consumidores (usuários) e provedores (artistas).

[1] kiva.org

3.1.2 Framework conceitual

Sacharidis, Mouratidis e Kleftogiannis (2019) apresentam um *framework* para o conceito de justiça, no qual é necessário especificar dois componentes: 1) Classificação do item; 2) Distribuições sobre as classes. A exata definição desses componentes depende do domínio em que o conceito de justiça é aplicado. Além de propor o *framework* os autores também propõem o uso de um algoritmo guloso para reordenar a lista em um pós-processamento.

- **Classificação do item:** Assume-se que os itens $i \in I$ estão definidos dentro de um conjunto de classes ou gêneros $g \in G$; Assim, $p(g|i)$ é denotado como o valor da distribuição da classe g em um item i e $p(g|u)$ é denotado como o valor da distribuição da classe g em um usuário u;

- **Distribuição Alvo (DA):** É uma distribuição discreta de probabilidade ou não sobre G e representa a distribuição desejada para cada classe $g \in G$ na lista de recomendação. Existem algumas formas de obter a distribuição, por exemplo, uma distribuição uniforme sobre as classes, ou o sistema de recomendação pode avaliar o histórico do usuário para determinar estes valores da distribuição. Steck (2018) apresenta uma distribuição baseada nos gêneros dos itens que estão presentes nas preferências do usuário;

- **Distribuição Realizada (DR):** A distribuição realizada representa a distribuição discreta de probabilidade ou não sobre um conjunto de classes G, capturando a proporção de cada classe na lista de recomendação. Existem várias maneiras de encontrar o valor da distribuição. Um forma simples é somar o peso que cada classe tem em um item. Outra abordagem é considerar o ranque com fator enviesador, isto é devido aos sistemas de recomendação colocarem os itens que o usuário preferirá mais ao topo da lista. Steck (2018) apresenta uma distribuição que usa o Mean Reciprocal Rank (MRR) ou o normalized Discounted Cumulative Gain (nDCG) como formas de obter o peso do ranque.

Figura 3.2 Gêneros e distribuições utilizadas durante o *framework* conceitual.

Fonte: Figura elaborada pelo autor.

A Figura 3.2 apresenta conceitos que envolvem o *framework*: a distribuição alvo com itens $i \in DA$ e a distribuição realizada com itens $j \in DR$ que pertencem a classes $g \in G$ definidas pelas cores; a distância entre as distribuições que é dada por $F(DA, DR)$, onde seus valores $[0, \infty]$ significam que quanto menor o valor menor será a distância entre as distribuições. O resultado de $F(DA, DR)$ pode ser encontrado usando qualquer medida de distância entre duas distribuições.

3.2 ESTUDOS RELACIONADOS A JUSTIÇA

Kamishima, Akaho e Asoh (2012) apresentam uma solução para o problema do filtro bolha. Os autores propõem um sistema de recomendação que garante neutralidade nas sugestões. Para realizar a tarefa é necessário considerar neutralidade a partir de pontos de vista. Por exemplo, se o usuário especifica um atributo para uma busca a neutralidade vem de não permitir que os dados protegidos do usuário influenciem nos resultados. Para garantir essa neutralidade eles apresentam um regulador. Entretanto, entende-se que uma completa neutralidade quando se usa os dados do usuário é impossível.

Hardt, Price e Srebro (2016) propõem noções de não discriminação nas predições respeitando classes protegidas do usuário. Eles apresentam um algoritmo não discriminatório que utiliza os dados de treinamento X para predizer as classes Y respeitando a não discriminação do atributo protegido A. Eles também propõem formas de mensurar a discriminação na predição: *Equalized Odds* e *Equal opportunity*.

- **Equalized Odds:** Diz-se que um preditor $\hat{Y}$ satisfaz *Equalized Odds* com respeito ao atributo A e a saída Y, se $\hat{Y}$ e A são independentes em Y;
- **Equal opportunity:** É considerar uma não discriminação nas classes positivas de Y, por exemplo, ser aceito em uma universidade ou conseguir um empréstimo. Esta adaptação permite que quem está requisitando um empréstimo pela primeira vez tenha a mesma chance de quem já pegou e pagou outros empréstimos.

Equalized Odds e *Equal opportunity* nos ajudam a entender que existe uma propagação do atributo protegido A em outros atributos de X, demostrando, assim, a necessidade da independência entre os dados.

Yang e Stoyanovich (2017) comparam métricas estatísticas para divergência em ranque, considerando o problema em duas partes: primeiro gerar saídas mais interpretáveis e transparentes, segundo quantificar a justiça para ajudar a melhorar os algoritmos. Por natureza, a atividade de classificar um item em positivo ou negativo é diferente de ranquear itens, devido ao fato de que ranque top-5 precisa ser mais apropriado que o top-10 que, por sua vez, é preciso ser mais apropriado que o top-20. Os itens no topo da lista ganham benefícios, devido a isso é importante calcular a justiça a partir do topo da lista. As primeiras posições do ranque necessitam ser mais justas do que as do final da lista, assim é aplicado um desconto logarítmico como o nDCG.

Liu e Burke (2018) apresentam um sistema de recomendação baseado no contexto de justiça para os provedores. Para isso eles propõem um algoritmo (*FAR/PFAR*) que em um pós-processamento ordena o ranque balanceando entre acurácia e justiça nos provedores.

Eles apresentam que ao adicionar justiça nos provedores permitindo que todos tenham chances semelhantes, a acurácia é pouco afetada. O algoritmo *FAR* leva em consideração variáveis de balanceamento global, e o *PFAR* busca personalizar o balanceamento entre acurácia e justiça para cada usuário.

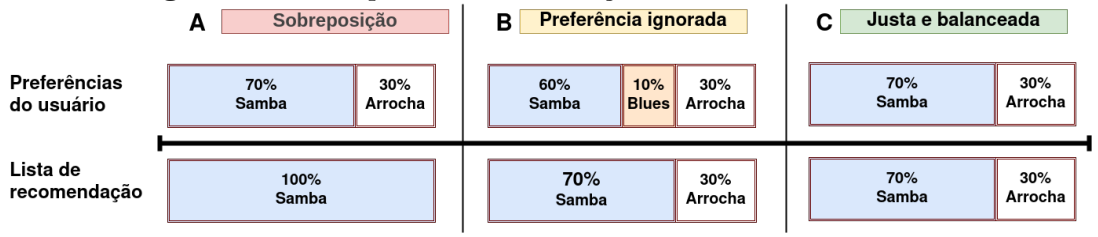
3.3 CALIBRAGEM

Sistemas de Recomendação baseados em relevância encontram listas de recomendação onde seus itens são o máximo relevantes aos usuários, focando assim nas principais áreas de sua preferência. A busca pelos itens mais relevantes é dada por uma função de similaridade entre cada item e as preferências do usuário. Isto pode causar um desbalanceamento dos gostos nas recomendações, devido ao sistema reproduzir itens similares ao da classe dominante, sobrepondo as áreas de menor interesse ou recentemente adquiridas e em alguns casos esses interesses podem nem ser representados nas recomendações, levando o usuário a um ambiente de superespecialização. Este problema também pode ser encontrado em contas compartilhadas, em que um grupo de pessoas utilizam a mesma conta, nas quais o interesse de um usuário pode subjugar os outros (STECK, 2018).

Espera-se de uma recomendação que ela agrade ao usuário mas também venha a fazer jus a suas preferências. Assim, por exemplo, se as preferências do usuário possuem 70% de filmes de Comédia e 30% de filmes de ação, espera-se que as recomendações venham a fazer justiça a essa proporção, calibrando a lista de recomendação baseando-se nas preferências, o que não acontece em sistemas voltados somente para a relevância.

Na Figura 3.3 é possível visualizar três exemplos de recomendação, duas descalibradas (A e B) e uma calibrada (C). O bloco A possui uma preferência composta por 70% Samba e 30% Arrocha, entretanto nas suas recomendações Samba subjugou o gênero Arrocha. No bloco B o gênero Rock obteve uma maior proporção da lista de recomendação, o que não condiz com as preferências. E no bloco C, onde todos os gêneros recebem a mesma proporção na lista de recomendação, diz-se que a recomendação é justa e calibrada.

Figura 3.3 Exemplos de recomendação calibrada e descalibrada.



Fonte: Figura elaborada pelo autor.

Neste trabalho, foi estudada a recomendação calibrada, que é uma forma de prover algum grau de justiça na lista de recomendação. Um sistema de recomendação é considerado calibrado quando a proporção das classes na lista de sugestão é concisa com a proporção do histórico de preferências. Ao invés da discriminação ser contra características individuais de pessoas, aqui justiça será respeitar a variedade de interesses.

O conceito de *C-fairness* pode ser aplicado a recomendações calibradas, onde os

usuários devem receber recomendações de acordo com seu histórico de preferências. A distribuição alvo é especificada para cada usuário e é representada pela proporção de cada classe em suas preferências. A distribuição realizada é a lista de recomendação encontrada. O processo de calibragem das recomendações é realizado após o sistema de recomendação filtrar e criar um conjunto de itens que o usuário pode vir a gostar, realizando assim uma reordenação na lista de recomendação.

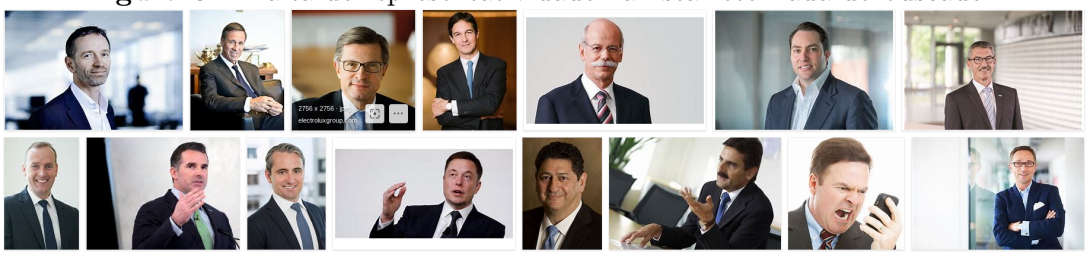
Como descrito por Steck (2018) e Kaya e Bridge (2019), a calibragem é realizada como uma etapa de pós-processamento do sistema. A primeira etapa é um pré-processamento, onde os dados são preparados como entrada do sistema. A etapa de processamento é onde o sistema de recomendação aprende sobre as preferências do usuário e seleciona itens que possam vir a agradá-los. A calibragem ocorre nesta terceira etapa chamada de pós-processamento, onde os itens selecionados pela etapa de processamento são dados como entrada ao pós-processamento para que assim uma lista de recomendação calibrada possa ser gerada ao término do sistema.

3.3.1 Enviesamento

Os sistemas de recomendação focados em relevância tendem a enviesar as sugestões. Algoritmos de filtragem colaborativa enviesam as sugestões com tendências baseadas no que outros usuários gostaram, normalmente levando à recomendação de itens mais conhecidos, como afirmado por Abdollahpouri, Burke e Mobasher (2017). Já algoritmos de filtragem baseada em conteúdo sofrem com a superespecialização nas preferências do usuário, recomendando os itens com o maior nível de similaridade.

O enviesamento não é sinônimo de algo ruim, pelo contrário, sistemas de recomendação utilizam este enviesamento para saber o que usuário pode vir a gostar. Entretanto, em alguns contextos esse viés pode ser problemático. Por exemplo, a Figura 3.4 mostra que ao procurar pela palavra "CEO" na ferramenta de busca online Bing[2] a lista retornada é restrita em representatividade e diversidade, focando em apenas uma classe de ser humano e excluindo todos os outros tipos de humanos existentes, reproduzindo um velho e conhecido problema social.

Figura 3.4 Falta de representatividade na lista retornada do buscador.



Fonte: Figura capturada pelo autor, Bing, 2020.

Lin et al. (2019) buscam entender como diferentes algoritmos de filtragem colaborativa propagam o enviesamento e como isso pode afetar os usuários. O estudo indica que

[2]Bing.com

cada algoritmo propaga de forma diferente o enviesamento, sendo que alguns amplificam e outros diminuem. Nos resultados o atributo protegido "sexo" é analisado. Para as mulheres, para as quais a maioria das preferências é composta de filmes de romance, os resultados indicam que o enviesamento é para os filmes de ação, isso é devido aos homens gostarem mais de filmes de ação na base de dados usada. Eles também mostram que os algoritmos baseados em vizinhança amplificam o enviesamento, recomendando itens pertencentes a um grupo majoritário para um grupo minoritário que não necessariamente possui preferência pelo item. Já os algoritmos baseados em fatoração de matrizes mostram o contrário, eles reduzem o enviesamento na lista, focando menos no grupo majoritário.

3.3.2 Cenários de calibragem

Steck (2018) apresenta três cenários como exemplos de calibragem. Em todos os cenários ele utiliza como exemplo gêneros de filmes e um usuário que prefere 70% romance e 30% ação, sendo que um filme possui exclusivamente um gênero.

3.3.2.1 Classes desbalanceadas No primeiro e mais extremo cenário apresentado por Steck (2018), é considerado que se conhecesse apenas os gêneros das preferências dos usuários. Este problema é análogo à classificação não balanceada no aprendizado de máquina. É conhecido que a melhor forma de obter acurácia nas predições é sempre prever a classe majoritária. No problema de classificação binária em que 70% dos dados são classificados com a classe positiva e 30% classificados com a classe negativa, o melhor é predizer a classe positiva para todos os dados de entrada e com isso espera-se acertar 70% das classificações. Em contraste, se é predito aleatoriamente as classes positiva e negativa com a probabilidade de 70% e 30% (exatamente como ocorre nos dados de treinamento), pode-se esperar que a predição esteja correta para somente $0{,}7 \cdot 70\% + 0{,}3 \cdot 30\% = 58\%$ dos dados de entrada.

Traduzindo para o exemplo em que o usuário prefere 70% romance e 30% ação, pode-se obter a melhor acurácia se for recomendado 100% das vezes filmes de romance para o usuário e nenhum filme de ação.

Steck (2018) afirma que o cenário de escassez de informações possuindo nada além do gênero é muito extremo e argumenta que no mundo real, existirão mais informações disponíveis de alguma forma, mas de todo jeito sempre existirá um limite para os dados.

3.3.2.2 Variação das probabilidades Para este cenário apresentado por Steck (2018), é necessário desenvolver um experimento mental, onde assume-se que cada filme i possui uma probabilidade $p(i|g)$ de ser reproduzido caso o usuário u decida assistir um determinado gênero g. Previamente foi apresentado $p(g_r|u) = 0,7$ e $p(g_a|u) = 0,3$, se o usuário u decidir reproduzir filmes dos gêneros romance g_r e ação g_a. Ao assumir que os filmes possuem exclusivamente um gênero, a probabilidade do usuário u assistir ao filme i do gênero g é dada por $p(i|u) = p(i|g) \cdot p(g|u)$. Para a melhor acurácia considera-se a lista de recomendação com os 10 filmes de maior probabilidade $p(i|u)$ de serem reproduzidos por u. Ao considerar o filme i de ação g_a mais provável de ser escolhido $i_{g_a,1}$ e o décimo filme mais provável de romance $i_{g_r,10}$, assim é obtido:

$$\frac{p(i_{g_r,10}|u)}{p(i_{g_a,1}|u)} = \frac{p(i_{g_r,10}|g_r)}{p(i_{g_a,1}|g_a)} \cdot \frac{p(g_r|u)}{p(g_a|u)} = \frac{1}{2,1} \cdot \frac{0,7}{0,3} = \frac{0,7}{0,63} > 1 \tag{3.1}$$

onde os valores $p(i_{g_r,10}|g_r) = 1$ e $p(i_{g_a,1}|g_a) = 2,1$ são determinados sobre a base de dados pública de filmes *MovieLens 20 Million* e apresentado no estudo de Harper e Konstan (2015). Como pode-se verificar pela Equação 3.1, neste cenário o décimo filme de romance possui uma maior probabilidade de ser reproduzido pelo usuário u do que o filme de maior probabilidade do gênero de ação. Assim, observando a acurácia, os 10 filmes recomendados neste cenário são novamente os de romance e nenhum de ação.

3.3.2.3 Latent Dirichlet Allocation Neste ultimo cenário apresentado por Steck (2018) é utilizado como inspiração o Latent Dirichlet Allocation (LDA)(BLEI; NG; JORDAN, 2003), que descreve o processo de o usuário selecionar um filme em dois passos: o usuário primeiro seleciona um gênero g e então o filme i que pertence ao gênero selecionado. Existem três razões para mencionar o LDA.

Primeiro, é assumido que no mundo real o usuário realmente segue esses dois passos para selecionar um filme, então o LDA é o modelo recomendado. Quando o modelo é treinado, ele se torna hábil a capturar o balanceamento correto dos interesses do usuário, e com sua devida proporção. Recomendações balanceadas são esperadas quando se segue o processo, onde os filmes da lista de recomendação são gerados interativamente adicionando-se um a um: primeiro um gênero g é escolhido pelo aprendizado a partir da distribuição $p(g|u)$ para o usuário u, seguido da escolha do filme i pelo aprendizado a partir da distribuição $p(i|g)$. A escolha do filme é obtida através do resultado do ranqueamento de acordo com a probabilidade $p(i|u) = \sum_g p(i|g) \cdot p(g|u)$. Entretanto, como já mostrado nos cenários anteriores, ao ordenar a lista de recomendação baseando-se na probabilidade de um filme i, têm-se uma lista descalibrada.

Segundo, nota-se que o problema de classes desbalanceadas não está restrito ao caso de quando possui-se categorias explícitas, como o gênero, mas aplicado também a casos de tópicos latentes ou embutidos que normalmente usam LDA como modelo.

Terceiro, o problema das recomendações desbalanceadas pode surgir independentemente do fato de um filme pertencer a um único gênero ou pertencer a vários gêneros.

3.4 ESTUDOS RELACIONADOS A CALIBRAGEM

Nesta seção são apresentados os estudos relacionados com justiça e calibragem. A maioria dos trabalhos sobre justiça e calibragem utilizam técnicas de filtragem colaborativa e extensa análise das bases de dados.

Abdollahpouri, Burke e Mobasher (2017) apresentam um *framework* para otimizar a questão de justiça no topo das recomendações. O algoritmo usado por eles propõe um balanceamento entre precisão e cobertura da cauda longa das recomendações, este que visa regularizar e controlar um dos maiores problemas da filtragem colaborativa, o enviesamento por popularidade. O estudo mostrou a importância do balanceamento entre os itens mais populares e itens ainda não conhecidos.

Steck (2018) apresenta o método de recomendação calibrada em um sistema de recomendação de filmes usando a técnica de filtragem colaborativa que calibra a lista de recomendação baseando-se nos gostos prévios dos gêneros do usuário. O autor explica a importância das recomendações calibradas nos três cenários previamente explanados (Seção 3.3.2). A etapa de pós-processamento é composta por um balanceamento entre relevância e justiça. Para controlar o balanceamento um coeficiente/peso é adicionado dos dois lados na formulação.

A relevância é dada como uma soma da similaridade do ranque com o usuário, sendo que no caso de Steck, ele utiliza o peso do ranque. Como medida de justiça, o autor usa o Kullback-Leibler (KL) (KL-divergence), que computa a divergência entre duas distribuições, para isso ele propõe duas distribuições baseadas nos gêneros das preferências do usuário e da lista de recomendação. Como coeficiente de balanceamento, Steck (2018) usa valores constantes que variam de 0 a 0,999, sendo que quanto maior o valor do coeficiente maior o peso para as recomendações virem mais calibradas. Para encontrar os itens que venham a agradar ao usuário na composição da lista de recomendação, o autor usa a relevância marginal máxima em conjunto com um algoritmo guloso chamado Surrogate Submodular. Este algoritmo possui algumas vantagens: entrega uma lista ordenada/ranqueada e, para cada tamanho de lista, ele a cria obtendo um resultado ótimo em $1 - 1/e$, onde e é o número de Euler. Tendo que, encontrar a melhor lista de recomendação calibrada é um problema de combinatória e NP-difícil.

Abdollahpouri et al. (2019b) apresentam um sistema colaborativo que calibra para cada usuário a proporção entre itens famosos (*popularity*) e itens não conhecidos (*long-tail*). Se um usuário prefere 30% de itens não conhecidos e 70% de itens conhecidos, espera-se que a recomendação respeite essa proporção. Eles chegaram à conclusão de que os usuários que gostam de itens menos conhecidos possuem uma maior variedade de itens classificados em suas preferências. Os autores também comparam diferentes algoritmos colaborativos para verificar os itens populares e não populares. Eles dividiram os usuários em três grupos: Focado, Diverso e Nicho. Abdollahpouri et al. (2019b) também demonstram que existe uma grande relação entre o número de itens nas preferências do usuário e o número de itens mais populares. Entretanto a porcentagem desses itens diminui conforme a quantidade de itens nas preferências vai crescendo. O que pode ser tomado como óbvio, sendo que quanto mais itens nas preferências maior a chance de um item muito popular ser adicionado, mas existem mais itens não populares que populares.

Kaya e Bridge (2019) realizam um estudo comparando calibragem com o contexto de intenção (*Intent-aware*). O contexto de intenção é aplicado na área de Recuperação da Informação (Information Retrieval (IR)), por exemplo, um usuário pesquisa "Jaguar" e o sistema não sabe se significa carro, felino ou o sistema operacional da *Apple*. Assim, o contexto de intenção coloca ao menos um item de cada tipo para ser representado nos resultados. Eles apresentam uma variação da métrica de calibragem que utiliza sub-perfis do usuário, além de utilizarem o Subprofile-Aware Diversifcation (SPAD) e o Query Aspect Diversifcation (xQuAD), que são algoritmos para prover diversidade na lista de recomendação. Os resultados indicam que o uso do KL utilizado por Steck (2018) é melhor em captar baixos valores de divergência. Os resultados da precisão indicam que o SPAD obtém uma melhor precisão seguido do KL usando sub-perfil. Quanto à

diversidade indicam que KL com uso de atributos tem uma melhor diversidade na lista quando avaliado com atributos. E o SPAD tem melhor diversidade quando avaliado com sub-perfis. Eles chegaram à conclusão de que *Intent-Aware* implica em calibragem que por sua vez implica em diversidade.

Abdollahpouri et al. (2019a, 2020) apresentam um estudo sobre a conexão de popularidade com a descalibragem. Quando um sistema de recomendação não segue a proporção das preferências do usuário diz-se que ele está descalibrado. Descalibragem, por si só, não quer dizer que o sistema é injusto, mas sim que não está devidamente personalizado, ou seja, não está calibrado. Entretanto, se usuários ou grupos de usuários experimentam diferentes níveis de descalibragem, isso pode indicar um tratamento injusto dos usuários. Os autores indicam que existe uma conexão entre enviesamento por popularidade e a descalibragem, mostrando que usuários com diferentes níveis de itens populares sofrem com a descalibragem e isto resulta em um tratamento injusto. Os mais afetados são os usuários com poucos itens populares. Além disso, os autores avaliaram a descalibragem baseada no sexo do usuário, chegando à conclusão de que na base de dados avaliada, as mulheres sofrem mais com a descalibragem devido a existir uma maior quantidade de homens no sistema. Assim, a base de dados ensina ao aprendizado gostos de uma classe, reproduzindo o aprendizado em uma classe que não possui as mesmas preferências.

Em complementação aos estudos de Abdollahpouri et al. (2019a, 2020), nesta dissertação é definida a calibragem como o grau de concisão da distribuição realizada com a distribuição alvo. Sendo a descalibragem o complemento do valor da calibragem, ou seja, o grau de não concisão entre as distribuições. Supondo uma métrica de descalibragem que possua valores entre $[0, 1] \in \mathbb{R}$, se o resultado obtido é de $0,6$, é entendido que esse valor é o grau de descalibragem entre as distribuições. Em complemento, o grau de calibragem seria de $0,4$. Assim, $0,6 + 0,4 = 1,0$.

Lin et al. (2020) analisam em diversos fatores o efeito da descalibragem da lista de recomendação, efeito que acontece para uns usuários e para outros não, buscando observar as características do usuário e quais dos fatores influenciam na descalibragem. Todas estas características provêm do perfil do usuário e podem ser encontradas na interação do usuário com os itens do sistema. Ao todo os autores trabalham com cinco algoritmos recomendadores, oito fatores divididos em quatro categorias e utilizam duas bases de dados. A pesquisa não aborda a etapa de pós-processamento onde a calibragem é realizada, assim apenas analisam os efeitos da descalibragem nos recomendadores. Os resultados mostram que: todos os algoritmos para ambas as bases de dados possuem descalibragem em diferentes níveis; os dados característicos possuem diferenças entre eles nos resultados; a entropia baseada em categorias obtém resultados promissores se as preferências do usuário possuem um equilíbrio entre as diversas classes às quais os itens pertencem. Classe, como descrito por Lin et al. (2020), é uma palavra genérica para categoria, *tag* ou gênero.

3.4.1 Características dos trabalhos relacionados

A Tabela 3.1 apresenta os estudos relacionados diretamente a calibragem. É possível verificar que todos os estudos aplicam a filtragem colaborativa, sendo que dois dos es-

tudos implementam o pós-processamento e dois não. Para definir os estudos que são chamados de estado-da-arte nesta dissertação, é aplicada uma metodologia de seleção, considerando como parte do estado-da-arte o trabalho que implemente, discuta ou analise o pós-processamento com as formulações voltadas para calibragem. Assim, tem-se que os estudos de Steck (2018) e Kaya e Bridge (2019) a partir desta seção serão chamados de estado-da-arte. Todos os estudos apresentados na tabela são caracterizados como trabalhos relacionados.

Tabela 3.1 Resumo sobre os estudos relacionados focados na calibragem.

Trabalhos Relacionados	Técnica	Possui pós-processamento?	Foco do estudo	Principal ponto
Steck (2018)	Colaborativa	Sim	Formulações	- Recomendações calibradas respeitando os gostos dos usuários
Kaya e Bridge (2019)	Colaborativa	Sim	Formulações	- Mostrou que calibragem implica em diversidade
Abdollahpouri et al. (2019a, 2020)	Colaborativa	Não	Análise	- Demonstra a ligação da popularidade com a descalibragem
Lin et al. (2020)	Colaborativa	Não	Análise	- Realiza experimentos com diversas características do usuário

Fonte: Tabela elaborada pelo autor.

3.5 SUMÁRIO

Neste capítulo foram apresentados os principais conceitos e estudos a respeito de justiça e calibragem. Sobre justiça foram apresentados os principais conceitos, seguidos da descrição de um *framework* conceitual para se trabalhar com justiça e, por fim, os trabalhos relacionados a justiça. Sobre calibragem foi apresentado o problema de enviesamento dos dados implicando em não respeitar o balanceamento das preferências do usuário, seguido dos cenários de calibragem que foram apresentados de acordo com outros estudos e, por fim, os trabalhos relacionados a calibragem. Assim, até o presente capítulo a base deste estudo foi apresentada, que é um sistema de recomendação *C-fairness* que calibra as recomendações baseando-se nas áreas de preferências dos usuários. No capítulo seguinte é apresentada a proposta relacionada diretamente com os trabalhos relacionados.

Capítulo 4

CALIBRAGEM PARA JUSTIÇA EM SISTEMAS DE RECOMENDAÇÃO

No Capítulo 2 foi apresentado como os sistemas de recomendação trabalham e em especial foi dado um foco nos sistemas baseados em filtragem colaborativa, assim como foi apresentado alguns dos seus algoritmos recomendadores. No Capítulo 3 foram abordados os conceitos de justiça em sistemas de recomendação, assim como também foram apresentados diversos trabalhos relacionados a calibragem, debatendo os principais pontos de cada trabalho e relacionando-os entre si.

Neste capítulo são apresentados o protocolo e o modelo de sistema calibrado proposto, juntamente com os componentes necessários para implementar um sistema de recomendação calibrado. O detalhamento de cada uma das três etapas que o sistema executa é apresentado ao longo das seções. Inicialmente são apresentados os modelos de dados usados e suas notações, assim como exemplos. As funções e algoritmos também são formalmente descritos apontando o que está presente no estado-da-arte e o que é exclusivo deste estudo.

A proposta é que o protocolo auxilie na implementação e escolha de qual a combinação de sistema deve ser implementado mediante um conjunto de formulações possíveis para cada componente do protocolo e do sistema. A proposta também é gerar mais de um sistema de recomendação *C-Fairness* baseado em filtragem colaborativa que respeita a distribuição dos gêneros presentes nas preferências de cada usuário, calibrando a lista de recomendação de forma a respeitar ao máximo as preferências dos usuários em suas devidas proporções.

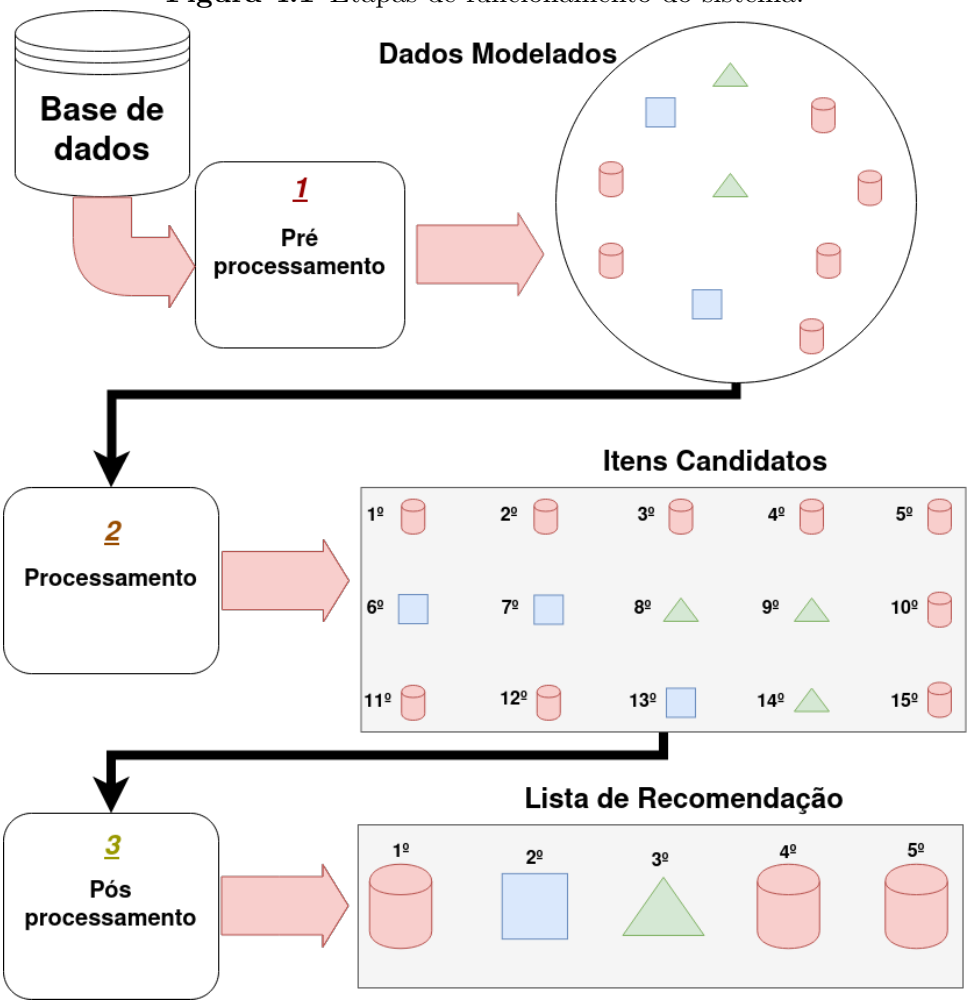
Para conceber as propostas são utilizadas formulações de trabalhos relacionados, como: as distribuições de gêneros, medidas de divergência, medida de ranque, formulação do balanceamento ranque-divergência e o algoritmo *surrogate* de ranqueamento. Como proposta original deste trabalho, têm-se: o uso de uma nova medida de divergência, duas formulações personalizadas para ponderar o balanceamento ranque-divergência, um balanceamento que considera o viés do usuário, duas novas métricas de avaliação para o contexto da calibragem e dois coeficientes decisórios. Além do próprio modelo de sistema

e o protocolo que irá auxiliar na escolha da melhor implementação para uma determinada base de dados.

4.1 MODELO DE SISTEMA DE RECOMENDAÇÃO CALIBRADO

Uma das principais contribuições desta dissertação é um modelo de sistema de recomendação calibrado. Este que incorpora processos comumente utilizados pelos trabalhos relacionados, assim como propõe novos processos. Os sistemas de recomendação, assim como outros sistemas baseados em aprendizado, possuem pré-processamento e processamento, para assim obter os resultados. Cada uma destas etapas possuem componentes que, a depender do objetivo e dos dados utilizados, podem ser adicionados ou retirados. Assim, o modelo de sistema calibrado proposto tem como objetivo levantar as etapas e componentes necessários para o funcionamento, tendo em vista que este modelo pode ser estendido a depender do domínio da aplicação.

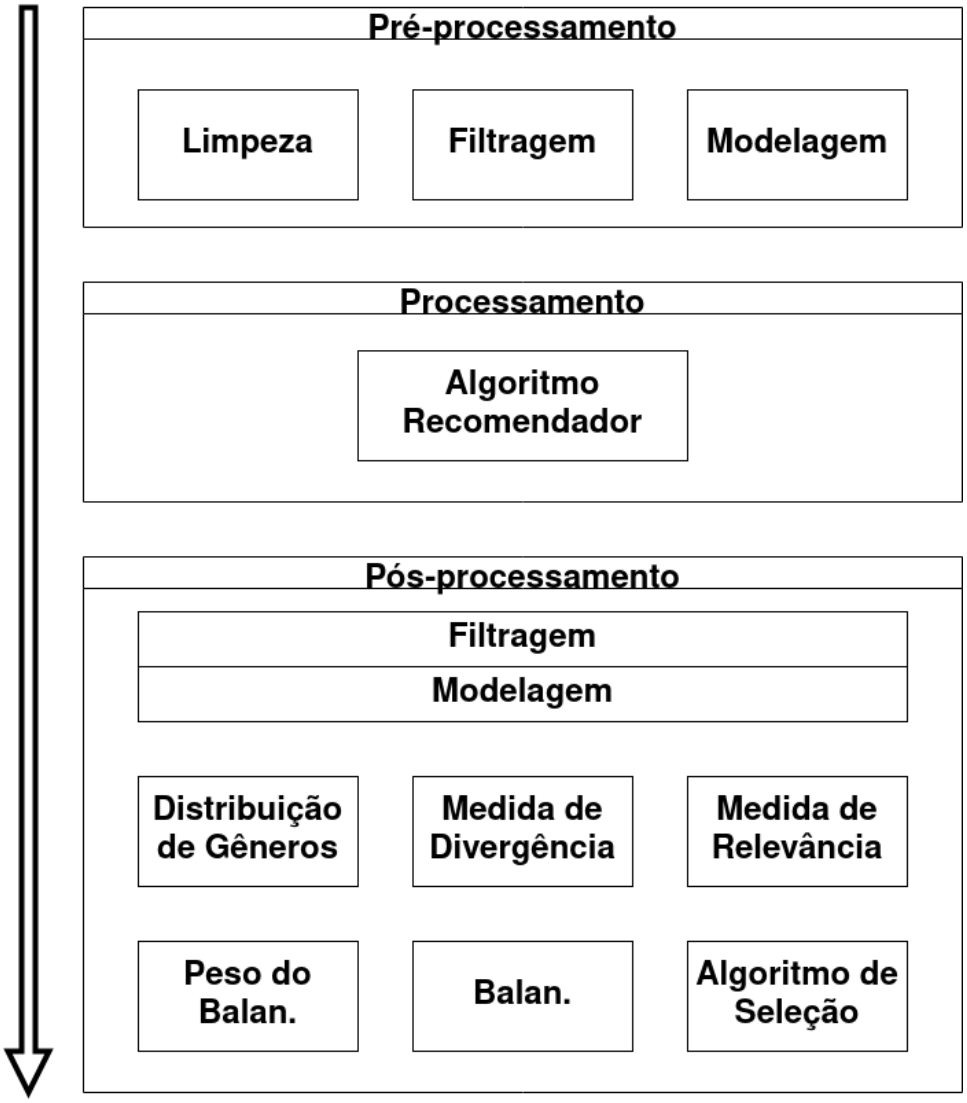
Figura 4.1 Etapas de funcionamento do sistema.



Fonte: Figura elaborada pelo autor.

A Figura 4.1 apresenta o funcionamento do modelo de sistema calibrado proposto, que é dado em três etapas bem definidas com suas devidas entradas e saídas. A saída da etapa anterior é a entrada da próxima etapa até a lista de recomendação ser criada como última saída do sistema. É possível verificar na figura que os itens representados como quadrado azul, triângulo verde e cilindro vermelho, mantêm alguma proporção entre as etapas. Proporção esta que é o foco da calibragem.

Figura 4.2 Componentes que compõem o modelo de sistema calibrado proposto.



Fonte: Figura elaborada pelo autor.

A Figura 4.2 apresenta os componentes de cada etapa do sistema que serão debatidos a seguir. A implementação específica de cada componente depende do domínio do sistema. Diferenças na implementação geram sistemas diferentes, que por sua vez geram diferentes listas de recomendação para cada usuário. Assim, ao longo dos próximos capítulos serão

apresentadas algumas possibilidades de implementação para cada componente.

Etapa 1: **Pré-processamento**: É a primeira etapa, onde os dados são **limpos**, **filtrados** e **modelados**. A limpeza e filtragem é dependente da base utilizada, tendo em vista que cada aplicação coleta os dados de forma diferente. Na Seção 5.1 será abordada a limpeza e filtragem das bases utilizadas no experimento. A modelagem dos dados utilizados pelo sistema é formalmente apresentada na Seção 4.3. Na Figura 4.1 tem-se que a base de dados é carregada na etapa de (1) pré-processamento e tem como saída dados modelados, estes que são passados como entrada da etapa (2) processamento. Na Figura 4.2 é apresentado o pré-processamento com seus componentes internos;

Etapa 2: **Processamento**: É a etapa onde o **algoritmo recomendador** encontra uma lista de itens candidatos que possam integrar a lista de recomendação. Na Seção 2.6 alguns algoritmos são apresentados, assim, nesta etapa qualquer um dos recomendadores descritos pode ser utilizado, desde que aceite como entrada os dados modelados. Na Seção 5.3 são apresentados quais algoritmos foram implementados para o experimento. Na Figura 4.1 tem-se que os dados modelados são passados como entrada na etapa (2) processamento, após o término do processamento itens candidatos são obtidos na saída. Na Figura 4.2 é apresentado que no processamento o único componente é o algoritmo recomendador, entretanto cada algoritmo possui um funcionamento diferente, assim para abstrair os detalhes de cada recomendador, ele será tratado como um componente único;

Etapa 3: **Pós-processamento**: Encontra para cada usuário uma lista de recomendação personalizada que respeite a proporção dos gêneros em suas preferências. Para o contexto de calibragem, o pós-processamento se faz necessário, sendo a etapa da aplicação dos métodos para calibragem. Esta terceira etapa é composta por **8 componentes**, estes que podem ser visualizados na Figura 4.2 e listados a seguir: filtragem (Seção 5.3), modelagem (Seção 4.3.6), distribuição de gêneros (Seção 4.5.1), medida de divergência (Seção 4.5.2), uma medida de relevância (Seção 4.5.3), peso do balanceamento (Seção 4.5.4), balanceamento (Seção 4.5.5) e algoritmo de seleção (Seção 4.5.7). Na Figura 4.1 tem-se que os itens candidatos são modelados como entrada da etapa (3) pós-processamento, onde a lista de recomendação calibrada é criada e dada como saída.

4.2 NOTAÇÕES

Ao longo deste capítulo notações matemáticas são utilizadas para permitir uma explicação formal e coesa do sistema independente dos detalhes de implementação. A Tabela 4.2 apresenta um resumo de todas as notações utilizadas neste e nos próximos capítulos. Conforme os tópicos são abordados, as formulações e notações são contextualizadas. A formalidade apresentada a seguir expande e contextualiza a Tabela 2.1.

Tabela 4.1 Notações que descrevem o sistema e seus modelos.

Símbolo	Descrição
U	Conjunto de todos os usuários
u, v	Um usuário qualquer de U
I	Conjunto de itens
i, j	Um item qualquer de I
G	Conjunto com todos os gêneros
g	Um ou mais gênero(s) quaisquer de G
MU	Conjunto com todos os modelos dos usuários
MU_u	Modelo do usuário u
MIB	Conjunto com todos os modelos de itens bloqueados
MIB_u	Modelo de itens bloqueados do usuário u
MID	Conjunto com todos os modelos de itens desconhecidos
MID_u	Modelo de itens desconhecidos pelo usuário u
MIC^{max}	Conjunto com todos os modelos de itens candidatos maximizados
MIC_u^{max}	Modelo de itens candidatos maximizados para o usuário u
$w_{u,i}$	Valor que o usuário u atribuiu ao item i
$\hat{w}_{r(u,i)}$	Valor predito pelo recomendador de u para i
p_u	Distribuição alvo do usuário u
q_u	Distribuição realizada do usuário u
LR	Todas as listas de recomendação
LR_u	Lista de recomendação do usuário u
S	Combinação de sistema com o melhor desempenho

Fonte: Tabela elaborada pelo autor.

4.3 MODELAGEM DOS DADOS

Independentemente da base de dados usada pelo sistema, os dados necessitam ser formalmente modelados e definidos. Cada modelo define como seus dados são manipulados durante a execução do sistema. Após os dados serem limpos e filtrados no pré-processamento, eles são modelados para serem usados como entrada na etapa de processamento, que por sua vez, modela dados para serem usados como entrada na etapa de pós-processamento.

4.3.1 Conjunto de Usuários

Agradar aos usuários é o objetivo do sistema de recomendação, visto que este é um *C-fairness*. No modelo de sistema proposto, a representação formal para um único usuário é a letra u ou v e para o conjunto de usuários é dado pela letra U. Todo usuário u ou v pertencente ao conjunto U pode receber recomendações. A Equação 4.1 representa formalmente o conjunto dos usuários. Cada $u \in U$ é formado por apenas uma chave única de identificação $u = \langle UID \rangle$ como o apresentado na Tabela 4.2. A formulação do usuário com apenas uma chave transforma o usuário em anônimo.

$$U = \{u_1, ..., u_z | 1 \leq z \leq \mathbb{N}\} \tag{4.1}$$

Tabela 4.2 Exemplo da representação de 3 usuários, cada usuário em uma linha.

UID
U-001
U-002
U-003

Fonte: Tabela elaborada pelo autor.

4.3.2 Conjunto dos Itens

Os itens são os objetos de sugestão do sistema, visto que eles compõem a lista de recomendação. A representação formal para um único item é dada pela letra i ou j e para o conjunto de itens é dado pela letra I. Os itens que compõem I são aqueles que em algum momento foram expostos a alguma interação ou receberam algum feedback dos usuários, visto que essa regra vem implícita com a técnica de filtragem colaborativa. A Equação 4.2 representa formalmente o conjunto de itens. Cada $i \in I$ no sistema é representado por uma tupla com dois metadados $i = \langle UID, g \subseteq G \rangle$, onde da esquerda para a direita: UID representa a identificação única do item e $g \subseteq G$ o subconjunto de gêneros aos quais o item pertence. Um item i pode conter um ou mais gêneros, sendo cada gênero acessado como $g \in i$. A Tabela 4.3 é uma possível representação visual da modelagem.

$$I = \{i_1, ..., i_z | 1 \leq z \leq \mathbb{N}\} \tag{4.2}$$

Tabela 4.3 Oito itens e seus gêneros.

UID	$g \subseteq G$
I-001	Rock
I-002	Pop
I-003	Blues
I-004	Samba
I-005	Arrocha
I-006	Maracatu
I-007	MPB
I-008	Pagode

Fonte: Tabela elaborada pelo autor.

4.3.3 Modelo do Usuário

Em qualquer Sistema de Recomendação é necessário que os usuários realizem algum tipo de interação com os itens do sistema, essas interações são chamadas de transações (Seção 2.2) e são selecionadas para serem modeladas como feedback dos usuários (Seção

2.3). Assim, neste sistema o modelo do usuário MU_u representa todas as transações entre o usuário u e um subconjunto de itens pertencentes a I. Neste trabalho, cada transação pertencente ao modelo do usuário possui um peso/*rating*/feedback $w_{u,i} \in \mathbb{R}$. MU_u representa o modelo do usuário u e MU representa o conjunto com todos os modelos dos usuários. A Equação 4.3 representa formalmente o modelo do usuário, onde a tupla é formada por dois metadados: o item i e o peso da transação(feedback) $w_{u,i}$. A Tabela 4.4 é uma possível representação visual para o MU, que é composto por três MU_u, onde MU_{U-001} possui três itens em seu modelo.

$$MU_u = \{\langle i_1, w_{u,1}\rangle, ..., \langle i_z, w_{u,z}\rangle | 1 \leq z \leq |I|\} \tag{4.3}$$

Tabela 4.4 Três usuários e seus modelos.

Usuário UID	Item UID	w_{ui}
U-001	I-001	1
U-001	I-002	4
U-001	I-008	5
U-002	I-004	2
U-002	I-007	10
U-003	I-001	9
U-003	I-003	11
U-003	I-004	3

Fonte: Tabela elaborada pelo autor.

4.3.4 Modelo dos Itens Bloqueados

O modelo dos itens bloqueados por u (MIB_u) é composto por todos os itens $i \in I$ que não devem compor a lista de recomendação do usuário u. u pode dar algum retorno indicando que este item está bloqueado de ser recomendado, como o sistema pode identificar estes itens durante o processo de funcionamento. A Equação 4.4 apresenta a descrição formal do modelo. A Tabela 4.5 descreve uma possível representação visual para MIB, que é composto por três MIB_u, onde MIB_{U-001} possui dois itens no modelo.

$$(\forall u \in U) MIB_u = \{i_1, ..., i_z | 1 \leq z \leq |I|\} \tag{4.4}$$

4.3.5 Modelo dos Itens Desconhecidos

O modelo dos itens desconhecidos por u (MID_u) é composto por todos os itens $i \in I$ que não pertencem ao modelo do usuário MU_u nem ao modelo de itens bloqueados MIB_u, ou seja, $\forall u \in U$ o conjunto dos itens I é dividido em três partes: o modelo do usuário MU_u, modelo de itens bloqueados MIB_u e o modelo dos itens desconhecidos MID_u. A Equação 4.5 apresenta a descrição formal do modelo. A Tabela 4.6 apresenta uma possível representação visual para MID, que é composto por três MID_u, onde MID_{U-001} possui três itens em seu modelo.

Tabela 4.5 Três modelos de itens bloqueados.

Usuário UID	Item UID
U-001	I-010
U-001	I-022
U-002	I-014
U-002	I-027
U-003	I-021
U-003	I-013

Fonte: Tabela elaborada pelo autor.

$$(\forall u \in U) MID_u = (I - MU_u) - MIB_u \tag{4.5}$$

Tabela 4.6 Três modelos de itens desconhecidos.

Usuário UID	Item UID
U-001	I-071
U-001	I-082
U-001	I-018
U-002	I-124
U-002	I-087
U-003	I-011
U-003	I-033
U-003	I-084

Fonte: Tabela elaborada pelo autor.

4.3.6 Modelo dos Itens Candidatos Maximizado

Todos os modelos apresentados anteriormente são provenientes do pré-processamento, sendo o modelo dos itens candidatos maximizado MIC_u^{max} o único que acontece em outra etapa. Os algoritmos recomendadores normalmente têm como saída itens e um valor predito atribuído a cada item. O MIC_u^{max} pode ter tamanho igual ou menor que o modelo de itens desconhecidos MID_u. No MIC_u^{max} os itens são ordenados a partir do peso predito para cada item, que é atribuído pelo algoritmo. Entre a saída do recomendador e a modelagem em MIC_u^{max} o sistema pode limpar e filtrar os dados novamente, buscando qualquer inconsistência entre etapas ou aplicando metodologias de aprimoramento. A definição formal do modelo de itens candidatos maximizado é apresentada na Equação 4.6. A Tabela 4.7 apresenta uma possível representação visual do MIC^{max} que é composto por três MIC_u^{max}, onde MIC_{U-001}^{max} possui dois itens em seu modelo. Como exemplo, na Tabela 4.6 o MID_{U-001} possuía três itens, entretanto o MIC_{U-001}^{max} possui dois itens, isto é devido a uma possível filtragem nos itens entre a saída do recomendador e a modelagem para a entrada do pós-processamento.

$$(\forall u \in U) MIC_u^{max} = \{\langle i_1, \hat{w}_{r(u,1)}\rangle, ..., \langle i_m, \hat{w}_{r(u,m)}\rangle / 1 \leq m \leq |MID_u|\} \tag{4.6}$$

Tabela 4.7 Três modelos de itens candidatos maximizados.

Usuário UID	**Item UID**	$\hat{w}_{r(u,i)}$
U-001	I-071	5
U-001	I-082	4
U-002	I-124	5
U-002	I-087	4
U-003	I-011	8
U-003	I-033	6

Fonte: Tabela elaborada pelo autor.

4.4 ALGORITMO DE CALIBRAGEM PARA RECOMENDAÇÃO

A etapa de processamento recebe do pré-processamento os dados devidamente modelados e os utiliza como base para o treinamento e a predição/agrupamento nos algoritmos recomendadores. Os algoritmos usados são baseados na técnica de filtragem colaborativa, que usam os modelos de todos os usuários MU como base de treinamento, para que então possa predizer um possível peso/*rating* que os usuários possam vir a dar aos itens pertencentes aos seus modelos de itens desconhecidos $(\forall u \in U)MID_u$. As predições geradas como saída a partir dos algoritmos recomendadores são filtradas e modeladas para cada usuário como modelo de itens candidatos maximizados $(\forall u \in U)MIC_u^{max}$ e são usadas pelo pós-processamento para encontrar os top-N itens mais justos para compor a lista de recomendação LR_u.

O Algoritmo 1 ilustra o código referente às principais funções do sistema. No início o sistema carrega como entrada: os modelos de todos os usuários MU, o conjunto de itens I e os modelos de itens desconhecidos de todos os usuários MID. Como saída o sistema retorna todas as listas de recomendação LR. Na Linha 1 é iniciado um algoritmo recomendador com os dados de treinamento MU. Com o modelo de sistema proposto é possível utilizar qualquer um dos algoritmos recomendadores baseados em filtragem colaborativa (Seção 2.6). Na Seção 4.8 será debatido como escolher o melhor algoritmo recomendador a partir do protocolo proposto. A Linha 2 inicia um laço que é executado para todos os usuários. Dentro do laço na Linha 3, é utilizado o algoritmo recomendador e o MID_u para predizer o $\hat{w}_{r(u,i)}$ $\forall i \in MID_u$, obtendo assim o MIC_u^{max}. Na Linha 4 é realizado o pós-processamento (Seção 4.5), onde é passado como parâmetro o MU_u e o MIC_u^{max}. A Linha 5 concatena a lista de recomendação do usuário u com as listas já geradas até o momento. Assim, ao término do laço são retornadas todas as listas de recomendação LR.

O Algoritmo 2 é a representação do pós-processamento, visando buscar uma lista de itens justa para um usuário u. Como entrada de dados na calibragem têm-se: o modelo de um usuário MU_u e o modelo de itens candidatos deste usuário MIC_u^{max}. Como saída o algoritmo retorna uma lista de recomendação justa para o usuário LR_u. O Algoritmo 2 é o pseudo-código da função na Linha 4 do Algoritmo 1.

A Linha 2 inicia o algoritmo com o cálculo da distribuição dos gêneros sobre MU_u, que é apresentado na Seção 4.5.1. Na Linha 3 é calculado o peso do balanceamento entre

Algorithm 1: Iniciar do sistema.

Input: MU
Input: I
Input: MID
Output: LR

```
1 recomendador = AlgoritmoRecomendador(MU)
2 foreach u ∈ U do
3 |   MIC_u^max = predizer(recomendador, MID_u)
4 |   LR_u = posprocessamento(MU_u, MIC_u^max)
5 |   LR = concat(LR, LR_u)
6 end
```

Algorithm 2: Pseudo-código do pós-processamento.

Input: MU_u
Input: MIC_u^{max}
Output: LR_u

```
1  Function posprocessamento(MU_u, MIC_u^max):
2  |   p_u = DistAlvo(MU_u)
3  |   λ_u = PesoLambda(p_u)
4  |   for k ← 1 to N do
5  |   |   maiorUtilidade = −∞
6  |   |   melhorLista = ListaVazia()
7  |   |   foreach i, ŵ_r(u,i) ∈ (MIC_u^max − LR_u) do
8  |   |   |   lt = Copiar(LR_u)
9  |   |   |   lt = concat@k(lt, ⟨i, ŵ_r(u,i)⟩)
10 |   |   |   utilidade = FuncaoUtilidade(λ_u, p_u, lt)
11 |   |   |   if utilidade > maiorUtilidade then
12 |   |   |   |   melhorLista = lt
13 |   |   |   end
14 |   |   end
15 |   |   LR_u = melhorLista
16 |   end
17 |   return LR_u
```

relevância e divergência (Seção 4.5.4). Na Linha 4 é aplicado um laço que itera sobre cada posição da lista de recomendação $[1, N] \in \mathbb{N}$, usando o algoritmo guloso *Surrogate Submodular* (Seção 4.5.7) para escolher qual é o melhor item para a posição@k. Na Linha 5 é iniciada uma variável para guardar o maior valor de utilidade da lista, ou seja, o valor da melhor lista até a posição@k. Na Linha 6 é criada uma lista vazia, onde são guardados os itens da melhor lista até a posição@k. Na Linha 7 um novo laço é iniciado para buscar o melhor item no MIC_u^{max} para a posição@k. Na Linha 8 é copiada a lista de recomendação até a posição@k-1. Na Linha 9 adiciona-se um item com o peso/*rating*

predito $\langle i, w_{u,i} \rangle \in MIC_u^{max}$ à posição@k de uma lista de recomendação temporária lt. Na Linha 10 são passadas como parâmetros o peso do balanceamento λ_u, a distribuição alvo p_u e a lista de recomendação temporária lt para encontrar o valor da utilidade da lista para o usuário u. Nas Linhas 11 e 12 é verificado se a utilidade da lista temporária é maior do que a melhor lista previamente encontrada para a mesma posição, caso verdadeiro, é atribuída a lista temporária como a melhor lista até o momento, caso contrário é mantida a lista anterior. Ao término do laço na Linha 15 é atribuída a melhor lista até a posição@k para a lista de recomendação do usuário LR_u, a qual é retornada ao término da função *posprocessamento*.

O Algoritmo 3 é a representação da função que calcula a utilidade da lista temporária de recomendação. Esta função recebe: o peso do balanceamento λ_u, a distribuição dos gêneros p_u e uma lista de recomendação temporária lt; e tem como retorno o valor da utilidade desta lista. Na Linha 2 é calculada a distribuição dos gêneros q_u presentes na lista de recomendação temporária. Na Linha 3 é computado o valor da divergência entre as distribuições p_u e q_u (Seção 4.5.2). Na Linha 4 é encontrado o valor da relevância dos itens da lista de recomendação temporária (Seção 4.5.3). Na Linha 5 é realizado o balanceamento entre a relevância e a divergência. Ao término da função é retornado o valor da utilidade de lt.

Algorithm 3: Pseudo-código da função de utilidade.

Input: λ_u
Input: p_u
Input: lt
Output: utilidade
1 **Function** `FuncaoUtilidade`(λ_u, p_u, lt)`:`
2 q_u = DistReal(lt)
3 vDiv = MedDiv(p_u, q_u)
4 vRel = Ranque(lt)
5 utilidade = Balanc(λ_u, vRel, vDiv)
6 **return** utilidade

4.5 PÓS-PROCESSAMENTO

A etapa de pós-processamento utiliza o modelo de itens candidatos maximizados MIC_u^{max} para encontrar uma lista de recomendação mais justa. Essa lista busca uma calibragem entre relevância e divergência, visando respeitar a proporcionalidade dos gêneros nas preferências $\forall u \in U$. A etapa é representada no Algoritmo 1 na Linha 4 e melhor detalhada nos Algoritmos 2 e 3.

4.5.1 Distribuição de gêneros

Os gêneros pertencentes aos modelos do usuário e dos itens candidatos maximizados podem ser representados cada um como uma distribuição. Assim, para encontrar a distri-

buição alvo, que é baseada no modelo do usuário, e a distribuição realizada, que é baseada na lista de recomendação, são utilizadas as funções p formalizada na Equação 4.8 e q formalmente apresentada na Equação 4.9. Estas formulações também são utilizadas por todos os trabalhos relacionados (STECK, 2018; KAYA; BRIDGE, 2019; ABDOLLAHPOURI et al., 2019a, 2020; LIN et al., 2020). p e q são definidas abaixo:

- **p(g|i):** função para identificar o valor da importância do gênero g no item i. Assim, nesta dissertação é definida como a probabilidade de escolher um gênero em um item, i. e., uma divisão onde o numerador é 1 e o denominador é o número de gêneros que o item possui ($|genresIn(i)|$). Como apresentado na Equação 4.7 abaixo:

$$p(g|i) = \frac{1}{|genresIn(i)|} \tag{4.7}$$

- **p(g|u):** realiza o cálculo da distribuição dos gêneros no modelo do usuário MU_u. Para um gênero g em um usuário u o valor da distribuição desse gênero é dado como divisão, na qual: o numerador é o somatório das multiplicações do peso $w_{u,i}$ com o valor de um gênero g no item i ($\sum_i^{MU_u} w_{u,i} \cdot p(g|i)$); $w_{u,i}$ é o peso da interação do usuário u com o item i; e o denominador é dado pelo somatório dos pesos $w_{u,i}$; $\mathbb{1}(g \in i)$ indica que se o gênero g está presente no item i, o retorno é 1, caso contrário é 0.

$$DistAlvo(MU_u) = (\forall g \in MU_u)\ p(g|u) = \frac{\sum_i^{MU_u} \mathbb{1}(g \in i) w_{u,i} \cdot p(g|i)}{\sum_i^{MU_u} \mathbb{1}(g \in i) w_{u,i}} \tag{4.8}$$

  Steck (2018) e Kaya e Bridge (2019) utilizam no peso $w_{u,i}$ valores 0 (não gostei) ou 1 (gostei), isto é, feedback implícito binário. Ambos os trabalhos forçam uma transformação das bases de dados em feedback binário. Assim, uma das diferenças entre este trabalho e o estado-da-arte é a exploração de um sistema que trabalha com um peso $w_{u,i}$ qualquer provindo de uma interação entre o usuário e o item.

- **q(g|u):** realiza o cálculo da distribuição dos gêneros sobre a lista de recomendação LR_u. Ao observar as Equações 4.8 e 4.9 percebe-se que são similares com a diferença de seus contextos e valores. O valor do peso $\hat{w}_{r(u,i)}$ pode ser dado pela posição do item na lista de recomendação, para isso métricas de ranque como Mean Reciprocal Rank (MRR) ou normalized Discounted Cumulative Gain (nDCG) podem ser usadas, assim como o valor predito do feedback.

$$DistReal(LR_u) = (\forall g \in LR_u)\ q(g|u) = \frac{\sum_i^{LR_u} \mathbb{1}(g \in i) \hat{w}_{r(u,i)} \cdot p(g|i)}{\sum_i^{LR_u} \mathbb{1}(g \in i) \hat{w}_{r(u,i)}} \tag{4.9}$$

  Steck (2018) e Kaya e Bridge (2019) utilizam a posição do ranque. Assim, uma das diferenças entre este trabalho e o estado-da-arte é a exploração de um sistema que trabalha com um peso $\hat{w}_{r(u,i)}$ predito pelo algoritmo recomendador.

- $\tilde{q}(g|u)$: existe a possibilidade do valor de $q(g|u)$ dar zero, assim para evitar uma divisão por zero realiza-se um balanceamento entre as probabilidades. O valor do α deve ser pequeno para que $q \approx \tilde{q}$ e para preservar o resultado original. Esta formulação também é utilizada pelo estado-da-arte.

$$DistReal(LR_u) = (\forall g \in LR_u)\ \tilde{q}(g|u) = (1-\alpha) \cdot q(g|u) + \alpha \cdot p(g|u) \qquad (4.10)$$

A função *DistAlvo* utilizada no Algoritmo 2 na Linha 2, recebe o MU_u como parâmetro e retorna uma distribuição contendo um valor para cada gênero $g \in MU_u$. O mesmo se aplica à função *DistReal* utilizada no Algoritmo 3 na Linha 2, que recebe a LR_u como parâmetro e retorna uma distribuição contendo um valor para cada gênero $g \in LR_u$.

Tabela 4.8 Um exemplo da probabilidade dos gêneros em cada item e seu peso.

Item	$w_{u,i}$	**Distribuição p(g|i)**
O Auto da Compadecida	5	$Aventura = 1; p(g\|i) = \frac{1}{2} = 0,5$ $Comedia = 1; p(g\|i) = \frac{1}{2} = 0,5$
Amor?	4	$Drama = 1; p(g\|i) = \frac{1}{1} = 1$
Deus e o Diabo na Terra do Sol	4	$Aventura = 1; p(g\|i) = \frac{1}{4} = 0,25$ $Crime = 1; p(g\|i) = \frac{1}{4} = 0,25$ $Drama = 1; p(g\|i) = \frac{1}{4} = 0,25$ $Faroeste = 1; p(g\|i) = \frac{1}{4} = 0,25$

Fonte: Tabela elaborada pelo autor.

Um exemplo do cálculo da distribuição de probabilidades está representado nas Tabelas 4.8 e 4.9, que por sua vez seguem a formulação da Equação 4.8. Para cada gênero g em um item i é atribuído o valor 1 dividido pelo número de gêneros que o item possui ($\frac{1}{|\#g \in i|}$). Na Tabela 4.8 tem-se na primeira linha o item "O Auto da Compadecida", que possui um peso de valor 5 (neste exemplo, 5 é referente à nota dada pelo usuário) com 2 gêneros e seus respectivos valores da probabilidade $p(g|i)$. Na Tabela 4.9 têm-se os gêneros na primeira coluna seguida de três colunas de itens e seus valores da multiplicação do peso com a probabilidade do gênero $w_{u,i} \cdot p(g|i)$ de cada item. Por fim na última coluna o valor de $p(g|u)$ é apresentado.

A distribuição realizada $q(g|u)$ é dada sobre a lista de recomendação do usuário LR_u. Os resultados seguem a formulação da Equação 4.9, em alguns casos, a Equação 4.10 pode ser utilizada. O peso $\hat{w}_{r(u,i)}$ neste exemplo é dado pelo valor predito pelo algoritmo recomendador (por exemplo, uma possível nota).

A Tabela 4.10 apresenta uma distribuição alvo de um usuário u representada na coluna p e uma distribuição realizada na coluna q e $\tilde{q}(g|u)$.

4.5.2 Medidas de divergência

Distância é definida como um grau do quão longe dois objetos estão. Existem inúmeras medidas de distância. Medidas que satisfazem as propriedades métricas são simplesmente

Tabela 4.9 Cálculo da importância de cada gênero g para o usuário u.

Gênero	O Auto da Compadecida	Amor?	Deus e o Diabo na Terra do Sol	p(g|u)
Aventura	5·0,5	4·0	4·0,25	3,5/9=0,388
Comédia	5·0,5	4·0	4·0	2,5/5=0,5
Crime	5·0	4·0	4·0,25	1/4=0,25
Drama	5·0	4·1	4·0,25	5/8=0,625
Faroeste	5·0	4·0	4·0,25	1/4=0,25

Fonte: Tabela elaborada pelo autor.

Tabela 4.10 Exemplo do cálculo de distribuição.

Gênero	p	q	$\tilde{q}(g\|u)$
Aventura	0,388	0,35	0,35038
Comédia	0,5	0,563	0,56237
Crime	0,25	0,4	0,3935
Drama	0,625	0,5	0,50125
Romance	0,0	0,0	0
Sci-fi	0,0	0,0	0
Faroeste	0,25	0,0	0,0025

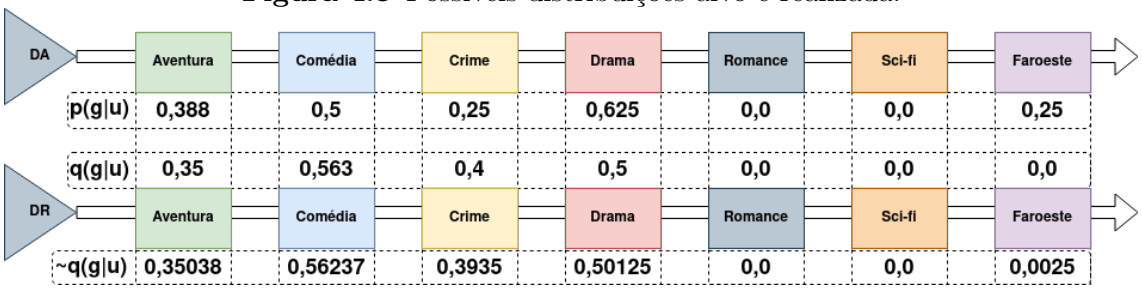
Fonte: Tabela elaborada pelo autor.

chamadas de métricas (ex. $[0,0;...;1,0]$). Ocasionalmente medidas que não satisfazem essas propriedades métricas são chamadas de divergência (CHA, 2007).

No Algoritmo 3 a Linha 3 é a representação da utilização da função no sistema. A função $MedDiv$ recebe duas distribuições p_u e q_u, e retorna um valor da divergência entre as duas distribuições de probabilidades dos gêneros. Nesta função qualquer medida que calcule o valor da divergência entre as duas distribuições pode ser utilizada. Em seu trabalho Cha (2007) apresenta mais de 45 medidas classificadas em grupos/famílias, entre as quais, nesta dissertação, são utilizadas três medidas de diferentes famílias: Kullback-Leibler (KL), representado como $F_{KL}(p_u, q_u)$ pertencente à família de Shannon; Hellinger, representado como $F_{HE}(p_u, q_u)$ pertencente à família Fidelity; e o Pearson Chi-Square, representado como $F_{PC}(p_u, q_u)$ pertencente à família χ^2. O estado-da-arte (STECK, 2018; KAYA; BRIDGE, 2019) utiliza o KL e os trabalhos relacionados de Abdollahpouri et al. (2019a, 2020) e Lin et al. (2020) implementam o Hellinger, assim, tomando como base os trabalhos anteriores o modelo de sistema proposto pode utilizar o KL e o Hellinger e como contribuição é implementado o Pearson Chi-Square. Durante a Seção 4.8 será debatido como escolher a melhor medida de divergência em combinação com os outros componentes do sistema. A Figura 4.3 apresenta duas distribuições, estas que podem ser utilizadas para calcular a divergência com as medidas apresentadas a seguir.

Na Seção 1.4 foi apresentado a questão de pesquisa que trata sobre a possibilidade da medida de divergência utilizada influenciar na criação das listas de recomendação. Assim, a seguir é apresentado três medidas que serão utilizadas no modelo de sistema proposto.

Figura 4.3 Possíveis distribuições alvo e realizada.



Fonte: Figura elaborada pelo autor.

4.5.2.1 Kullback-Leibler Proposto por Kullback e Leibler (1951) é uma medida de divergência entre dois conjuntos e pode assumir valores entre $[0, \infty] \in \mathbb{R}$, sendo que quanto menor o valor menos divergentes os conjuntos são (JOLAD et al., 2016). Steck (2018) lista os principais motivos de usar o KL para calibrar: 1) dá zero se as distribuições forem iguais: $p(g|u) = \tilde{q}(g|u)$; 2) é sensível a variações discrepantes; 3) é uniforme e consequentemente permite distribuições sem discrepância.

$$MedDiv(p_u, q_u) = F_{KL}(p_u, q_u) = \sum_{g}^{G} p(g|u) \log_2 \frac{p(g|u)}{\tilde{q}(g|u)} \tag{4.11}$$

4.5.2.2 Hellinger Proposto por Hellinger (1909) é uma medida de divergência entre dois conjuntos. No qual, $[0, \infty] \in \mathbb{R}$ é o valor da divergência entre os conjuntos de probabilidades p_u e q_u. A divergência Hellinger é relacionada ao coeficiente de *Bhattacharyya* (CHA, 2007; JOLAD et al., 2016), entretanto possuem diferenças em suas formulações.

$$MedDiv(p_u, q_u) = F_{HE}(p, q) = \frac{\sqrt{\sum_{g}^{G} (\sqrt{p(g|u)} - \sqrt{q(g|u)})^2}}{\sqrt{2}} \tag{4.12}$$

4.5.2.3 Pearson Chi-Square Pode assumir valores $[0, \infty] \in \mathbb{R}$. Se as distribuições de probabilidades p_u e q_u forem idênticas, o valor resultante é 0 e quanto maior a divergência entre os dois conjuntos o valor vai tendendo a infinito (CHA, 2007). É necessário realizar uma adaptação na formulação do χ^2, devido à possibilidade de $q(g|u)$ assumir valor 0, para isso é utilizado $\tilde{q}(g|u)$.

$$MedDiv(p_u, q_u) = F_{PC}(p_u, q_u) = \sum_{g}^{G} \frac{(p(g|u) - \tilde{q}(g|u))^2}{\tilde{q}(g|u)} \tag{4.13}$$

4.5.3 Medida de relevância

As listas de recomendação possuem uma ordem dos itens a serem apresentados aos usuários. Normalmente quanto maior a utilidade do item mais ao topo da lista ele aparece. Neste trabalho é implementada a técnica de filtragem colaborativa, na qual leva-se

em consideração o feedback do usuário, que é utilizado para prever o valor $\forall i \in MID$. Assim, para cada item pertencente à lista são somados os valores preditos $\hat{w}_{r(u,i)}$, como apresentado na Equação 4.14. No Algoritmo 3, a Linha 4 representa a chamada para o cálculo do ranque da lista, recebendo-a e retornando o valor do ranque desta. Os trabalhos do estado-da-arte utilizam a posição do item na lista como o peso $\hat{w}_{r(u,i)}$, isto acontece devido a considerarem o sistema apenas para classificação.

$$Ranque(LR_u) = \sum_{i}^{LR_u} \hat{w}_{r(u,i)} \tag{4.14}$$

A Tabela 4.11 demonstra uma possível lista de recomendação onde o valor da relevância do ranque calculado utilizando a equação acima é de $Ranque(LR_u) = 22,5$.

Tabela 4.11 Uma possível lista de recomendação.

Filmes	$\hat{w}_{r(u,i)}$
Star Wars IV	5
Interstellar	4,5
2001: A Space Odyssey	4,5
Star Wars V	4,5
Gravity	4
	22,5

Fonte: Tabela elaborada pelo autor.

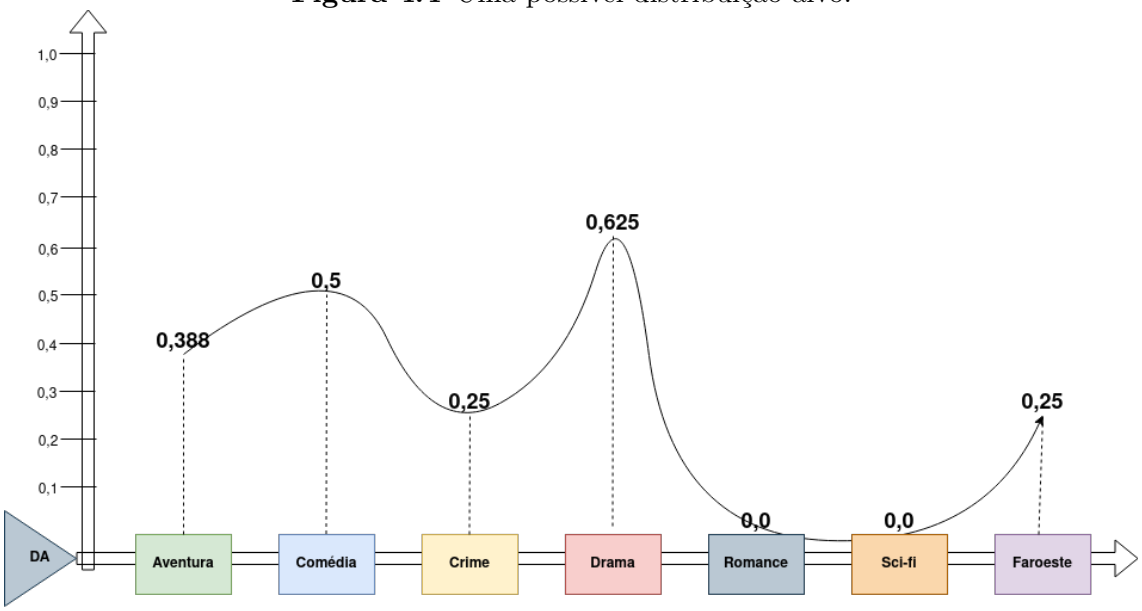
4.5.4 Peso do balanceamento relevância-divergência

As formulações do balanceamento normalmente utilizam-se de um peso λ para balancear a relevância do ranque com as medidas de divergência. No estado-da-arte dos trabalhos de Steck (2018) e Kaya e Bridge (2019) o λ é atribuído de forma manual e não personalizada, onde a proporção é dada diretamente pelo especialista do sistema. Isto faz com que o sistema de recomendação não respeite de forma devida as preferências de usuário perpetuando assim um enviesamento. Três são os principais problemas que a atribuição manual e não personalizada pode gerar: 1) todos os usuários recebem itens mais relevantes quando o valor do λ é pequeno, ignorando assim os gêneros menos representativos que formam as preferências do usuário e não respeitando os usuários que desejam uma lista de recomendação mais calibrada; 2) quando o λ é grande o oposto acontece, a calibragem predomina e todos os usuários recebem uma lista de recomendação calibrada, não respeitando assim os usuários que desejam receber recomendações com foco em relevância; 3) quando o valor de λ é 0,5 o balanceamento trata de forma igual relevância e calibragem, não respeitando assim tanto os usuários que preferem foco na relevância quanto os que preferem calibragem.

Esta dissertação propõe como contribuição duas formulações para encontrar o valor de λ de forma personalizada. Nenhum dos trabalhos do estado-da-arte explora formulações para o peso do balanceamento λ. As formulações contidas a seguir são as representações

da Linha 3 no Algoritmo 2. A Figura 4.4 contém uma possível distribuição alvo que é utilizada para calcular o valor do λ.

Figura 4.4 Uma possível distribuição alvo.



Fonte: Figura elaborada pelo autor.

Na Seção 1.4 foi apresentado a questão de pesquisa que trata sobre a possibilidade de utilizar pesos personalizados do balanceamento para obter melhorias ou manter desempenho quando comparados com pesos constantes. Assim, a seguir é apresentado duas formas de encontrar o peso personalizado.

4.5.4.1 Variância normalizada A variância é uma medida de dispersão nos valores possíveis de uma variável aleatória X. Nesta dissertação X é dado como um usuário, sendo o valor encontrado referente à dispersão dos gêneros em suas preferências. A variância normalizada obtém valores entre $[0, 1] \in \mathbb{R}$ e quanto maior o valor, maior será a distância entre os pontos a partir da média.

As Equações 4.15 e 4.16 são as representações formais, onde a Equação 4.15 é a média dos valores da distribuição de cada gênero $g \subset G$ pertencentes ao MU_u. A média desses valores é utilizada na Equação 4.16 como um valor central para o cálculo de distância entre os pontos e o valor central.

$$mp(p_u) = \frac{\sum_g^G p(g|u)}{|G|} \tag{4.15}$$

$$var(p_u) = \frac{\sum_g^G |p(g|u) - mp(p_u)|^2}{|G|} \tag{4.16}$$

O valor obtido da função $var(p_u)$ significa que quanto menor o valor, menor é a distância entre os pontos da distribuição dos gêneros p_u, o que indica que o usuário

u possui uma quantidade maior de gêneros. Entretanto nos modelos de recomendação quanto maior o λ, mais justa a lista de recomendação será, assim é necessário realizar uma inversão nos valores normalizados (Equação 4.17).

$$\lambda_u = PesoLambda(p_u) = 1 - var(p_u) \tag{4.17}$$

Ao utilizar a Figura 4.4 com as formulações apresentadas acima são obtidos os valores de $mp(p_u) = 0,287571429$, $var(p_u) = 0,337287714/7 = 0,048184$ e $\lambda_u = 1 - 0,018033102 = 0,951816041$. Assim, o usuário do exemplo receberá recomendações com maior foco nos gêneros.

4.5.4.2 Contagem de gêneros O sistema trabalha com um conjunto de gêneros G, pertencentes aos itens $i \in I$. Com isso, é possível extrair um subconjunto de G a partir de MU_u. Essa ideia permite usar uma outra forma para encontrar o λ_u, como é apresentado na Equação 4.19, onde para $u \in U$ conta-se quantos gêneros este usuário possui e divide-se pelo número de gêneros no sistema.

$$possui(u, g) = \begin{cases} 0, \ if p(g|u)=0 \\ \\ 1, outros \ casos \end{cases} \tag{4.18}$$

$$\lambda_u = PesoLambda(p_u) = \frac{\sum_g^G possui(u, \ g)}{|G|} \tag{4.19}$$

Ao utilizar a Figura 4.4 com as formulações apresentadas acima é obtido o valor de $\lambda_u = 5/7 = 0,714285714$. Assim, o usuário do exemplo receberá recomendações com maior foco nos gêneros.

4.5.5 Balanceamento relevância-divergência

Neste trabalho são utilizadas duas formulações de balanceamento para calibragem. A função $Balanc$ recebe o λ_u, o valor da relevância do ranque e o valor da divergência, e retorna um valor ligado à utilidade daquele balanceamento. No Algoritmo 3 a Linha 5 representa a chamada para o balanceamento no sistema, onde qualquer um dos balanceamentos propostos a seguir podem ser usados. O valor da utilidade é utilizado por algum algoritmo de ranqueamento para criar a lista de recomendação LR_u, ranqueando assim os itens com o maior valor de utilidade no topo da lista.

4.5.5.1 Balanceamento Linear Steck (2018) apresenta um balanceamento linear, este que nivela o peso proporcional do ranque e da divergência. A Equação 4.20 é a representação formal do balanceamento. A função é dividida em duas partes: a primeira é a multiplicação do $1-\lambda_u$ pelo valor da relevância do ranque, a segunda é a multiplicação do λ_u pelo valor da divergência. Assim, quanto maior a divergência, maior é a subtração no valor do ranque. Este balanceamento também é utilizado pelo estudo de Kaya e Bridge (2019). Ao final o valor da utilidade linear $uLin$ é retornada.

$$Balanc(\lambda_u, vRel, vDiv) = uLin = (1 - \lambda_u) \cdot vRel - \lambda_u \cdot vDiv \tag{4.20}$$

4.5.5.2 Balanceamento Logarítmico A calibragem se utiliza de gêneros/classes para buscar a concisão da lista de recomendação com as preferências do usuário, escolhendo o item a partir do efeito que causa nesta concisão, mas ainda utilizando somente o gênero como guia das recomendações. Este comportamento está presente no balanceamento linear apresentado na Equação 4.20. Uma forma de considerar os gêneros, os itens e o usuário é adicionar o uso do viés. Assim, como contribuição, esta dissertação propõe um novo balanceamento. Este é dado de forma logarítmica para suavizar a curva, além de contar com a adição de um viés do usuário. A inserção do viés permite ao pós-processamento em calibragem adaptar a lista de recomendação de acordo com os itens inseridos na lista, ou seja, além de considerar os gêneros na calibragem também são considerados os itens que compõem a lista.

A Equação 4.21 é dada em quatro partes: i) segue a formulação do balanceamento linear, que subtrai a divergência do ranque e retorna o valor deste cálculo ($uLin$); ii) é a aplicação de um módulo com soma +1 seguido do logaritmo sobre o valor $|uLin| + 1$; iii) é a aplicação da função $sign$ sobre o valor $uLin$; e iv) é o cálculo do viés do usuário $\hat{b}_u$. O valor de $uLin$ pode ser negativo, devido a isso a aplicação do módulo e a soma do +1 é para evitar o log zero quando $uLin$ for 0. O módulo dado em ii) altera o sinal, assim, para regularizar a situação, a função $sign$ retorna -1 se $uLin$ for negativo, 0 se $uLin$ for 0 e 1 se $uLin$ for positivo.

$$Balanc(\lambda_u, vRel, vDiv) = sign(uLin) \cdot \log^{|uLin|+1} + \hat{b}_u \tag{4.21}$$

O $\hat{b}_u$ é uma variante do viés do usuário descrito na Equação 2.14. Nesta proposta de modificação é usada a lista de recomendação do usuário LR_u em substituição às preferências, que neste caso é modelada como MU_u, assim o valor do feedback w_{ui} é trocado pelo valor predito pelo algoritmo recomendador $\hat{w}_{ui}$. O cálculo de b_i é apresentado na Equação 2.13 como o viés do item. É esperado que a adição do viés do usuário na etapa de pós-processamento aprimore a recomendação principalmente em recomendadores que não possuem em sua formulação o viés (ex. KNN, tradicionais, etc). Outro detalhe é que na formulação linear quando o λ é 0 somente o ranque é considerado na hora de gerar a lista de recomendação, já a formulação logarítmica do balanceamento influi na geração da lista mesmo com λ sendo 0. Isto acontece devido ao viés do usuário $\hat{b}_u$ não possuir parâmetro regulador.

$$\hat{b}_u = \frac{\sum_{i \in LR_u} (\hat{w}_{r(u,i)} - \mu - b_i)}{\sigma + |LR_u|} \tag{4.22}$$

Na Seção 1.4 foi apresentada a questão de pesquisa que trata sobre a possibilidade de que, ao considerar o viés do usuário na formulação do balanceamento, é possível obter melhoria no desempenho. Assim, foi apresentada a formulação que utiliza o viés. Este que será testado durante o Capítulo 5.

4.5.6 Maximum Marginal Relevance

O Maximum Marginal Relevance (MMR) (CARBONELL; GOLDSTEIN, 1998) é uma métrica comumente utilizada para maximizar a escolha dos itens na lista de recomendação. O MRR tende a capturar o conjunto de itens ótimos para compor a lista de recomendação. O estado-da-arte, Steck (2018) e Kaya e Bridge (2019) fazem uso do MMR no contexto de recomendações calibradas. A Equação 4.23 apresenta formalmente a formulação usada no sistema ($\forall u \in U$).

$$LR_u = \underset{lt \subset MIC_u^{max}, |lt|=N}{\arg\max} Balanc(\lambda_u, Ranque(lt), MedDiv(p_u, q(lt))) \tag{4.23}$$

No qual, a função de balanceamento (linear ou logarítmico) recebe três parâmetros: 1) o valor do λ_u dado nas Equações 4.17 e 4.19, sendo representado na Linha 3 no Algoritmo 2; 2) o valor da relevância do ranque sobre uma lista temporária lt dado na Equação 4.14, sendo representado no Algoritmo 3 na Linha 4; e 3) o valor da medida de divergência dado nas Equações 4.11, 4.12 e 4.13, sendo representado no Algoritmo 3 na Linha 3.

No funcionamento da Equação 4.23 o $lt \subset MIC_u^{max}, |lt| = N$ significa que uma lista de recomendação temporária lt é gerada como um subconjunto dos itens que constituem o modelo de itens candidatos maximizados MIC_u^{max}. A lista de recomendação temporária lt adiciona um item que maximize (arg max) o valor da utilidade até lt ter tamanho igual a N, i. e., os top-N melhores itens para u. Ao final, uma lista de recomendação para o usuário é retornada como LR_u.

4.5.7 Algoritmo de seleção

A lista de recomendação LR_u é composta pelos top-N itens que a tornem o mais justa possível, i.e., pelos itens que representados como uma distribuição dos gêneros obtenham a menor divergência com a distribuição baseada no modelo do usuário MU_u. Estes itens são selecionados a partir do modelo de itens candidatos maximizado MIC_u^{max}. A escolha dos itens que compõem a lista de recomendação calibrada é realizada por um algoritmo de seleção/reordenação que utiliza o valor da utilidade computado pelo MMR (seção anterior) para escolher item a item.

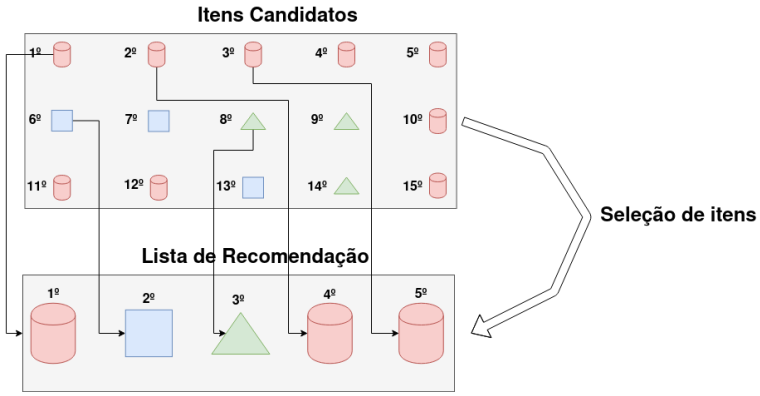
O contexto de recomendações calibradas tende a colocar os itens com maior utilidade no topo, itens estes que sejam os mais relevantes e possuam gêneros concisos com as preferências dos usuários, levando assim a uma dependência da posição em que o item aparece. Assim, a lista de recomendação calibrada é um arranjo de itens retirados de um conjunto base. Para selecionar estes itens, é necessário testar todos os m itens candidatos nas n posições, sendo assim um problema de arranjo sem repetição que pertence à classe NP-difícil. A Equação 4.24 apresenta a formulação de arranjo, o qual define a quantidade de listas testadas.

$$A_{m,n} = \frac{m!}{(m-n)!} \tag{4.24}$$

A Figura 4.5 demonstra um exemplo de seleção de itens onde um conjunto de $m = |MIC_u^{max}| = 15$ itens candidatos e $n = |LR_u| = 5$ posições na lista de recomendação.

Neste exemplo, seria necessário testar 360360 possíveis arranjos para, por fim, selecionar a melhor lista, o que se torna inviável quando o número de itens para seleção e a quantidade de posições aumenta, por exemplo, $|MIC_u^{max}| = 21000$ e $|LR_u| = 30$.

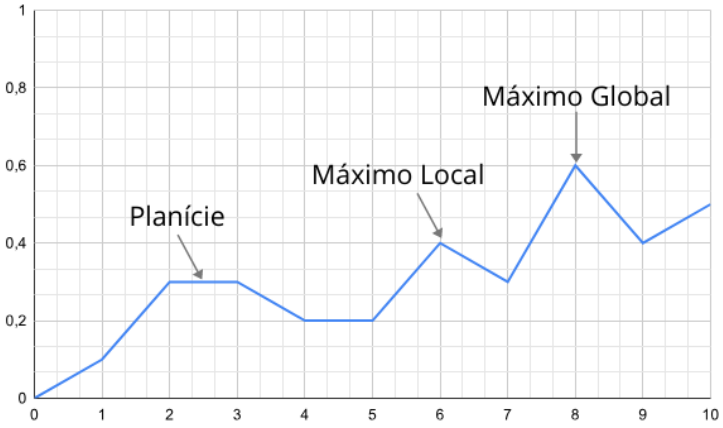
Figura 4.5 Exemplo de seleção/ordenação de itens.



Fonte: Figura elaborada pelo autor.

Existem diversas formas de resolver problemas de otimização combinatória/arranjo, mas nenhum em tempo satisfatório. Assim, como demonstrado na Figura 4.6, é necessário considerar outros valores além do melhor valor (máximo global) em troca de resultado em tempo satisfatório. A busca é ampliada mantendo a garantia de que o valor encontrado será o melhor valor possível (máximo local), este que pode ser alcançado a partir do uso de algoritmos gulosos. Em situação de planície alguns algoritmos ativam regras para alterar a situação, alguns outros irão considerar que a planície é o máximo local, só havendo modificação caso venha a existir um melhor resultado.

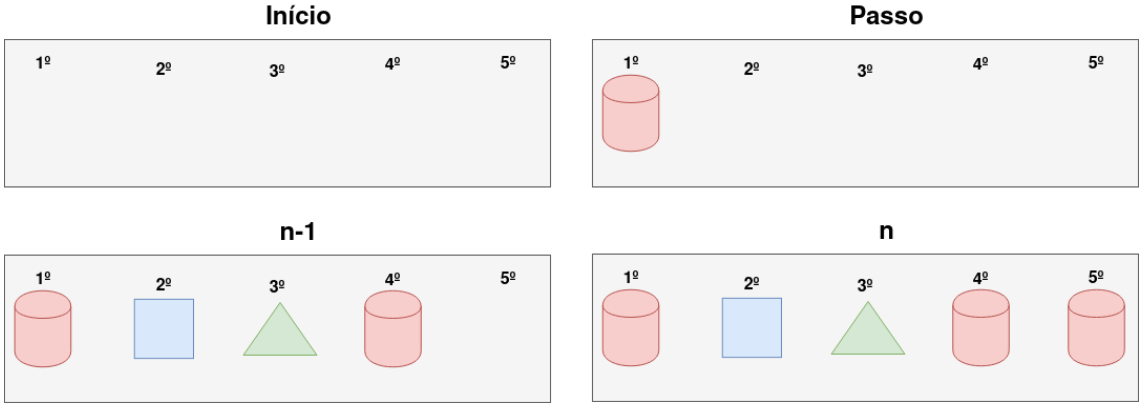
Figura 4.6 Espaço de possíveis estados pela função objetivo.



Fonte: Figura elaborada pelo autor.

O Surrogate Submodular proposto por Nemhauser, Wolsey e Fisher (1978), encontra uma lista justa com garantia de otimalidade em $1 - \frac{1}{e}$, onde o valor "e" é o número de

Figura 4.7 Funcionamento do *Surrogate*.



Fonte: Figura elaborada pelo autor.

Euler. Steck (2018) e Kaya e Bridge (2019) usam o *Surrogate* em seus estudos para realizar a reordenação dos itens, assim, seguindo a literatura, também é utilizado o *Surrogate* neste trabalho. No Algoritmo 2 o laço que começa na Linha 4 e vai até a Linha 17 é a representação da implementação do ranqueamento no sistema. Abaixo encontram-se descritos os passos de funcionamento da seleção:

- lt inicia com uma lista de recomendação vazia {};
- Para cada posição da lista de recomendação $k \leq n$ um item i é selecionado a partir do MIC_u^{max}, caso seja o argumento de maior valor entre LR_u e MU_u;
- No final a lista de recomendação possui $n - 1$ itens e o último item é adicionado obtendo uma lista de recomendação mais justa.

A Figura 4.7 representa os passos descritos acima e demonstra como são escolhidos dentre os itens candidatos os itens que compõem a lista de recomendação (Figura 4.5).

4.6 MÉTRICAS PARA AVALIAR CALIBRAGEM

O contexto de calibragem é sobre ir além da precisão, fazendo-se necessárias novas métricas específicas para o contexto. Todos os trabalhos relacionados (STECK, 2018; KAYA; BRIDGE, 2019; ABDOLLAHPOURI et al., 2019a, 2020; LIN et al., 2020) avaliam os resultados a partir de uma métrica chamada *Miscalibration*, que retorna o grau de descalibragem entre as preferências do usuário e a lista de recomendação. A métrica foi primeiro apresentada por Steck (2018). Entretanto, a formulação da *miscalibration* não consegue comparar resultados provindos de duas ou mais medidas de divergência. Além disso, a *miscalibration* não considera avaliar a evolução dos itens em cada posição da lista de recomendação. Assim, nesta dissertação são propostas duas novas métricas de avaliação para o contexto de recomendações calibradas.

4.6.1 Mean Average Calibration Error

A primeira métrica proposta, o Mean Average Calibration Error (MACE), é uma métrica de avaliação para entender o grau de erro na calibragem em cada lista de recomendação. A métrica calcula a diferença entre as distribuições p_u e q_u de cada usuário. O intervalo de resultados possíveis está entre $[0, 1] \in \mathbb{R}$.

$$CE(u, p, q) = \frac{\sum_{g \in G} |p(g|u) - q(g|u)|}{|G|} \tag{4.25}$$

$$ACE(u) = \frac{\sum_{j=1}^{N} CE(u, p_u, q(LR_u@j))}{N} \tag{4.26}$$

$$MACE = \frac{\sum_{u \in U} ACE(u)}{|U|} \tag{4.27}$$

A Equação 4.27 mostra o cálculo do erro da calibragem para um usuário. Para cada gênero calcula-se a diferença absoluta entre as duas distribuições ($\sum_{g \in G} |p(g|u) - q(g|u)|$) e no final uma normalização é retornada. A Equação 4.26 é uma interação sobre cada posição da lista de recomendação ($LR_u@j$) de $[1..N] \in \mathbb{N}$, onde N é o tamanho da lista, j é o tamanho em avaliação ($1 \leq j \leq N$). A distribuição alvo p_u é previamente calculada e a distribuição realizada q atua com todas as posições da lista, computando o erro progressivamente. Ao final uma média é realizada e a média do erro da calibragem é retornada. De forma similar, a Equação 4.27 demonstra o cálculo que obtém a média do valor aplicado para cada usuário. Valores baixos indicam uma melhor calibragem. Esta métrica nos ajuda a entender se a medida de divergência utilizada provê uma lista justa e também permite comparar os valores entre medidas.

Quando o MACE é utilizado é necessário atentar-se que em alguns casos o erro absoluto da calibragem será alto, devido aos gêneros de maior dominância nas preferências causarem uma discrepância. O motivo é entendível, desde que a calibragem gera uma lista de recomendação com mais gêneros se o usuário deseja assim, isto faz com que os gêneros menos representativos ganhem espaço na lista. Devido a este fato, o gênero mais representativo terá menos espaço na lista, causando assim uma discrepância quando se analisa o absoluto. Algumas medidas de divergência consideram discrepância em sua formulação, por exemplo, o KL.

4.6.2 Mean Rank MisCalibration

Steck (2018) e Abdollahpouri et al. (2019a, 2020) utilizam a métrica *MisCalibration* para computar o grau de calibragem/descalibragem da lista de recomendação com as preferências do usuário. Nesta dissertação, esta métrica é modificada, possibilitando novas interpretações, visando obter valores normalizados $[0, 1] \in \mathbb{R}$ que, assim, independentemente da medida de divergência usada possam criar meios de comparação entre si. A novo métrica também permite observar a lista conforme vai sendo criada. Assim, na formulação final é obtida a Mean Rank Miss Calibration (MRMC).

$$MC(p,q) = \frac{MedDiv(p,q)}{MedDiv(p,q(\{\}))} \tag{4.28}$$

$$RMC(u) = \frac{\sum_{j=1}^{N} MC(p, q(LR_u@j))}{N} \tag{4.29}$$

$$MRMC = \frac{\sum_{u \in U} RMC(u)}{|U|} \tag{4.30}$$

A Equação 4.28 obtém o valor normalizado da descalibragem entre duas distribuições. No numerador alguma medida de divergência é utilizada para calcular o valor entre as duas distribuições $MedDiv(p,q)$ e no denominador a medida é utilizada para calcular a diferença entre a distribuição alvo e uma distribuição realizada sem nenhum item $MedDiv(p,q(\{\}))$. A Equação 4.29 calcula para cada posição o valor da descalibragem normalizado. No numerador, para cada posição é encontrado o valor da descalibragem que são somados $\sum_{j=1}^{N} MC(p, q(LR_u@j))$ e no final o denominador realiza uma média N. Na Equação 4.30 para cada usuário no sistema o valor da descalibragem acumulada em todas as posições é chamado e somado e, ao final, uma média é realizada.

4.7 COEFICIENTES DECISÓRIOS PARA CALIBRAGEM

Para além das métricas de avaliação previamente apresentadas, é necessário elucidar um meio de decidir qual combinação de sistema obtém os melhores desempenhos, considerando que este independa do valor do peso do balanceamento. Esta forma de avaliação permite afirmar qual a melhor: medida de divergência, formulação do balanceamento e algoritmo recomendador. Assim, como parte desta dissertação são propostos dois coeficientes decisórios. Estes que são baseados nas medidas de dispersão dos coeficientes estatísticos, por exemplo, o coeficiente de variação.

A Equação 4.31 demonstra o primeiro coeficiente decisório, o Coefficient of Calibration Error (CCE). A proposta é uma divisão do MACE, apresentado na seção anterior, pelo Mean Average Precision (MAP), apresentado na Equação 2.26. Ambas as métricas trabalham sobre toda a lista de recomendação, analisando posição a posição, acumulando os resultados ao longo da lista. Na Equação 4.31 tem-se que, o numerador é a média de todos os valores do MACE obtidos sobre todas as observações, independente do valor do peso do balanceamento; o denominador é a média de todos os valores do MAP obtidos sobre todas as observações da métrica, independente do valor do peso do balanceamento.

$$CCE = \frac{\overline{MACE}}{\overline{MAP}} \tag{4.31}$$

A Equação 4.32 demonstra o segundo coeficiente decisório, o Coefficient of MisCalibration (CMC). A proposta é uma divisão do MRMC, apresentado na seção anterior, e o MAP, apresentado na Equação 2.26. Ambas as métricas trabalham sobre toda a lista de recomendação, analisando posição a posição, acumulando os resultados ao longo da lista. Na Equação 4.32 tem-se que, o numerador é a média de todos os valores do MRMC

obtidos sobre todas as observações, independente do valor do peso do balanceamento; o denominador é a média de todos os valores do MAP obtidos sobre todas as observações da métrica, independente do valor do peso do balanceamento.

$$CMC = \frac{\overline{MRMC}}{\overline{MAP}} \tag{4.32}$$

Os coeficientes com valores grandes indicam um alto erro/descalibragem e uma baixa precisão. Os valores dos coeficientes quando pequenos indicam uma baixa descalibragem e uma alta precisão. É recomendável escolher a combinação de sistema que possua um pequeno valor de coeficiente.

4.8 PROTOCOLO DECISÓRIO PARA SISTEMAS CALIBRADOS

Como uma das principais contribuições, esta dissertação propõe um protocolo de decisão para sistemas de recomendação calibrados. Este protocolo visa auxiliar a implementação e escolha do melhor sistema calibrado para o domínio da aplicação. As implementações dos sistemas de recomendação são dependentes do domínio da aplicação.

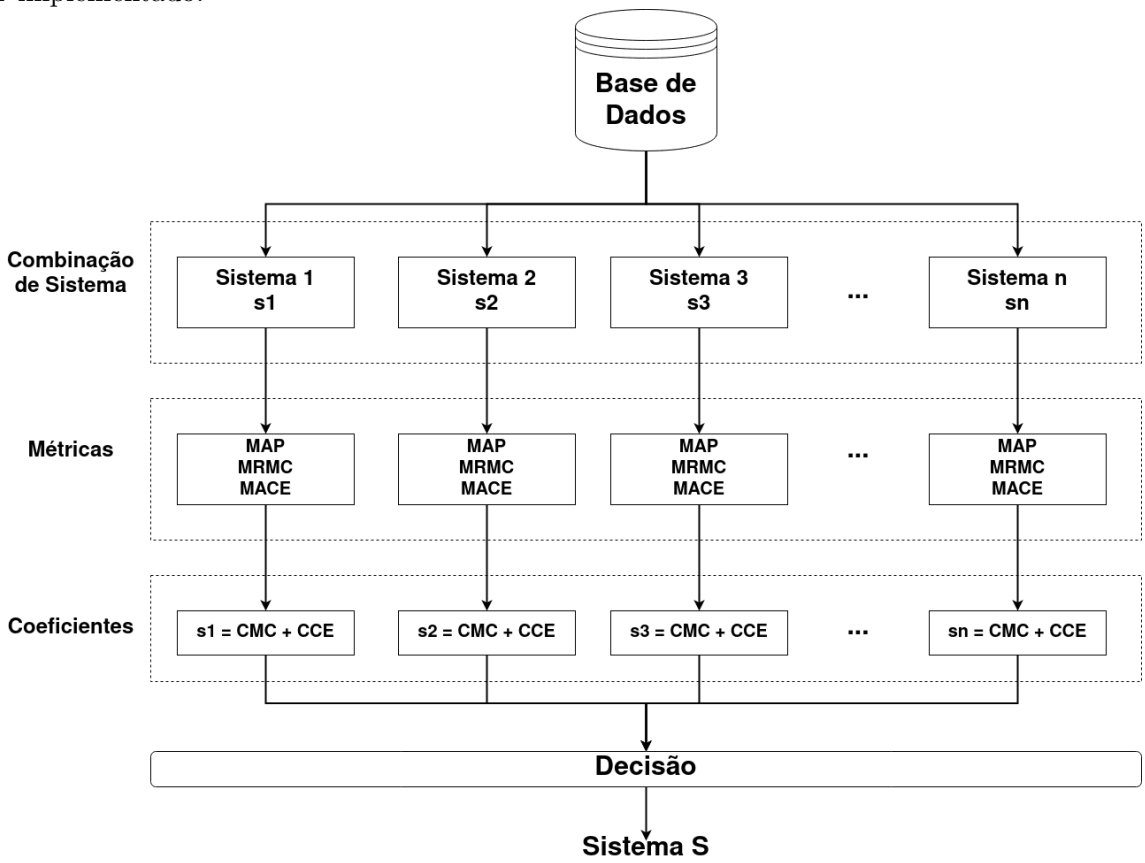
Na seção 4.1 é proposto um modelo de sistema de recomendação calibrado dividido em três etapas, este que foi detalhado ao longo do capítulo. Para avaliar o sistema, na Seção 4.6, duas métricas de avaliação foram apresentadas, estas que são especializadas em avaliar sistemas calibrados. Ao executar e avaliar um sistema calibrado pode-se entender o quão satisfatório ou não o seu desempenho representa. Entretanto, quando se deseja encontrar para uma determinada base de dados de um domínio seu melhor sistema S, faz-se necessário o uso dos coeficientes decisórios apresentados na seção anterior. Assim, o protocolo proposto visa encontrar este sistema S a partir de comparações entre duas ou mais combinações de sistemas calibrados.

A Figura 4.8 ilustra o protocolo proposto. Como é possível verificar na figura, a base de dados é o ponto de início do protocolo. A partir do modelo de sistema proposto na Seção 4.1 cria-se combinações de sistema calibrado, que por sua vez são executados de forma independente como demonstrado na figura. As Figuras 4.1 e 4.2 demonstram como o modelo de sistema proposta funciona. Com os resultados de cada combinação pode-se metrificar o desempenho do sistema. Os valores obtidos pelas métricas são utilizados para obter os coeficientes, que por sua vez permitem elucidar melhor o desempenho do sistema, levando em consideração a precisão e a calibragem. Com os coeficientes de cada sistema (s_1, s_2, s_3... s_n) pode-se extrair qual é a melhor combinação de sistema calibrado para a base de dados. Esta decisão é dada observando qual é o menor valor de s_i. A partir da equação apresentada abaixo:

$$S = \min(s_1,\ s_2,\ s_3 ...\ s_n) \tag{4.33}$$

O protocolo necessita que cada uma das combinações de sistema possua as etapas apresentadas, assim como cada componente das etapas. Na etapa de processamento qualquer algoritmo recomendador pode ser utilizado, desde que gere os itens candidatos. Assim como, na etapa de pós-processamento pode-se variar as formulações de cada com-

Figura 4.8 Protocolo para decidir qual combinação de sistema calibrado é mais indicado para ser implementado.



Fonte: Figura elaborada pelo autor.

ponente, como os apresentados na Seção 4.5, em que diversos componentes apresentam mais de uma formulação proposta nesta dissertação.

O total de combinações que sistemas calibrados que podem ser gerados é dependente de quantas formulações diferentes serão utilizadas em cada componente. Por exemplo, supondo que para uma determinada base de dados: 2 tipos de pré-processamento tenham sido aplicados, 5 algoritmos recomendadores sejam utilizados, 1 distribuição de gêneros, 3 medidas de divergência, 2 medidas de relevância, 21 pesos do balanceamento, 2 balanceamentos e 2 algoritmos de seleção. Assim, ao total o protocolo executará $2 \cdot 5 \cdot 1 \cdot 3 \cdot 2 \cdot 21 \cdot 2 \cdot 2 = 5040$ combinações de sistemas. Para cada combinação as três métricas (MAP, MACE e MRMC) são aplicadas, obtendo um valor de desempenho cada. A partir das métricas os coeficientes são calculados para cada uma das combinações. E por fim, o melhor sistema S é encontrado.

Na Seção 1.4 foi apresentada a questão de pesquisa que trata sobre como encontrar o melhor sistema de recomendação calibrado. Nesta seção foi traçado um meio de encontrar este sistema. No Capítulo 5 é apresentado o experimento realizado para testar o protocolo. Na Seção 1.4 também foi apresentada a questão que trata sobre a possibilidade de padronizar os sistemas calibrados de modo que auxilie o desenvolvimento. Assim, nesta

seção e na Seção 4.1 foram apresentados o protocolo e o modelo de sistema calibrado propostos nesta dissertação, buscando padronizar e facilitar futuras implementações.

4.9 COMPARAÇÃO COM O ESTADO-DA-ARTE

A fim de possibilitar uma melhor comparação deste trabalho com o estado-da-arte, a Tabela 4.12 apresenta o que é proposto pelo estado-da-arte até o momento e no que este trabalho difere. As colunas são referentes ao que o estado-da-arte propõe e o que este trabalho propõe e nas linhas são apresentados os tópicos. As marcações com asterisco (*) representam a proposta deste trabalho. Os dois estudos tidos como estado-da-arte na tabela foram selecionados baseados na seguintes diretivas, se o estudo implementa a etapa de pós-processamento e o usa para gerar listas de recomendação então é considerado como estado-da-arte. Estudos de Abdollahpouri et al. (2019a, 2020) e Lin et al. (2020) são relevantes para a comunidade, entretanto os trabalhos são focados em análises da descalibragem provindas diretamente da etapa de processamento, assim não há implementação do pós-processamento.

Tabela 4.12 Comparação entre contribuições do estado-da-arte e desta dissertação.

Tópicos	**Steck (2018)**	**Kaya e Bridge (2019)**	**Esta dissertação**
Recomendadores	1	1	9*
Distribuições	Distribuição Alvo - Gêneros Distribuição Realizada - Gêneros	Distribuição Alvo - Gêneros Distribuição Realizada - Gêneros Distribuição Alvo - Sub-perfil Distribuição Realizada - sub-perfil	Distribuição Alvo - Gêneros Distribuição Realizada - Gêneros
Medida de Divergência	Kullback-Leibler	Kullback-Leibler	Kullback-Leibler Hellinger* Pearson Chi-Square*
Peso do Balanceamento	Constante	Constante	Constante Contagem de gêneros* Variância*
Balanceamento	Linear	Linear	Linear Logarítmico*
Métricas	MC	MC	MACE* MRMC* CCE* CMC*
Bases de Dados	MovieLens 20M	MovieLens 20M Taste Profile	MovieLens 20M Taste Profile

Fonte: Tabela elaborada pelo autor.

As principais contribuições deste estudo são:

- Larga comparação entre nove algoritmos recomendadores;
- Duas formulações para encontrar o peso personalizado do balanceamento: Contagem de gêneros e Variância;

- Uso de duas medidas de divergência para gerar as recomendações: Hellinger e Pearson Chi Square;
- Uma nova formulação de balanceamento (Logarítmico);
- Duas novas métricas: MACE e MRMC;
- Dois coeficientes decisórios: CCE e CMC;
- Um modelo de sistema de recomendação calibrado;
- Um protocolo de decisão para escolha do melhor sistema calibrado para o domínio.

4.10 DETALHAMENTO TÉCNICO

O desenvolvimento do modelo de sistema proposto neste capítulo envolveu diversas tecnologias. Nesta seção são apresentadas as tecnologias mais importantes utilizadas durante a confecção do código, assim como suas descrições.

O sistema e o protocolo propostos foram implementado em Python[1], que é uma linguagem de propósito geral de alto nível, multi paradigma e que suporta os paradigmas de: orientado a objetos, imperativo, funcional e procedural. Possui tipagem dinâmica e uma de suas principais características é permitir a fácil leitura. A linguagem foi projetada com a filosofia de enfatizar a importância do esforço do programador sobre o esforço computacional. Prioriza a legibilidade do código sobre a velocidade ou expressividade. Para as bibliotecas externas, o Python provê o Pip, que é o seu sistema de gerenciamento de bibliotecas. Para além do Pip, o Python conta com Anaconda, que é um gerenciador de versões de ambiente Python. Todos os recursos apresentados foram utilizados para gerenciar as seguintes bibliotecas:

- **Surprise**[2]: É uma biblioteca para experimentos com sistemas de recomendação, possuindo diversos algoritmos e métricas já implementadas;
- **Pandas**[3]: É uma biblioteca de código aberto amplamente utilizada na comunidade acadêmica. Esta se tornou extremamente útil pelo seu desempenho e pela sua capacidade de simplificar tarefas complicadas de manipulação de dados;
- **Matplotlib**[4]: É uma biblioteca com recursos para a geração de gráficos 2D a partir de vetores. Gráficos comuns podem ser criados com alta qualidade a partir de comandos simples, inspirados nos comandos gráficos do MATLAB;
- **NumPy**[5]: É um biblioteca para a linguagem Python que suporta arrays e matrizes multidimensionais. A NumPy facilita a manipulabilidade dos dados, assim como a sua estruturalização;

[1] https://www.python.org/
[2] http://surprise.readthedocs.io/
[3] https://pandas.pydata.org/
[4] https://matplotlib.org/
[5] http://www.numpy.org/

- **SciPy**[6]: É uma biblioteca de código aberto que foi feita para matemáticos, cientistas e engenheiros. Visa facilitar a implementação de modelos matemáticos, contendo diversas funções implementadas.

4.11 SUMÁRIO

Neste capítulo foram apresentadas as propostas desta dissertação. Inicialmente o sistema calibrado e suas etapas foram definidas, desde a modelagem dos dados, a predição de valores e a calibragem no pós-processamento. Foram descritos formalmente cada um dos modelos de dados utilizados. Foi demonstrado o funcionamento do sistema através de um pseudo código. Também foi apresentada cada formulação utilizada no pós-processamento, assim como foi apontado o local de chamada no pseudo-código. Métricas e coeficientes que fazem parte da proposta foram descritos e debatidos. E, por fim, o protocolo decisório para sistemas calibrados foi apresentado, debatido e exemplificado. No próximo capítulo o protocolo será avaliado a partir de um experimento utilizando bases de dados públicas.

[6]https://www.scipy.org/

Capítulo 5

AVALIAÇÃO EXPERIMENTAL

Neste capítulo é discutida a avaliação experimental do modelo de sistema proposto a partir do protocolo de decisão apresentado previamente. Como resultado do experimento deseja-se encontrar qual é a melhor combinação de sistema calibrado S para cada base de dados. Inicialmente, são apresentadas as bases de dados usadas e como seus dados foram tratados durante a etapa de pré-processamento, assim como uma análise dos dados. Em seguida, é discutida a metodologia aplicada no sistema e na avaliação, assim como os parâmetros do sistema e seus valores. Após, são apresentadas as métricas de avaliação que permitem elucidar o desempenho do sistema. Os resultados obtidos são apresentados e devidamente debatidos. Por fim, são discutidos e sumarizados os resultados, assim como são apontados pontos de melhorias.

5.1 BASES DE DADOS

Para possibilitar uma avaliação ampla das combinações de sistemas são utilizadas duas bases de dados pertencentes a diferentes domínios. As duas bases de dados foram escolhidas a partir de trabalhos do estado-da-arte citados na Tabela 4.12, assim como a escolha pelos experimentos serem *offline*.

5.1.1 MovieLens

O *MovieLens*[1] é uma base de dados pública e de acesso gratuito, utilizada nos trabalhos relacionados de Steck (2018), Kaya e Bridge (2019) e Abdollahpouri et al. (2019a, 2020). Ela contém metadados que descrevem os filmes, como: título, gênero e *tags*, além de *links* para mais informações em outras bases; e também possui dados sobre as preferências dos usuários. Ao longo dos anos, o *MovieLens* lançou diversos subconjuntos, em particular, neste trabalho foi utilizado o *MovieLens 20M*[2].

[1] https://grouplens.org/datasets/movielens/
[2] https://www.kaggle.com/grouplens/movielens-20m-dataset

O *MovieLens 20M* possui: 19 gêneros, 138.493 usuários, 27.278 filmes e 20.000.263 *ratings* (transações) com os valores $w_{u,i}$ entre $]0;5] \in \mathbb{R}$. Estes valores $w_{u,i}$ expressam a nota que o usuário $u \in U$ atribuiu ao filme $i \in I$ (feedback explícito). Sobre a base de dados bruta foi aplicada uma sequência de passos para **limpar** e **filtrar** base de dados que é semelhante a outros estudos (LIANG et al., 2016; STECK, 2018; KAYA; BRIDGE, 2019):

- i) foi atribuído ao modelo do usuário MU_u qualquer peso $w_{u,i}$ da transação (*rating*) maior ou igual a 4;
- ii) foram eliminados todos os filmes que não possuam informações sobre os gêneros;
- iii) foi eliminado qualquer filme que não possua transações com os usuários, ou seja, filmes que nunca foram assistidos.

Entretanto, diferente de Steck (2018), Kaya e Bridge (2019) e semelhante a Liang et al. (2016) foram realizados mais dois passos na seguinte ordem:

- iv) foram atribuídos ao modelo de itens bloqueados MIB_u todos os itens com o valor $w_{u,i}$ da transação (*rating*) menor que 4;
- v) foram removidos todos os usuários que possuem um total de preferências menor que 30;
- vi) foram eliminados todos os itens com menos de 3 transações com os usuários.

Após todos os passos, os dados resultantes estão limpos, filtrados e foram modelados de acordo com cada modelo de dados, possuindo assim: 80.672 usuários (i.e. $|U| = 80.672$), 12.736 filmes (i.e. $|I| = 15.400$), 9.013.655 *ratings* com $w_{u,i}$ entre $[4;5] \in \mathbb{R}$ (i.e. $|MU| = 9.013.655$) e 19 gêneros (i.e. $|G| = 19$). O tamanho total de instâncias dentro do modelo de itens bloqueados de todos os usuários foi 10.004.853 (i.e. $|MIB| = 10.004.853$).

5.1.2 One Million Songs

O projeto One Million Songs (OMS)[3] disponibiliza uma base de dados pública e de acesso gratuito, que é utilizada por diversos trabalhos (BERTIN-MAHIEUX et al., 2011; MCFEE et al., 2012; LIANG et al., 2016; KAYA; BRIDGE, 2019). A base contém metadados que descrevem as músicas em mais de 55 campos, como: título, álbum, artista, ano, duração, tempo, tons, dentre outros. As chaves únicas que identificam cada música também podem ser utilizadas para conectar a base de dados a *plugins* que trazem informações complementares. Um destes *plugins* é o conjunto de dados do *tagtraum genre annotations*[4], que garante informações sobre os gêneros das músicas. Outro *plugin* é o *Taste Profile*[5], que possui informações sobre as preferências dos usuários, contendo tuplas

[3] http://millionsongdataset.com/
[4] http://www.tagtraum.com/genres/msd_tagtraum_cd2.cls.zip
[5] http://millionsongdataset.com/tasteprofile/

com informações sobre: usuário, música e quantas vezes o usuário ouviu a música. A base de dados e os dois *plugins* foram utilizados neste experimento.

A base de dados do OMS e os *plugins tagtraum genre annotations* e *Taste Profile* possuem juntos: 15 gêneros, 1.019.318 usuários, 1.000.000 de músicas e 48.373.586 transações com os valores $w_{u,i}$ de $[1; 9.667] \in \mathbb{N}$. Este valores $w_{u,i}$ representam quantas vezes um usuário $u \in U$ ouviu uma música $i \in I$ (feedback implícito). Assim, foi aplicada uma sequência de passos para **limpar** e **filtrar** a base de dados que é semelhante a outros estudos (LIANG et al., 2016; KAYA; BRIDGE, 2019):

- i) foram eliminadas as músicas que não possuem informações sobre o gênero;
- ii) foram removidas todas as músicas que nunca foram escutadas pelos usuários.

Diferentemente de Liang et al. (2016) e Kaya e Bridge (2019), outros valores foram atribuídos para os passos:

- iii) foram atribuídos ao modelo do usuário MU_u qualquer peso $w_{u,i}$ da transação (vezes que ouviu) maior ou igual a 2;
- iv) foram atribuídos ao modelo de itens bloqueados MIB_u todos os itens com o valor $w_{u,i}$ da transação (vezes que ouviu) menor que 2;
- v) foram removidos todos os usuários que possuem um total de preferências menor que 30;
- vi) foram eliminados todos os itens com menos de 3 transações.

Após a aplicação de todos os passos o conjunto de dados resultantes possuía: 15 gêneros (i.e. $|G| = 15$), 94.611 usuários (i.e. $|U| = 94.611$), 98.305 músicas (i.e. $|I| = 98.305$) e 4.807.615 transações com $w_{u,i}$ entre $[2; 9.667] \in \mathbb{N}$ (i.e. $|MU| = 4.807.615$). O tamanho total de instâncias dentro do modelo de itens bloqueados de todos os usuários foi 18.158.299 (i.e. $|MIB| = 18.158.299$).

5.2 ANÁLISE DAS BASES

A Tabela 5.1 apresenta um resumo sobre a composição numérica dos modelos. O tratamento na base de dados do *MovieLens* produziu uma amostra onde: o número de usuários representa aproximadamente 59% da população, o número de itens representa aproximadamente 57%, mantendo assim aproximadamente 46% do total de transações. Isto significa que, 41% da população de usuários pertence ao contexto da partida a frio, que não é abordado neste trabalho. O corte na base do OMS produziu uma amostra onde: o número de usuários representa aproximadamente 10% da população, o número de itens representa aproximadamente 10%, mantendo assim aproximadamente 10% do total de transações. Isto implica que 90% da população de usuários pertence ao contexto da partida a frio. Em ambas as bases todos os gêneros foram mantidos.

Cada base de dados é dada como entrada da combinação de sistema e passa por uma sequência de passos previamente apresentados: limpeza e filtragem apresentadas seção

anterior e modelagem apresentada na Seção 4.3. Assim, nesta seção são analisados os dados pertencentes a estes modelos, descrevendo-os e apontando o grau de correlação de *Spearman* (KOKOSKA; ZWILLINGER, 1999) entre as variáveis. A seguir são apresentados os principais pontos de análise: popularidade e gênero.

Tabela 5.1 Total de instâncias em cada modelo utilizado no sistema.

Base	$\|G\|$	$\|U\|$	$\|I\|$	$\|MU\|$	$\|MIB\|$
Movielens	19	80.672	15.400	9.013.655	10.004.853
OMS	15	94.611	98.305	4.807.615	18.158.299

Fonte: Tabela elaborada pelo autor a partir dos dados das bases.

5.2.1 Popularidade

Um dos problemas que a técnica de filtragem colaborativa enfrenta é o viés por popularidade. Este que toma os itens mais populares como os mais possíveis de serem recomendados, criando assim uma bolha onde os mais populares continuam populares e os menos populares não ganham destaque, como apresentado na Seção 2.6. Assim, neste trabalho é analisada a popularidade dos itens dentro da base de dados. Abdollahpouri et al. (2020) estuda o viés por popularidade nas recomendações observando a popularidade antes da etapa de processamento e depois da etapa de processamento.

Tabela 5.2 Análise da popularidade nas bases.

Base	**Máximo**	**Mínimo**	**Média**	**Mediana**	**Desvio padrão**	**Variância**	**Assimetria**	**Curtose**
Movielens	41.335	3	585	41	2.124	4.513.517	8	94
OMS	11.574	3	48	11	176	31.209	20	764

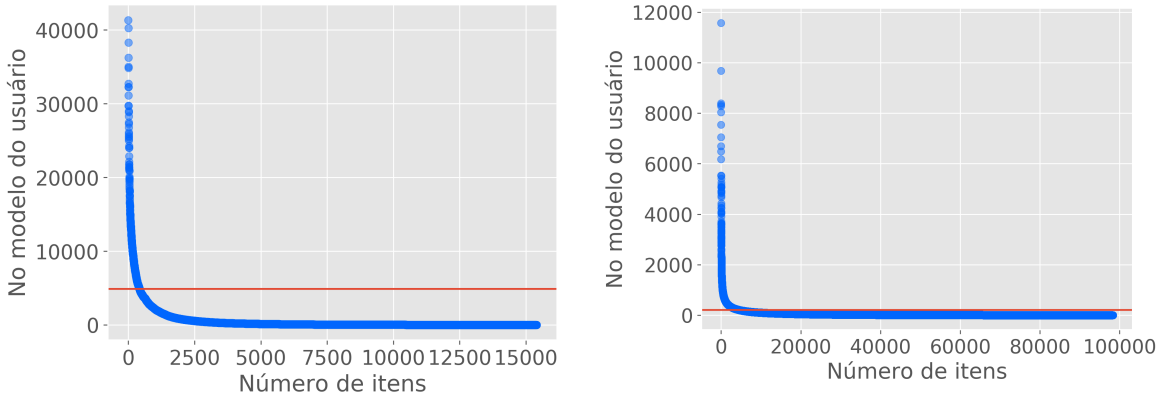
Fonte: Tabela elaborada pelo autor a partir dos dados das bases.

A Tabela 5.2 apresenta os números sobre a popularidade dos itens nas bases. No *Movielens* tem-se que: o item mais popular encontra-se em 41.335 modelos dos usuários e o menos popular em 3 modelos; a média de popularidade é de 585 e a mediana é de 41, indicando uma centralidade baixa da popularidade; o desvio padrão é de 2.124 e a variância é de 4.513.517, demonstrando a alta variação dos valores da popularidade; a assimetria é de 8 e a curtose é de 94, indicando que a maior quantidade dos itens possui uma baixa popularidade concentrando-se sobre uma faixa de valores. Na base do OMS têm-se que: o item mais popular encontra-se em 11.574 modelos dos usuários e o menos popular em 3 modelos; a média de popularidade é de 48 e a mediana é de 11, indicando uma centralidade baixa da popularidade; o desvio padrão é de 176 e a variância é de 31.209, demonstrando a alta variação dos valores da popularidade; a assimetria é de 20 e a curtose é de 764, indicando que a maior quantidade dos itens possui uma baixa popularidade concentrando-se sobre uma pequena faixa de valor da popularidade.

Como indicado pela assimetria e a curtose, a base de dados possui uma concentração da popularidade em valores pequenos, ou seja, a maioria dos itens não são populares. Assim, a Figura 5.1 representa este estado dos dados que é comumente encontrado em bases de dados reais e é chamado de cauda longa (*long-tail*) (ABDOLLAHPOURI; BURKE;

Figura 5.1 A cauda longa: poucos itens com muitas transações e muitos itens com poucas.

(a) Movielens (b) One Million Songs



Fonte: Figuras elaboradas pelo autor a partir dos dados das bases.

MOBASHER, 2017). Este estado mostra que poucos itens da base possuem uma grande quantidade de transações com os usuários enquanto a maior parte dos itens possuem uma menor interação com os usuários. A Figura 5.1(a) apresenta a cauda longa na base do *MovieLens* e a 5.1(b) representa a cauda longa na base do OMS. A correlação de *Spearman* entre o número de itens no sistema e a popularidade dos mesmos é: No *Movielens* de -0,999, demonstrando que há uma correlação muito forte e negativa; No OMS, de -0.999 demonstrando que também há uma correlação muito forte e negativa. Estes valores da correlação mostram que quanto maior o número de itens no sistema menor é a popularidade do mesmo.

A cauda longa pode ser dividida em três partes (ABDOLLAHPOURI; BURKE; MOBASHER, 2017; ABDOLLAHPOURI et al., 2019b, 2020): cabeça curta (*short head*), meio da cauda (*medium tail*) e cauda longa distante (*distant long tail*). No estudo de Abdollahpouri, Burke e Mobasher (2017) é definido que: os itens populares (cabeça curta) são aqueles que possuem a maior quantidade de transações na base, os itens do meio da cauda são todos os outros que são alcançáveis pelo algoritmo recomendador e a cauda longa distante é composta pelos itens que não são alcançados pelo algoritmo recomendador, devido ao estado de partida a frio. Assim, neste trabalho segue-se a mesma definição. Entretanto, diferente deles, neste trabalho não é estudada a cauda longa distante, estes itens são removidos durante a etapa de pré-processamento. Na Figura 5.1 os itens que estão acima da linha vermelha são os mais populares, ou seja, juntos possuem mais transações que todos os itens abaixo da linha.

A Tabela 5.3 demonstra a quantidade de itens pertencentes a cada partição da cauda longa e a quantidade de transações que cada partição acumula. No *Movielens* têm-se que: a cabeça curta possui 424 itens, estes que possuem juntos 4.507.950 transações; o meio da cauda possui ao todo 14.976 itens, estes que possuem 4.505.705 transações. Na base do OMS têm-se que: a cabeça curta possui 4.611 itens, estes que possuem ao todo 2.403.880 transações; o meio da cauda possui ao todo 93.694 itens, estes que possuem 2.403.735 transações.

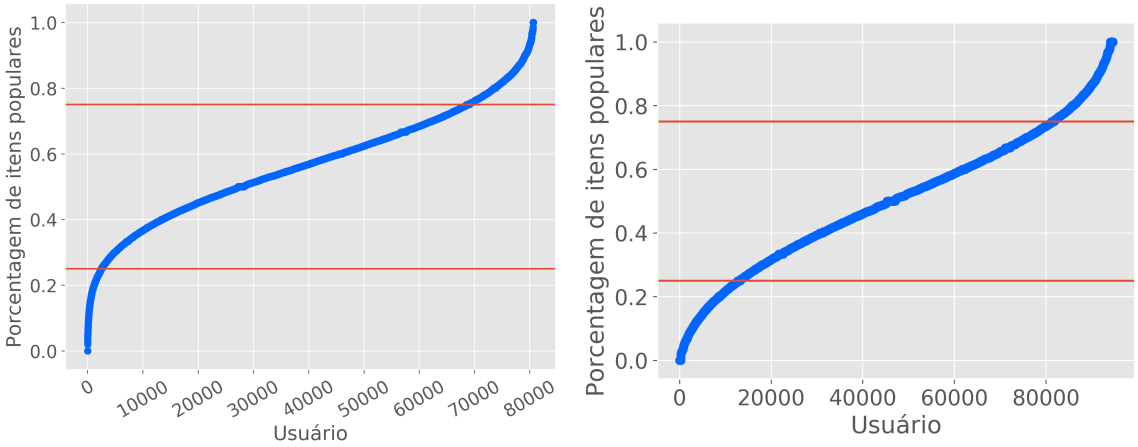
Tabela 5.3 Total de itens pertencentes a cada partição da cauda.

Bases	Cabeça curta	Total de transações (Cabeça)	Meio da cauda	Total de transações (Meio)
Movielens	424	4.507.950	14.976	4.505.705
OMS	4.611	2.403.880	93.694	2.403.735

Fonte: Tabela elaborada pelo autor a partir dos dados das bases.

Figura 5.2 Usuários e as porcentagens de itens populares que constituem seus modelos.

(a) Movielens (b) One Million Songs



Fonte: Figuras elaboradas pelo autor a partir dos dados das bases.

Tabela 5.4 Total de usuários pertencentes a cada grupo.

Bases	Nicho	Diverso	Focado
Movielens	2.554	65.997	12.121
OMS	13.260	67.932	13.419

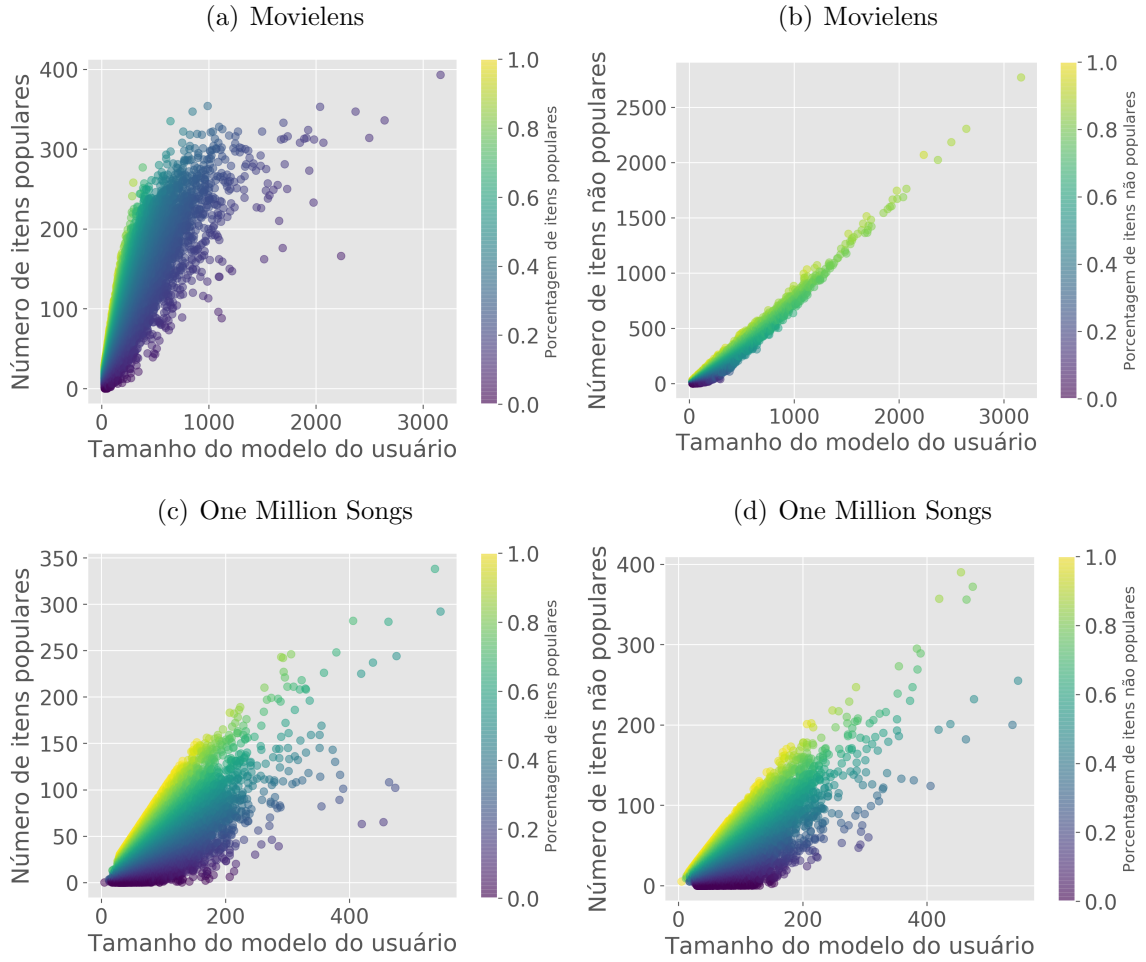
Fonte: Tabela elaborada pelo autor a partir dos dados das bases.

Em seu trabalho Abdollahpouri et al. (2019b) apresentam três grupos de usuários: nicho, diverso e focado (Seção 3.4). Neste trabalho segue-se esta divisão, entretanto é considerada a divisão dos grupos baseada nos quartis da distribuição dos dados. A Figura 5.2 é obtida através da verificação da porcentagem de itens populares dentro do modelo de cada usuário, desenhando-se no gráfico os dados ordenados pela porcentagem. Assim, a Figura 5.2 e a Tabela 5.4 descrevem a designação de cada grupo. Para ambas as bases a primeira linha vermelha de baixo pra cima representa o primeiro quartil, ou seja, usuários neste quartil possuem poucos itens populares, logo são adicionados ao grupo de nicho.

No *Movielens* o grupo de nicho possui 2.554 usuários e no OMS possui 13.260 usuários. Entre a primeira e a segunda linha vermelha (primeiro e terceiro quartil) encontram-se os usuários que possuem tanto itens populares quanto não populares, sendo assim classificados no grupo diverso. No *Movielens* o grupo diverso possui 65.997 usuários e no OMS possui 67.932 usuários. Por fim, acima da segunda linha vermelha de baixo pra

acima encontram-se os usuários focados em itens populares, sendo seus modelos compostos por maioria desses itens. No *Movielens* o grupo focado em itens populares possui 12.121 usuários e no OMS possui 13.419 usuários. Ao correlacionar os dados da Figura 5.2, são obtidos os valores usando *Spearman*: no *Movielens* é de 0.999, demonstrando uma correlação muito forte e positiva, e no OMS é de 0.999, demonstrando também uma correlação muito forte e positiva.

Figura 5.3 Composição do modelo do usuário com itens populares e não populares e suas respectivas escalas.



Fonte: Figuras elaboradas pelo autor a partir dos dados das bases.

A Figura 5.3 analisa o tamanho dos modelos dos usuários - eixo x - pelo número de itens populares ou não populares - eixo y - que compõem cada modelo em ambas as bases, sendo a variação gradual da cor a representação da porcentagem de itens populares ou não populares que compõem cada modelo. A Figura 5.3(a) representa a base do *Movielens* e a Figura 5.3(c) representa a do OMS, ambas analisadas a partir dos itens populares (cabeça curta). A variação das cores nos mostra que conforme aumenta o tamanho do modelo do usuário a porcentagem de itens populares diminui, apesar do aumento no

número destes itens nos modelos. Isto acontece devido à pequena porcentagem de itens populares nas bases (0,027 para o *Movielens* e de 0,051 para o OMS) que o usuário vai consumindo conforme o tamanho do modelo aumenta até não restar mais nenhum item popular para interagir. A Figura 5.3(b) representa a base do *Movielens* e a Figura 5.3(d) representa a do OMS, ambas as bases analisadas a partir dos itens não populares. A variação gradual das cores nos mostra que conforme aumenta o tamanho do modelo do usuário a porcentagem de itens não populares aumenta, seguido pelo aumento no número dos itens não populares. Isto acontece devido às bases possuírem um grande número de itens não populares aos quais os usuários recorrem quando esgotam os itens populares.

Tabela 5.5 Correlações de *Spearman* baseadas na Figura 5.3.

Dados	**Movielens**	**OMS**
Tamanho do modelo do usuário x Número de itens populares	0,9	0,573
Tamanho do modelo do usuário x Porcentagem de itens populares	-0,344	-0,027
Tamanho do modelo do usuário x Número de itens não populares	0,883	0,585
Tamanho do modelo do usuário x Porcentagem de itens não populares	0,344	0,027

Fonte: Tabela elaborada pelo autor a partir dos dados das bases.

A Tabela 5.5 mostra as correlações entre as variáveis da Figura 5.3. A correlação entre o tamanho do modelo e o número de itens populares para o *Movielens* é forte e positiva, e para o OMS é moderada. Em seguida mostra uma correlação fraca e negativa para o *Movielens* e bem fraca e negativa no OMS entre o tamanho do modelo do usuário e a porcentagem de itens populares que os compõem. A tabela também mostra que entre o tamanho do modelo e o número de itens não populares há uma correlação forte e positiva no *Movielens*, e moderada e positiva no OMS. Também mostra uma correlação fraca e positiva para o *Movielens*, e bem fraca e positiva para o OMS entre o tamanho do modelo do usuário e a porcentagem de itens não populares que compõem o modelo.

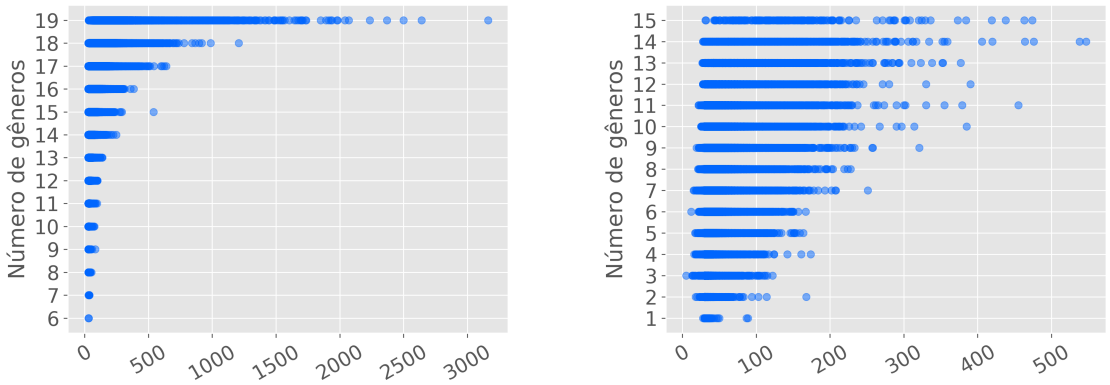
5.2.2 Gêneros

A proposta deste trabalho é visar ser justo com os gêneros pertencentes aos modelos de cada usuário $g \in MU_u$. Assim, é realizada uma análise dos gêneros que compõem os modelos. Como é possível visualizar na Tabela 5.1: o *MovieLens* possui 19 gêneros, sendo cada item composto por um, dois ou mais gêneros simultaneamente; o OMS possui 15 gêneros, sendo cada item composto por um ou dois gêneros simultaneamente.

A Figura 5.4 representa o crescimento do tamanho dos modelos dos usuários pela quantidade de gêneros que compõem estes modelos. A Figura 5.4(a) demonstra que

Figura 5.4 Tamanho do modelo do usuário pelo número de gêneros que o compõem, demonstrando que quanto maior o modelo maior a quantidade de gêneros que os compõem.

(a) Movielens (b) One Million Songs



Fonte: Figuras elaboradas pelo autor a partir dos dados das bases.

no *Movielens* há uma correlação forte e positiva entre os tamanhos dos modelos e a quantidade de gêneros que os compõem, sendo o valor da correlação 0,725. A figura também mostra que o limite inferior e superior na quantidade de gêneros nos modelos está entre $[6, 19] \in \mathbb{N}$. A base do OMS demonstra na Figura 5.4(b) que também existe a correlação fraca e positiva entre os tamanhos dos modelos e a quantidade de gêneros que os compõem, sendo o valor da correlação 0,363. A figura também mostra que o limite inferior e superior na quantidade de gêneros nos modelos está entre $[1, 15] \in \mathbb{N}$, contendo assim uma maior variação que o *Movielens*.

A Figura 5.5 representa na barra vermelha a proporção média dos gêneros existentes nos itens I e na barra azul a proporção média dos gêneros que compõem os modelos dos usuários $g \in MU$. O *Movielens*, representado na Figura 5.5(a), possui dois gêneros com maior destaque (Drama e Comédia) e dois com menor destaque (*Film-Noir* e *IMAX*). A Figura 5.5(b) mostra que no OMS o gênero *Rock* ganha um grande destaque e o *New Age* um menor destaque.

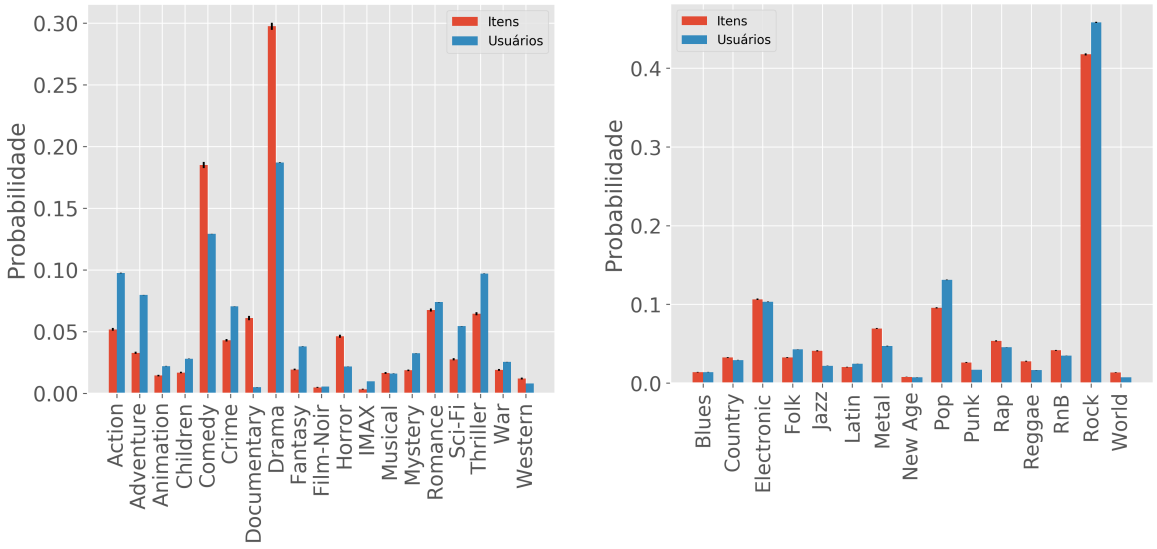
5.3 METODOLOGIA

Nesta seção é apresentada a metodologia aplicada nos experimentos para avaliar as combinações do modelo de sistema proposto, desde os algoritmos utilizados e seus hiper parâmetros até os detalhamentos dos valores constantes utilizados.

O modelo de sistema proposto é desenvolvido utilizando a técnica de filtragem colaborativa apresentada e descrita na Seção 2.6. A escolha por esta técnica foi inspirada nos trabalhos do estado-da-arte que fazem uso da filtragem colaborativa durante a etapa de processamento. A escolha de trabalhar com gêneros durante o pós-processamento para realizar a calibragem também foi baseada no estado-da-arte, sendo que todos os trabalhos relacionados (Seção 3.4.1) fazem uso dos gêneros dos itens. O uso dos gêneros como informação não altera a característica do sistema de ser baseado em filtragem colaborativa,

Figura 5.5 Média das proporções dos gêneros que constituem os itens e os modelos dos usuários.

(a) Movielens (b) One Million Songs



Fonte: Figuras elaboradas pelo autor a partir dos dados das bases.

devido ao gênero ser a única informação dos itens utilizada, além de não alterar a etapa de processamento, onde se usa apenas os *feedbacks* dos usuários.

5.3.1 Algoritmos Recomendadores

Para obter uma maior compreensão da questão de justiça, foram utilizados nove algoritmos recomendadores popularmente conhecidos. Ademais foram comparados seus desempenhos diante das métricas de avaliação. Alguns algoritmos possuem valores de hiper parâmetros que podem otimizar os resultados obtidos. Assim, para encontrar estes valores foi utilizado o *Grid Search*. Para cada algoritmo os valores dos hiper parâmetros foram encontrados e adequados para as bases de dados. A aplicação do *Grid Search* no sistema decide, entre os conjuntos de hiper parâmetros dados como entrada, qual é o melhor conjunto. Como métrica decisória para encontrar o melhor conjunto de hiper parâmetros é utilizado o Mean Absolute Error (MAE) (Seção 2.8.2.1), buscando o menor erro entre o feedback real do usuário ($w_{u,i}$) e o predito pelo recomendador ($w_{r(u,i)}$).

A validação dos melhores hiper parâmetros foi obtida com o *cross-validation* dividindo cada base de dados em três partições. O valor de três partições foi escolhido por limitações técnicas de memória, processamento e tempo de acesso aos recursos. Foram executadas $3 \cdot v^h$ vezes cada combinação de sistema, onde v^h é o total de conjuntos de hiper parâmetros: 3 é o número de partições, v é o número de valores usados em cada um dos hiper parâmetros e h é o número de hiper parâmetros que o algoritmo recomendador possui. Os algoritmos usados e seus hiper parâmetros ótimos são apresentados abaixo:

- **User-KNN:** O K Nearest Neighbors (KNN) baseado no usuário, apresentado na

Seção 2.6.3.1, foi testado com $3^2 = 9$ combinações de hiper parâmetros em cada partição, totalizando 27 execuções. Ao final o melhor conjunto foi: $k = 30$ como o número de vizinhos e o Mean Squared Difference (MSD) como medida de similaridade. Os hiper parâmetros foram iguais para ambas as bases;

- **Item-KNN:** O KNN baseado no item, apresentado na Seção 2.6.3.2, foi testado com $3^2 = 9$ combinações de hiper parâmetros em cada partição, totalizando 27 execuções. Ao final o melhor conjunto foi: $k = 30$ como o número de vizinhos e a Correlação de Person como medida de similaridade. Os hiper parâmetros foram iguais para ambas as bases;

- **Slope One:** Apresentado na Seção 2.6.5, o algoritmo recomendador não possui hiper parâmetros;

- **NMF:** Apresentado na Seção 2.6.4.3, o algoritmo recomendador foi testado com $2^8 = 256$ combinações de hiper parâmetros em cada partição, totalizando 768 execuções. Ao final o melhor conjunto foi: i) para o *Movielens* o número de épocas $ne = 50$, o número de fatores latentes $f = 50$, a taxa de aprendizado do usuário como $\gamma_u = 0.005$ e do item como $\gamma_i = 0.005$, o termo de regularização constante do usuário como $\lambda_u = 0.005$ e do item como $\lambda_i = 0.05$ e termo de regularização do viés do usuário como $\lambda_{bu} = 0.005$ e do item como $\lambda_{bi} = 0.005$; ii) para o OMS o número de épocas $ne = 30$, o número de fatores latentes $f = 50$, a taxa de aprendizado do usuário como $\gamma_u = 0.003$ e do item como $\gamma_i = 0.005$, o termo de regularização constante do usuário como $\lambda_u = 0.003$ e do item como $\lambda_i = 0.05$ e o termo de regularização do viés do usuário como $\lambda_{bu} = 0.003$ e do item como $\lambda_{bi} = 0.005$;

- **SVD:** O Singular Value Decomposition (SVD) apresentado na Seção 2.6.4.1, foi testado com $3^4 = 81$ combinações de hiper parâmetros em cada partição, totalizando 243 execuções. Ao final o melhor conjunto foi: i) para o *Movielens* o número de épocas é de $ne = 50$, o número de fatores latentes é de $f = 50$, a taxa de aprendizado como $\gamma = 0.005$, e a taxa de regularização constante como $\lambda = 0.01$; ii) para o OMS o número de épocas é de $ne = 30$, o número de fatores latentes é de $f = 150$, a taxa de aprendizado como $\gamma = 0.001$ e a taxa de regularização constante como $\lambda = 0.05$;

- **SVD++:** A melhoria Singular Value Decomposition Plus Plus (SVD++) apresentada na Seção 2.6.4.2, foi testada com $3^4 = 81$ combinações de hiper parâmetros em cada partição, totalizando 243 execuções. Ao final o melhor conjunto foi: o número de épocas é de $ne = 50$, o número de fatores latentes é de $f = 50$, a taxa de aprendizado como $\gamma = 0.005$ e a taxa de regularização constante como $\lambda = 0.01$. Os valores foram os mesmos para ambas as bases;

- **Co-Clustering:** Apresentado na Seção 2.6.6, o algoritmo recomendador foi testado com $3^3 = 27$ combinações de hiper parâmetros, totalizando 81 execuções. Ao final o melhor conjunto foi: i) para o *Movielens* o número de épocas é $ne = 50$, o número de *clusters* de usuários como $\overline{C_u} = 3$ e de itens como $\overline{C_i} = 7$; ii) para o OMS o

número de épocas é $ne = 10$ e o número de *clusters* de usuários como $\overline{C_u} = 3$ e de itens como $\overline{C_i} = 3$;

- **Melhor nota:** O algoritmo não possui hiper parâmetros (Seção 2.6.2.2);
- **Popularidade:** O algoritmo não possui hiper parâmetros (Seção 2.6.2.1).

Todos os algoritmos usados neste trabalho foram apresentados na Seção 2.6. Não pertence ao escopo desta dissertação propor nenhum tipo de melhoria nos parâmetros ou no funcionamento dos recomendadores. A implementação em Python dos algoritmos é disponibilizada pela biblioteca Surprise (HUG, 2017), assim como o *Grid Search*. A escolha dos recomendadores é baseada nos trabalhos relacionados.

Os algoritmos User-KNN, Item-KNN e SVD++ foram utilizados nos trabalhos de Abdollahpouri et al. (2019a, 2020) e Lin et al. (2020). O SVD foi adicionado neste trabalho como um contra-ponto ao SVD++, que por sua vez é uma melhoria do SVD, assim será verificado também se a afirmação realizada por Koren e Bell (2015) é verdadeira quanto ao SVD++ obter melhores resultados que o SVD. Os trabalhos relacionados utilizam o Biased Matrix Factorization - BMF e o Weighted Regularized Matrix Factorization (WRMF) como fatoradores de matrizes (além do SVD++), assim foi implementado o recomendador Non-negative Matrix Factorization (NMF) com o bias do usuário e do item. O Co-Clustering foi implementado por ser um agrupador, possuindo um funcionamento diferente dos algoritmos apresentados. O Slope One foi adicionado por ser um recomendador sem hiper parâmetros. O recomendador por Popularidade foi implementado por também ser usado nos trabalhos de Abdollahpouri et al. (2019a, 2020) e Lin et al. (2020). No mesmo grupo o recomendador baseado em Melhor nota também foi implementado.

Neste estudo, não é considerado qualquer recomendador baseado em redes neurais, tomando como base que nenhum dos trabalhos relacionados exploram esses algoritmos no contexto de recomendações calibradas. Assim, considera-se que as redes neurais podem ser estudadas em um trabalho futuro.

5.3.2 Experimentos

Com os hiper parâmetros otimizados de cada recomendador é iniciada a etapa de processamento, onde os modelos dos usuários MU são utilizados como base para o treinamento dos algoritmos recomendadores (Algoritmo 1 Linha 1). Neste experimento, de cada modelo do usuário MU_u foram retirados aleatoriamente 70% dos dados para treinamento e 30% para teste. Espera-se que esses 30% dos itens venham a compor a lista de recomendação LR_u ao final de toda a execução. Assim, os dados de testes foram atribuídos ao modelo de itens desconhecidos do usuário MID_u. Sendo esses MID_u utilizados como dados da predição, ou seja, os itens ao qual o algoritmo recomendador prevê o peso $w_{r(u,i)}$. Os trabalhos de Steck (2018), Kaya e Bridge (2019) e Abdollahpouri et al. (2019a, 2020) também realizam o particionamento aleatório, entretanto com outros valores de treinamento e teste.

O algoritmo recomendador recebe dos dados de treinamento, aprende sobre as preferências de cada usuário, para então receber de cada usuário seu modelo de itens desconhecidos e assim prevê um possível valor para o peso $w_{r(u,i)}$. Por exemplo, a Figura 5.3(a)

mostra que há um usuário na base de dados do *Movielens* com 2500 itens em seu MU_u, assim têm-se que: 1750 itens são usados para o aprendizado de suas preferências; 750 itens são itens de testes, estes itens que são adicionados ao modelo itens desconhecidos totalizando assim $|MID_u| = 12900 + 750 = 13650$. Antes do MID_u ser dado como entrada do preditor (Algoritmo 1 Linha 3), são retirados os itens pertencentes do modelo de itens bloqueados MIB_u, assim se $|MIB_u| = 1300$ têm-se que o $|MID_u| = 13650 - 1300 = 12350$. Logo o algoritmo recomendador prevê o $w_{r(u,i)}$ para 12350 itens neste usuário. Com exceção de dois modelos, todos os MID_u na base de dados do *Movielens* são maiores que o apresentado neste exemplo.

O algoritmo de ranqueamento apresentado na Seção 4.5.7 é combinatorial/arranjo, o que torna o trabalho de testar para cada usuário todos os itens dos MID_u inviável, devido à limitação de tempo no uso dos recursos computacionais. Ao aplicar a Equação 4.24 no exemplo anterior têm-se $(12350!)/(12350 - 10)! = 8,224110^{40}$ combinações diferentes de lista de recomendação. Assim, após o algoritmo recomendador predizer o peso $w_{r(u,i)}$ para todos os itens do MID_u é aplicado um filtro, onde o tamanho dos modelos de itens candidatos maximizados é de $|MIC_u^{max}| = 100$ (Algoritmo 1 Linha 3). A filtragem e modelagem dos dados no pós-processamento foi apresentado na Figura 4.2. Os 100 itens foram os maiores $w_{r(u,i)}$, indicando assim uma possível maior preferência do usuário por estes itens. Os trabalhos de Steck (2018) e Kaya e Bridge (2019) utilizam o mesmo valor para filtrar os itens.

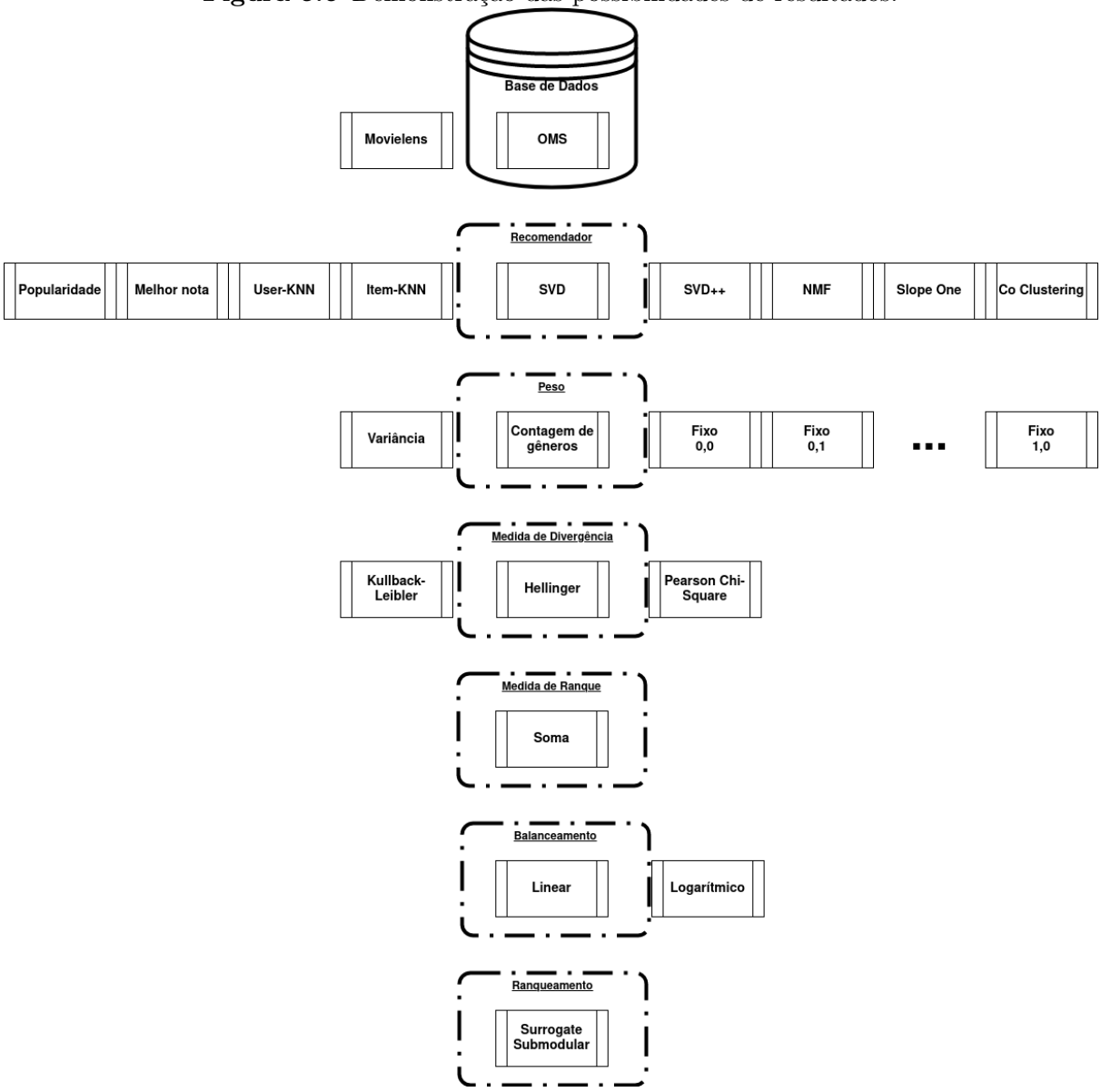
Com o MIC_u^{max} definido e modelado, é iniciado o pós-processamento (Algoritmo 1 Linha 4). Nesta etapa é aplicada a calibragem e a lista de recomendação é gerada. Neste experimento foi utilizando: as três medidas de divergência (Seção 4.5.2), a medida de relevância (Seção 4.5.3), os dois pesos personalizados de balanceamento (Seção 4.5.4), os dois balanceamentos relevância-divergência (Seção 4.5.5), o Maximum Marginal Relevance (MMR) (Seção 4.5.6) e o *surrogate submodular* para o ranqueamento dos itens (Seção 4.5.7).

Os trabalhos de Steck (2018) e Kaya e Bridge (2019) utilizam valores fixos nos pesos do balanceamento, diferente desta dissertação onde os valores são dados de forma individual para os usuários baseando-se nas suas preferências. Entretanto para estabelecer uma comunicação entre os estudos, foram utilizados onze valores fixos de $\lambda \in [0,0; 0,1; 0,2; 0,3; ...; 1,0]$. O valor de $\lambda = 0,0$ significa que o ranque recebe uma total importância e a divergência é ignorada, o resultado da lista de recomendação é o mesmo dos algoritmos recomendadores usados sem o pós-processamento. O valor de $\lambda = 1,0$ significa que o ranque é ignorado e a divergência ganha uma total importância, recomendando assim os itens que são menos divergentes com cada modelo do usuário.

A Figura 5.6 demonstra todas as possibilidades de resultados que o protocolo obtém e avaliará. Cada bloco na figura representa um passo do sistema, desde a inserção da base de dados até o ranqueamento dos itens e em cada passo foi executado um algoritmo. A troca de um desses algoritmos retorna listas de recomendação diferentes do algoritmo anterior executado. Assim, foram obtidas a partir da figura, contando de cima para baixo, $2 \cdot 9 \cdot 13 \cdot 3 \cdot 1 \cdot 2 \cdot 1 = 1404$ combinações de resultados.

Na Seção 4.5.1 foi descrita a possibilidade do uso de um α para evitar a divisão por zero, assim foi utilizado o valor de $\alpha = 0.01$, este valor que também foi aplicado nos

Figura 5.6 Demonstração das possibilidades de resultados.



Fonte: Figura elaborada pelo autor.

trabalhos de Steck (2018), Abdollahpouri et al. (2020), Kaya e Bridge (2019). Na mesma seção foi definido o $p(g|i)$ como parte das Equações 4.8 e 4.9, assim neste trabalho foi aplicada a regra $\frac{1}{|genresIn(i)|}$, ou seja, cada gênero recebe uma probabilidade igual em cada item, como apresentado nd Equação 4.7.

Para validar cada modelo de sistema e estabelecer uma estabilidade nos resultados, cada combinação de sistema foi executado 3 vezes para cada base de dados de forma independente, ou seja, a cada execução os dados de treinamento MU e os de predição MID_u foram diferentes. Os resultados apresentados a seguir são uma média dos valores obtidos em cada base nas 3 execuções. As listas de recomendação foram geradas e avaliadas com

os tamanhos de $[1; 10] \in N$.

5.3.3 Máquina

Como explanado na Seção 4.5.7 e demonstrado em um exemplo na seção anterior, o modelo de sistema proposto é NP-difícil (arranjo), o que o faz realizar uma grande quantidade de operações durante o pós-processamento. Assim, é necessário um computador de alto desempenho (High Performance Computer - HPC) para executar a tarefa integral em um tempo mais hábil. A implementação do sistema proposto foi dada em duas partes.

A primeira parte da implementação do sistema foi executada no computador de alto desempenho alocado na Universidade Estadual de Santa Cruz - UESC. Máquina esta que é gerenciada pelo Núcleo de Biologia Computacional e Gestão de Informações Biotecnológicas - NBCGIB. O nó de execução cedido para o sistema possuía 12 núcleos e 44 GB de *RAM* e foi permitido o uso exclusivo dos recursos da máquina para a execução do sistema proposto. Os devidos reconhecimentos podem ser encontrados na seção de agradecimentos desta dissertação. Os resultados desta primeira parte estão compilados no artigo Silva, Manzato e Durão (2021a).

A segunda parte da implementação do sistema proposto foi executada no HPC chamado SDumont[6], que é localizado no Laboratório Nacional de Computação Cientifica - LNCC. Em Novembro de 2020, o SDumont possuía 33856 cores e uma capacidade de 1849 TFlop/s, ocupando a posição 276 no ranque sobre os 500[7] melhores HPCs. A permissão para o uso do SDumont é realizada sob um processo de submissão de projeto. O projeto do sistema proposto foi aprovado com uma cota de 500 mil unidades de alocação, sendo 1 hora de um núcleo dos nós permitidos para uso equivalente a 2 unidades de alocação. Durante a realização dos experimentos no SDumont dois[8] foram os nós utilizados: 1) MESCA2 com 240 núcleos e 6 TB de *RAM* e 2) GDL com 48 núcleos e 384 GB de *RAM*. Ambos os nós foram utilizados sob o regime de uso exclusivo dos recursos da máquina. Os devidos reconhecimentos podem ser encontrados na seção de agradecimentos desta dissertação. Os resultados desta segunda parte estão contidos nesta dissertação e são apresentados na seção a seguir.

5.4 MÉTRICAS

Para avaliar a proposta foram aplicadas cinco métricas, todas apresentadas ao longo deste trabalho. O processo de encontrar os melhores parâmetros descrito na seção anterior utilizou da MAE para decidir qual o melhor conjunto de parâmetros, tal métrica foi apresentada na Seção 2.8.2 como uma forma de medir o erro na predição dos sistemas de recomendação. A métrica de erro absoluto foi usada, devido a verificar se o valor predito para a nota é semelhante ao valor atribuído pelo usuário, assim quanto menor o erro da MAE mais assertivo o sistema será.

Para entender os resultados olhando para a disposição do ranque, foram utilizadas

[6]https://sdumont.lncc.br/alloc.php?pg=alloc#
[7]https://www.top500.org/lists/top500/list/2020/11/?page=3
[8]https://sdumont.lncc.br/support_manual.php?pg=support#5

as métricas Mean Average Precision (MAP) e Mean Reciprocal Rank (MRR) apresentadas na Seção 2.8.3. Ambas as métricas consideram como itens relevantes na lista de recomendação o subconjunto de itens esperados que foi introduzido ao MID_u durante a divisão dos dados para validação, ou seja, ao avaliar a lista de recomendação as métricas verificam quantos são e em qual posição estão os itens esperados.

Esta dissertação baseia-se no desenvolvimento de um sistema de recomendação que trabalhe além da precisão/relevância. Para isto, são necessárias métricas que avaliem a partir de outras visões os resultados. Assim, as métricas de erro de calibragem foram utilizadas para verificar o grau de calibragem/descalibragem do sistema. Para isso, são utilizadas a Mean Average Calibration Error (MACE) e a Mean Rank Miss Calibration (MRMC), ambas as métricas foram apresentadas na Seção 4.6.

Como é possível observar na Figura 5.6, as combinações de sistema totalizam 1404, sendo 702 combinações para cada base de dados. Assim, para poder afirmar qual é a combinação de sistema que obtém os melhores desempenhos, foram utilizados os coeficientes decisórios apresentados na Seção 4.7.

5.5 RESULTADOS

Nesta seção, são apresentados os resultados obtidos a partir da execução experimental das combinações do sistema proposto. A organização dos resultados é dada como: 4 subseções correspondendo a uma métrica de avaliação cada e 2 subseções de cruzamento dos resultados. Em cada subseção as duas bases de dados são avaliadas de forma independente.

Para cada métrica e base de dados uma figura é apresentada composta de 9 gráficos, correspondente aos 9 algoritmos recomendadores apresentados na Seção 5.3. Cada gráfico é composto de 6 linhas e 13 pontos no eixo x, sendo cada linha uma combinação do balanceamento com uma medida de divergência ($2 \cdot 3 = 6$), sendo que cada ponto do eixo x é referente a um peso do balanceamento (11 constantes e 2 personalizados). O valor obtido pela métrica encontra-se no eixo y de cada gráfico.

Para os cruzamentos de resultados, têm-se que cada figura é formada por dois gráficos, sendo cada um referente a uma combinação do balanceamento com uma medida de divergência. Cada gráfico é composto de 7 linhas que representam cada um dos recomendadores (retirando os algoritmos Popularidade e Melhor nota) e 13 pontos referentes ao peso do balanceamento. Os valores do eixo x e y são os resultados obtidos nas métricas, sendo o valor de x referente à métrica Mean Average Precision (MAP) e o de y referente a uma das métricas Mean Average Calibration Error (MACE) ou Mean Rank Miss Calibration (MRMC). Ademais, cada figura é acompanhada de uma tabela contando os coeficientes decisórios.

5.5.1 MAP

O resultados do MAP são apresentados nas Figuras 5.7 e 5.8, que são respectivamente os resultados do *Movielens* e One Million Songs (OMS).

5.5.1.1 Movielens Os resultados obtidos através da métrica MAP no *Movilens* (Figura 5.7) mostram que os recomendadores possuem diferentes tipos de comportamentos. Os algoritmos com as melhores performances na métrica são Popularidade e Melhor nota, ambos são recomendadores tradicionais e não personalizados. Este resultado é justificável, desde que tome-se como diretriz que a base de dados é composta de uma cauda-longa e os itens mais populares estão em maioria nos modelos dos usuários. Com essa diretriz, quando a partição é realizada e os dados de teste são gerados estes itens integram a maioria das listas de recomendação. Assim, nesta dissertação, este dois algoritmos são tratados como exceções durante a análise. Os estudos de Abdollahpouri et al. (2019a, 2020) e Lin et al. (2020) debatem o tópico sobre popularidade no contexto de calibragem.

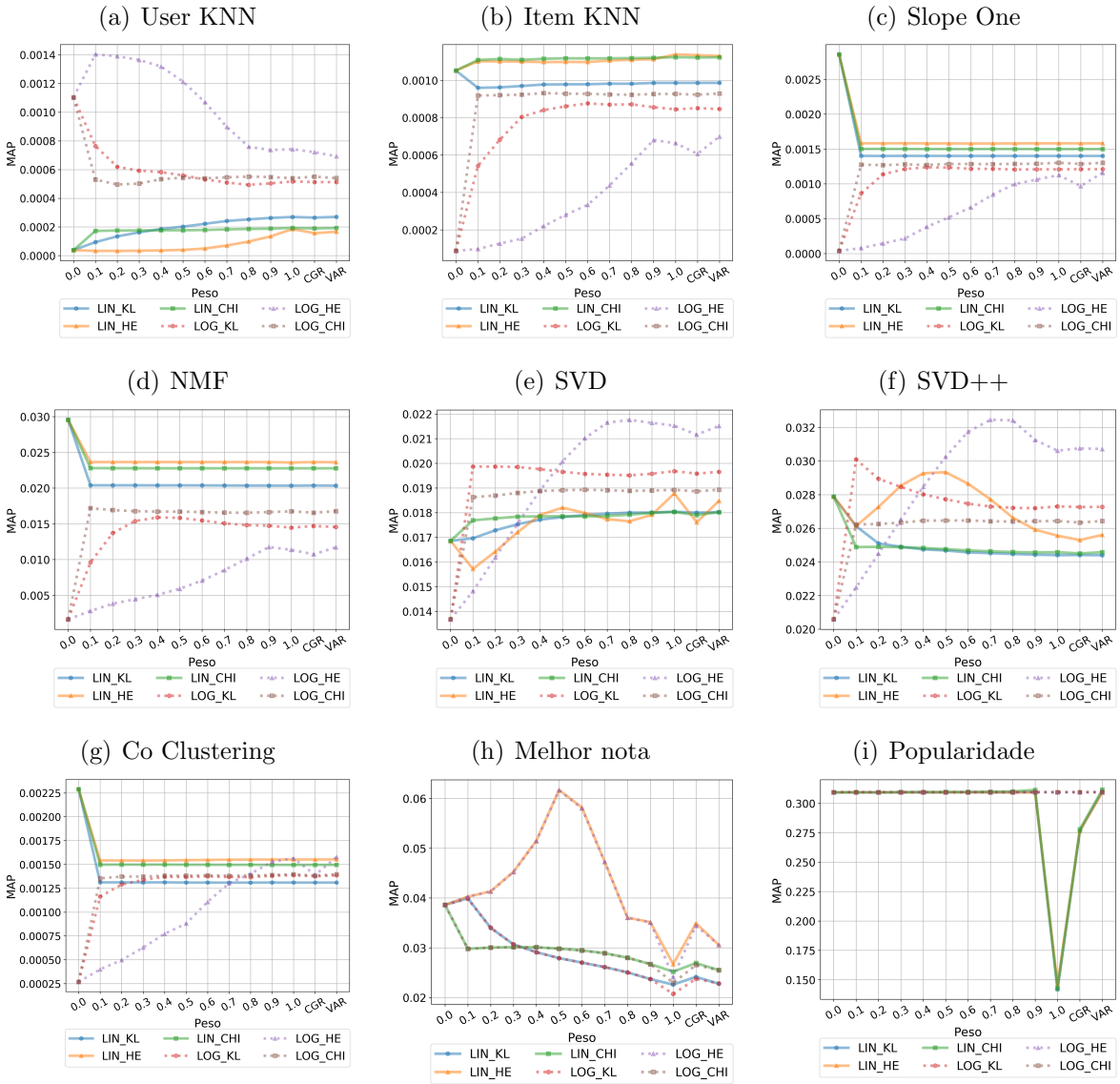
Para além, os melhores resultados, sem considerar as exceções, são dos recomendadores Singular Value Decomposition Plus Plus (SVD++), Non-negative Matrix Factorization (NMF) e Singular Value Decomposition (SVD), respectivamente, todos pertencentes ao grupo de fatoração de matrizes. Os algoritmos Slope One e o Co Clustering obtêm resultados estatisticamente similares, assim como o comportamento entre eles também possui similaridade. Nos últimos lugares com os menores valores na métrica MAP estão os recomendadores User-KNN e Item-KNN, respectivamente. Os algoritmos SVD++, SVD e Item-KNN possuem aumento em seus desempenhos na métrica quando comparados aos seus pontos sem a aplicação do pós-processamento ($\lambda = 0.0$). Os outros recomendadores obtêm uma variação de acréscimo e decréscimo de desempenho, dependendo da formulação do balanceamento utilizado.

Quando o resultado é observado a partir da perspectiva da formulação do balanceamento (linear LIN ou logarítmico LOG), pode-se concluir que, dependendo do recomendador, o comportamento vem a ser diferente. Para os algoritmos Popularidade e Melhor nota, ambas as formulações de balanceamento obtêm resultados semelhantes, com duas diferenças nos pesos $\lambda = 1.0$ e CGR, onde o balanceamento logarítmico (LOG) obteve melhores resultados.

Para os algoritmos SVD++, SVD e User-KNN, o LOG obtém a maioria dos melhores resultados juntamente com a medida Hellinger (LOG_HE). Para os algoritmos NMF, Co Clustering, Slope One e Item-KNN, o balanceamento linear (LIN) obtém os melhores resultados, com uma breve exceção no Co Clustering, onde o LOG_HE chega a obter resultados estatisticamente similares com o LIN em todas as medidas de justiça. Para todos os recomendadores, excluindo as exceções (Popularidade e Melhor nota), o LIN e LOG obtêm diferentes resultados com o peso $\lambda = 0.0$. Isto acontece devido ao viés do usuário contido no LOG, apresentado na Equação 4.22, fazer-se sempre influente na lista de recomendação. Para o algoritmo User-KNN, o viés incrementa a performance quando comparado com o mesmo peso no LIN. Para os algoritmos SVD++ e SVD, o viés melhora o desempenho conforme o peso do balanceamento tende a dar mais importância à divergência ($\lambda > 0.0$).

O peso do balanceamento mostra que há acréscimo e decréscimo do valor do MAP dependendo do recomendador, medida de justiça e formulação do balanceamento utilizadas. Para o algoritmo Popularidade, os pesos constantes obtêm os mesmos resultados, com uma exceção no valor do peso $\lambda = 1.0$ e no personalizado CGR, no qual há um decréscimo no desempenho do LIN. O LOG juntamente com o peso personalizado VAR e todas as

Figura 5.7 Resultados dos recomendadores na base do *Movielens*, que é avaliado com a métrica MAP.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

medidas de justiça mantêm o melhor resultado no recomendador Popularidade. Para o algoritmo Melhor nota, o peso constante produz um acréscimo nos resultados, entretanto após o pico de melhora um decréscimo acontece. O peso personalizado segue o mesmo comportamento no recomendador.

Para o algoritmo SVD++, o peso constante produz um acréscimo nos resultados para todas as medidas de justiça juntamente com a formulação do balanceamento logarítmico (*LOG*). O *LIN* produz um acréscimo no desempenho juntamente com a medida Hellinger *LIN_HE*. Para todos os resultados, os pesos constantes e personalizados obtiveram em alguns momentos o mesmo desempenho, com exceção do valor $\lambda = 0.0$. Para o SVD,

o peso constante possui um acréscimo no resultado do MAP e o personalizado VAR segue o comportamento, obtendo um resultado igualmente melhor. Todos os melhores resultados do recomendador são obtidos através do uso da formulação logarítmica proposta nesta dissertação. O incremento de desempenhos dos algoritmos SVD++ e SVD, quando usado o LOG, indica que a formulação proposta produz uma melhoria ao longo da lista de recomendação. Para o algoritmo Co Clustering, o peso personalizado VAR com o LOG_HE obtém uma melhora nos resultados e chega a valores estatisticamente similares aos melhores desempenhos do recomendador no MAP, que é propiciado pelo LIN. Para os outros algoritmos, o peso constante e personalizado obtém resultados estatisticamente similares ou ao menos um comportamento similar, onde ambos melhoram ou pioram o desempenho e comportamento do recomendador.

Quando o resultado é verificado a partir das medidas de justiça, é possível observar que o Hellinger obtém os melhores resultados do MAP na base com o LOG nos recomendadores SVD++ e SVD. Para o algoritmo User-KNN, o HE obtém o melhor resultado do recomendador. Para estes três algoritmos debatidos, o segundo e terceiro lugares são ocupados pelo LOG_KL e pelo LOG_CHI, respectivamente, quando observados a partir da perspectiva das medidas de justiça, indicando que a proposta de usar o χ^2 produz melhora nas performances dos recomendadores. Para os algoritmos Co clustering, NMF, Slope One e Item-KNN, o LIN_HE e LIN_CHI obtêm estatisticamente resultados similares, que por vez são os melhores desempenhos dos algoritmos.

O trabalho de Steck (2018) e Kaya e Bridge (2019) apontam que a calibragem causa uma diminuição do valor da precisão ao longo da lista, o que é confirmado em parte pelos resultados. Há combinações de sistema aos quais a calibragem causa um efeito positivo na precisão e em alguns casos quase dobrando o seu valor de desempenho. Todos os estudos do estado-da-arte previamente citados utilizam do peso constante no balanceamento linear e, como já abordado, este não é tão justo com os usuários, assim a nossa proposta de encontrar o peso personalizado para cada usuário obtém uma resposta positiva na precisão, indicando a possibilidade do uso ao longo de novos estudos.

5.5.1.2 OMS O MAP no OMS (Figura 5.8) mostra que os recomendadores possuem diferentes tipos de comportamentos, assim como na base do *Movielens*. O maior resultado é obtido através do algoritmo Popularidade, devido à particularidade da base de dados conter alguns itens que quando recomendados agradam uma maioria, devido a serem populares, assim como debatido na base do *Movielens*. Assim, o algoritmo Popularidade também é considerado uma exceção na análise da base OMS. O algoritmo Melhor nota não acompanha o desempenho do recomendador Popularidade nesta base, indicando que ser popular é mais influente do que ser ouvido muitas vezes individualmente.

Para além das exceções, o melhor resultado ao considerar a análise por algoritmo fica com o Item-KNN. O segundo e terceiro lugar ficam com os fatoradores de matriz SVD++ e SVD. O menor desempenho fica com o recomendador Co Clustering. Todos os algoritmos possuem o comportamento de acréscimo no desempenho ou manutenção do melhor resultado, com exceção do Item-KNN que apesar de obter o maior desempenho, sem considerar a exceção, também obtém um comportamento de decréscimo ou manutenção na sua performance, sem possuir assim um ganho real frente a métrica.

Figura 5.8 Resultados dos recomendadores usando o OMS avaliado com a métrica MAP.

Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Quando o resultado é observado a partir da perspectiva da formulação do balanceamento (linear LIN ou logarítmico LOG), pode-se concluir que, dependendo do recomendador, o comportamento vem a ser diferente, assim como na base do *Movielens*. No algoritmo Popularidade, ambas as formulações de balanceamento obtêm resultados semelhantes, com diferenças nos pesos $\lambda = 1.0$ e CGR, onde o balanceamento logarítmico (LOG) obteve melhores resultados, assim como na base do *Movielens*.

No algoritmo Item-KNN, o LIN obtém a maioria dos melhores resultados juntamente com a medida Hellinger (LIN_HE). No algoritmo SVD++, os melhores resultados são obtidos com o LIN_KL, entretanto para o peso $\lambda = 1.0$, o LOG_CHI acompanha o desempenho e obtém resultados estatisticamente similares. As outras formulações do

LOG chegam a obter melhores resultados que as formulações do LIN para o peso $\lambda = 1.0$. No SVD, os LIN_HE e LIN_CHI obtêm resultados e comportamentos similares, assim como também são os melhores resultados do recomendador, com exceção do peso $\lambda = 1.0$, onde o LOG_HE obtém resultado melhor, os LOG_HE e LOG_CHI também acompanham o desempenho e chegam a obter resultados estatisticamente similares aos LIN_HE e LIN_CHI.

Para a maioria dos recomendadores, assim como no *Movielens*, o LIN e LOG obtêm diferentes resultados com o peso $\lambda = 0.0$. Para o algoritmo Melhor nota, o LOG obtém a maioria dos melhores resultados, sendo que o algoritmo possui um constante acréscimo de performance, conforme o peso do balanceamento aumenta em direção à justiça ($\lambda > 0.0$). Este comportamento também pode ser encontrado no algoritmo Co Clustering.

O peso do balanceamento mostra que, na maioria de seus valores, há acréscimo no valor do MAP conforme o peso do balanceamento cresce ou é usado de forma personalizada. É possível perceber que o peso $\lambda = 1.0$ causa um efeito diferenciado na lista de recomendação. Como é possível verificar, todos os algoritmos apresentados na figura passaram por mudanças com o uso do peso, que em maioria obteve um acréscimo significativo em seus desempenhos. No algoritmo Popularidade, os pesos constantes obtêm os mesmos resultados para todas as combinações de balanceamento, com uma exceção no valor do peso $\lambda = 1.0$, no qual há um decréscimo no desempenho do LIN. O LOG mantém-se com o melhor resultado no recomendador.

Para o algoritmo Item-KNN, o LIN_KL com o valor personalizado VAR obtém um dos melhores desempenhos do sistema na base. Resultado este que é estatisticamente similar a alguns pesos constantes. Para o algoritmo SVD++, o melhor resultado é compartilhado por todos os pesos constantes e personalizados que, como abordado, é estatisticamente acompanhado pelo resultado do LOG_CHI com o peso $\lambda = 1.0$. A descrição do SVD é similar ao SVD++, que foi previamente apresentado, com o adendo de que o LOG_HE obtém o melhor resultado do SVD durante o pico do peso $\lambda = 1.0$. Os algoritmos User-KNN, Co Clustering e Melhor nota, possuem um comportamento de acréscimo no desempenho que, como é possível observar, influi em todas as linhas dos gráficos.

Quando o resultado é verificado a partir das medidas de justiça, é possível observar que, em valor absoluto, o HE obtém os melhores resultados para a base. Entretanto, para cada algoritmo as medidas de justiça que obtêm os melhores resultados são variadas. Para o algoritmo SVD++, o KL obtém a maioria dos melhores resultados. Para os algoritmos Item-KNN, NMF e SVD, o HE obtém maioria dos melhores resultados, sendo que em alguns casos, há resultados compartilhados, onde eles são estatisticamente similares. Para os algoritmos User-KNN, Slope One, NMF, SVD, Melhor nota e Popularidade, o CHI obtém maioria dos melhores resultados, em alguns casos há resultados compartilhados, onde eles são estatisticamente similares a outra medidas de justiça, demonstrando que a proposta possui uma potencialidade em seu uso.

O trabalho de Kaya e Bridge (2019) aponta que a calibragem causa um acréscimo do valor da precisão ao longo da lista, o que é confirmado pelos resultados apresentados acima. Conforme o valor do peso cresce ($\lambda > 0.0$), o valor da precisão aumenta. Resultado este encontrado apenas na base do OMS, tanto nesta dissertação quanto no trabalho de Kaya e Bridge (2019).

5.5.2 MRR

Os resultados obtidos através da métrica Mean Reciprocal Rank (MRR) são apresentados nas Figuras 5.9 e 5.10, que são respectivamente os resultados do *Movielens* e OMS.

5.5.2.1 Movielens Os resultados do *Movielens* a partir da métrica MRR (Figura 5.9) mostra que os recomendadores possuem diferentes tipos de comportamento. É possível observar que os resultados dos algoritmos no MRR seguem o comportamento apresentado no MAP. Essa conexão entre as duas métricas é esperada, devido a ambas trabalharem sobre o ranqueamento dos itens, como apresentado na Seção 2.8.3. Os melhores resultados são dos algoritmos Popularidade e Melhor nota, nesta ordem. Como previamente esclarecido, ambos os recomendadores são tratados como exceção em nossas análises, devido à condição das bases de dados usadas.

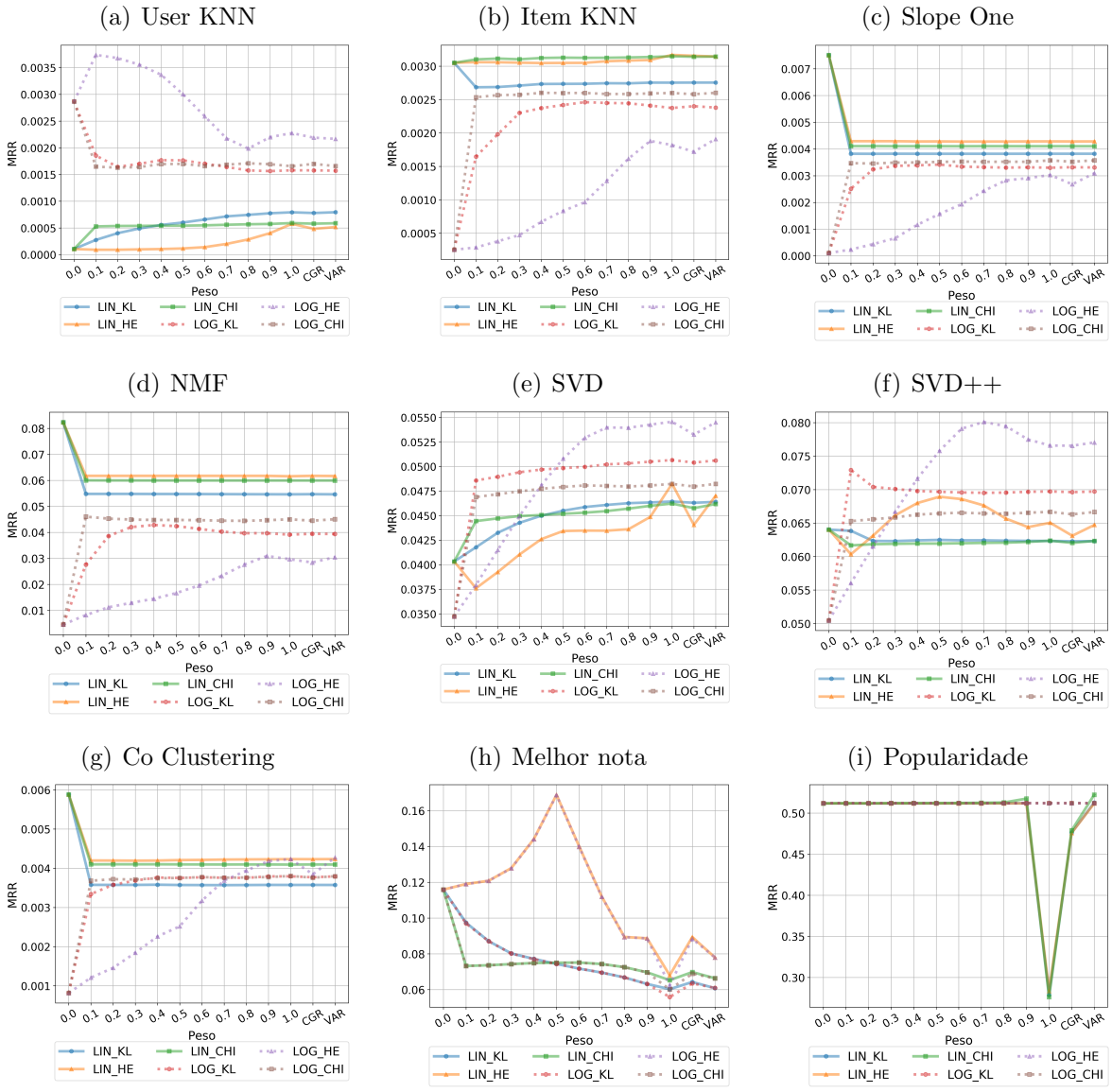
Além disso, os melhores resultados, sem considerar as exceções, e assim como na métrica MAP, são dos recomendadores SVD++, NMF e SVD, nesta ordem. O Slope One e o Co Clustering obtêm comportamentos similares. Nos últimos lugares, com os menores valores na métrica MRR estão os recomendadores User-KNN e Item-KNN, respectivamente. Os algoritmos SVD++, SVD e Item-KNN possuem aumento em seus desempenhos na métrica quando comparados aos seus pontos sem a aplicação do pós-processamento ($\lambda = 0.0$). Os outros recomendadores obtêm uma variação de acréscimo e decréscimo no desempenho, dependendo da formulação do balanceamento utilizado.

Quando o resultado é observado a partir da perspectiva da formulação do balanceamento (linear LIN ou logarítmico LOG), assim como na métrica MAP, pode-se concluir que, dependendo do recomendador, o comportamento vem a ser diferente. É observável que as formulações do LOG possuem similaridades entre si, assim como as formulações do LIN possuem similaridades entre elas. Para os algoritmos Popularidade e Melhor nota, ambas as formulações de balanceamento obtêm resultados semelhantes. Com diferenças no algoritmo Popularidade, nos pesos $\lambda = 1.0$ e CGR, onde o balanceamento logarítmico (LOG) obteve melhores resultados.

Para os algoritmos SVD++, SVD e User-KNN, o LOG obtém todos os melhores resultados juntamente com a medida Hellinger (LOG_HE), demonstrando o efeito positivo da formulação proposta sobre a adição do primeiro item relevante na lista. Para os algoritmos NMF, Co Clustering, Slope One e Item-KNN, o balanceamento linear (LIN) obtém os melhores resultados, com uma breve exceção no Co Clustering, onde o LOG_HE chega a obter resultados estatisticamente similares com o LIN em todas as medidas de justiça. Para todos os recomendadores, sem as exceções (Popularidade e Melhor nota), o LIN e LOG obtêm diferentes resultados com o peso $\lambda = 0.0$, assim como na métrica MAP, isto acontece devido ao viés do usuário contido no LOG fazer-se sempre influente na lista de recomendação. Para o algoritmo User-KNN, o viés incrementa a performance quando comparado com o mesmo peso no LIN. Para os algoritmos SVD++ e SVD, o viés melhora o desempenho conforme o peso do balanceamento tende a dar mais importância à divergência ($\lambda > 0.0$).

O peso do balanceamento mostra que há acréscimo e decréscimo do valor do MRR dependendo do recomendador, medida de justiça e formulação do balanceamento utilizadas.

Figura 5.9 Resultados dos recomendadores usando o *Movielens* avaliado com a métrica MRR.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Para o algoritmo Popularidade, os pesos constantes até $\lambda = 0.8$ obtêm os mesmos resultados. Para os pesos $\lambda = 1.0$ e no personalizado CGR, um decréscimo no desempenho do LIN ocorre, entretanto a performance é mantida pelo LOG_CHI. Para o algoritmo Melhor nota, o peso constante produz um acréscimo nos resultados, entretanto após o pico de melhora um decréscimo acontece.

Para o algoritmo SVD++, o peso constante possui um acrescimento nos resultados juntamente com a medida Hellinger para ambas as formulações do balanceamento (LIN e LOG). Para o SVD, o peso constante possui um acréscimo no resultado do MRR e o personalizado VAR segue o comportamento, obtendo um resultado estatisticamente similar, sendo estes os melhores resultados. Para o algoritmo Co Clustering, o peso

personalizado VAR com o LOG_HE obtém uma melhora nos resultados e chega a valores estatisticamente similares aos melhores desempenhos do recomendador no MRR. Para os outros algoritmos, o peso constante e personalizado obtém resultados estatisticamente similares ou ao menos um comportamento similar, onde ambos melhoram ou pioram o desempenho e comportamento do recomendador.

Quando o resultado é verificado a partir das medidas de justiça, é possível observar que o Hellinger obtém a maioria dos melhores resultados no MRR com o LOG para os recomendadores SVD++ e SVD. Quando não o Hellinger, o melhor resultado é ocupado pelo KL. Para o algoritmo User-KNN, o HE obtém todos os melhores resultados do recomendador. Para estes três últimos algoritmos debatidos, o segundo e terceiro lugar são ocupados pelo LOG_KL e pelo LOG_CHI, respectivamente, ou seja, indicando que a proposta do balanceamento logarítmico (LOG) produz melhora nas performances dos recomendadores, quando avaliado a partir do primeiro item relevante na lista de recomendação. Para os algoritmos Co clustering, NMF, Slope One e Item-KNN, o LIN_HE e LIN_CHI obtêm estatisticamente resultados similares.

O trabalho de Steck (2018) aponta que a calibragem causa uma diminuição do valor do *recall* ao longo da lista, o que é confirmado em parte pelos resultados. Há combinações de sistema nos quais a calibragem causa um efeito positivo no ranqueamento e em alguns casos quase dobrando o seu valor de desempenho.

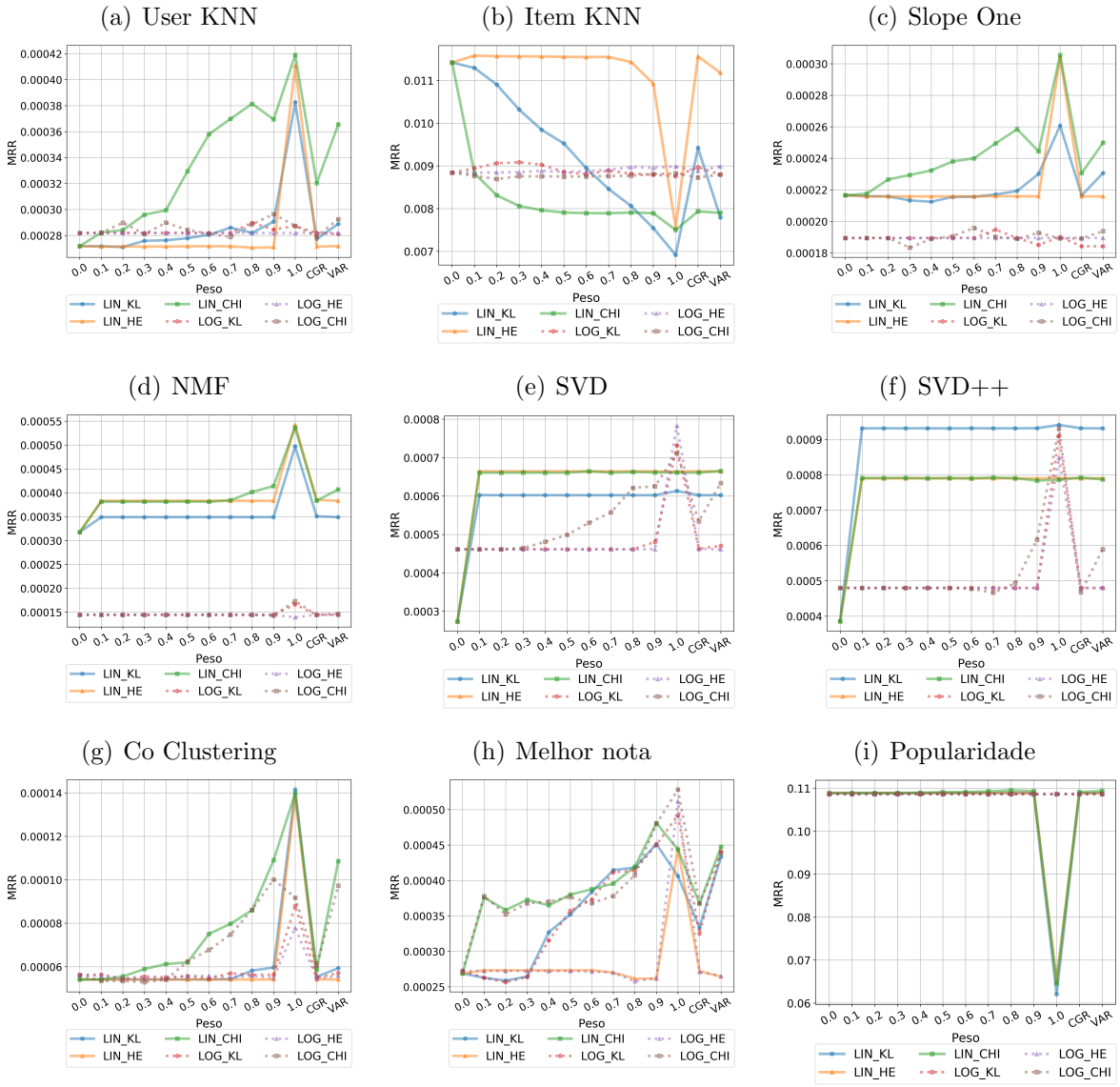
5.5.2.2 OMS O MRR no OMS (Figura 5.10) mostra que os recomendadores possuem diferentes tipos de comportamento entre si, assim como na métrica MAP. O melhor resultado é obtido através do algoritmo Popularidade, devido à particularidade da base de dados conter alguns itens que são populares e quando recomendados agradam uma maioria, assim como previamente debatido. O algoritmo Melhor nota não acompanha o Popularidade nesta base, assim como no MAP, indicando que ser popular é mais influente do que ser ouvido muitas vezes individualmente pelos usuários.

Para além dos algoritmos que são analisados como uma exceção, o melhor resultado ao considerar a análise por algoritmo fica com o Item-KNN, assim como analisado no MAP. Os algoritmos de fatoração de matrizes SVD++ e SVD seguem o Item-KNN ocupando as próximas posições quanto aos melhores resultados. O pior desempenho fica com o recomendador Co Clustering. Todos os algoritmos possuem o comportamento de acréscimo no desempenho ou manutenção do resultado, com exceção do Item-KNN que, apesar de obter o melhor desempenho, também obtém um comportamento de decréscimo na sua performance.

Quando o resultado é observado a partir da perspectiva da formulação do balanceamento (linear LIN ou logarítmico LOG), pode-se concluir que, dependendo do recomendador, o comportamento vem a ser diferente, assim como na métrica MAP. Para o algoritmo Popularidade, ambas as formulações de balanceamento obtêm resultados semelhantes entre si, com uma diferença no peso $\lambda = 1.0$, onde o balanceamento logarítmico (LOG_CHI) obteve o melhor resultado, assim como no *Movielens*.

Para o algoritmo Item-KNN, o LIN obtém a maioria dos melhores resultados juntamente com a medida Hellinger (LIN_HE). O algoritmo SVD++ obtém melhores resultados com o LIN_KL, entretanto para o peso $\lambda = 1.0$, o LOG_CHI acompanha o

Figura 5.10 Resultados dos recomendadores usando o OMS avaliado com a métrica MRR.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

desempenho e obtém resultados estatisticamente similares. Para o SVD, os LIN_HE e LIN_CHI obtêm resultados e comportamento similares entre si, assim como também são os melhores resultados do recomendador, com exceção do peso $\lambda = 1.0$, onde o LOG com todas as medidas de justiça obtém resultado melhor. Para a maioria dos recomendadores, assim como no *Movielens* e na métrica MAP, o LIN e LOG obtêm diferentes resultados com o peso $\lambda = 0.0$.

O peso do balanceamento mostra que em maioria há acréscimo no valor do MRR conforme o peso do balanceamento cresce ou é usado de forma personalizada. É possível perceber que o peso $\lambda = 1.0$ causa um efeito imediato na lista de recomendação, assim como na métrica MAP. Como é possível verificar, todos os algoritmos apresentados na

figura passaram por mudanças com o uso do peso. Para o algoritmo Popularidade, o peso constante obtém os mesmos resultados, com uma diferença no valor do peso $\lambda = 1.0$, no qual há um decréscimo no desempenho.

Para o algoritmo Item-KNN, o LIN_KL com o valor personalizado VAR obtém um dos melhores desempenhos do sistema na base. Resultado este que é estatisticamente similar a alguns pesos constantes. Para o algoritmo SVD++, o melhor resultado é compartilhado por todos os pesos constantes e personalizados que, como abordado, é estatisticamente acompanhado pelo resultado do LOG_CHI com o peso $\lambda = 1.0$. A descrição do SVD é similar ao SVD++, que foi previamente apresentado, com o adendo de que o LOG obtém os melhores resultados do SVD durante o pico do peso $\lambda = 1.0$. Os algoritmos User-KNN, Co Clustering e Melhor nota, possuem um comportamento de acréscimo constante no desempenho que, como é possível observar, influi em todas as linhas dos gráficos. Todos os comportamentos dos pesos no OMS quando analisados a partir do MRR são similares à análise da mesma base a partir da métrica MAP.

Quando o resultado é verificado a partir das medidas de justiça, é possível observar que, em valor absoluto, o HE obtém os melhores resultados para a base. Entretanto, para cada algoritmo as medidas de justiça que obtêm os melhores resultados são variadas, comportamento similar ao *Movielens*. Para o algoritmo SVD++, o KL obtém a maioria dos melhores resultados. Para os algoritmos Item-KNN, NMF e SVD, o HE obtém maioria dos melhores resultados, em alguns casos há resultados compartilhados, onde eles são estatisticamente similares. Para os algoritmos User-KNN, Slope One, NMF, SVD, Co Clustering, Melhor nota e Popularidade, o CHI obtém maioria dos melhores resultados, em alguns casos há resultados compartilhados, onde eles são estatisticamente similares a outra medidas de justiça.

Os trabalhos do estado-da-arte não abordam as avaliações a partir das métricas MAP e MRR. Kaya e Bridge (2019) realizam uma análise a partir da precisão, entretanto a precisão é uma métrica onde a análise é sobre uma determinada posição da lista, sendo as métricas usadas nesta dissertação uma análise de todas as posições da lista. Assim, as análises realizadas acima não podem ser comparadas ao estado-da-arte, devido a seu ineditismo. A seguir, os resultados são analisados a partir das métricas que fazem parte da proposta desta dissertação.

5.5.3 MACE

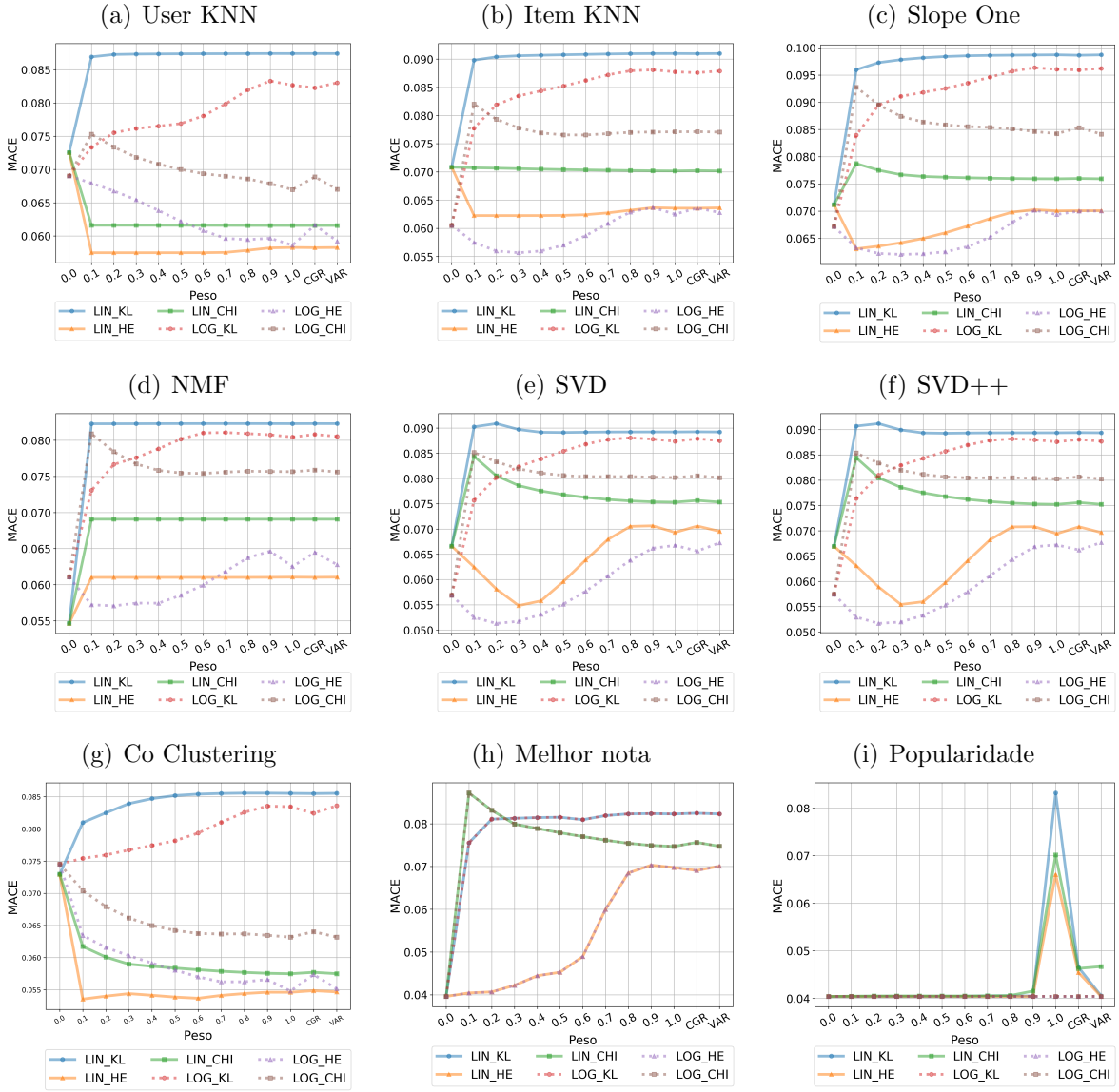
Os resultados obtidos através da métrica MACE são apresentados nas Figuras 5.11 e 5.12, que são respectivamente os resultados do *Movielens* e OMS.

5.5.3.1 Movielens Os resultados do *Movielens* a partir da métrica MACE (Figura 5.11) mostra que os recomendadores possuem comportamentos similares, apesar de haver exceções. É possível observar que os valores do MACE (eixo y), possuem um intervalo similar. O menor valor na métrica, que é o melhor, encontra-se com o recomendador Popularidade, seguido do algoritmo Melhor nota. Como alertado esses dois recomendadores fazem parte de uma exceção, tendo que ao recomendar os itens mais populares ou com melhor média de nota entre os usuários, o gênero de maior dominância é favorecido pelo

próprio funcionamento do algoritmo. Durante a apresentação do MACE na Seção 4.6, é debatida esta situação sobre a métrica, que ao recomendar o gênero dominante o valor pode ser baixo e ao recomendar o gênero menos dominante o valor pode ser alto.

Após entender as exceções, tem-se que ocupando os melhores resultados estão os fatoradores de matrizes SVD++ e SVD. Em seguida estão os algoritmos Co Clustering, Item-KNN, User-KNN e Slope One. Todos estes recomendadores possuem um comportamento similar. As melhores posições quando analisadas a partir dos recomendadores são ocupadas pelos mesmos algoritmos na mesma ordem que os obtidos no MAP e MRR, indicando uma possível relação entre as métricas.

Figura 5.11 Resultados dos recomendadores usando o *Movielens* avaliado com a métrica MACE.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Ao observar os resultados do MACE a partir das formulações de balanceamento (LIN e LOG), tem-se que a maioria dos piores resultados são obtidos pela LIN, entretanto alguns resultados positivos também. A formulação linear obtém resultados distintos a depender da medida de justiça utilizada. No algoritmo Melhor nota, ambas as formulações obtêm resultados similares. Para o algoritmo Popularidade há apenas diferença para os valores do peso $\lambda = 1.0$ e CGR, como debatido no MAP e MRR, demonstrando que pode haver uma ligação, quando o valor do MAP e MRR diminuiu o valor do MACE aumentou.

Os melhores resultados do balanceamento entre os algoritmos, sem as exceções (Popularidade e Melhor nota), são do LOG nos recomendadores SVD++ e SVD, reforçando os indícios, sendo que para as métricas anteriores a mesma formulação também obteve os melhores desempenhos. Para os Item-KNN, NMF e Slope One a mesma formulação também obteve os melhores resultados nos recomendadores.

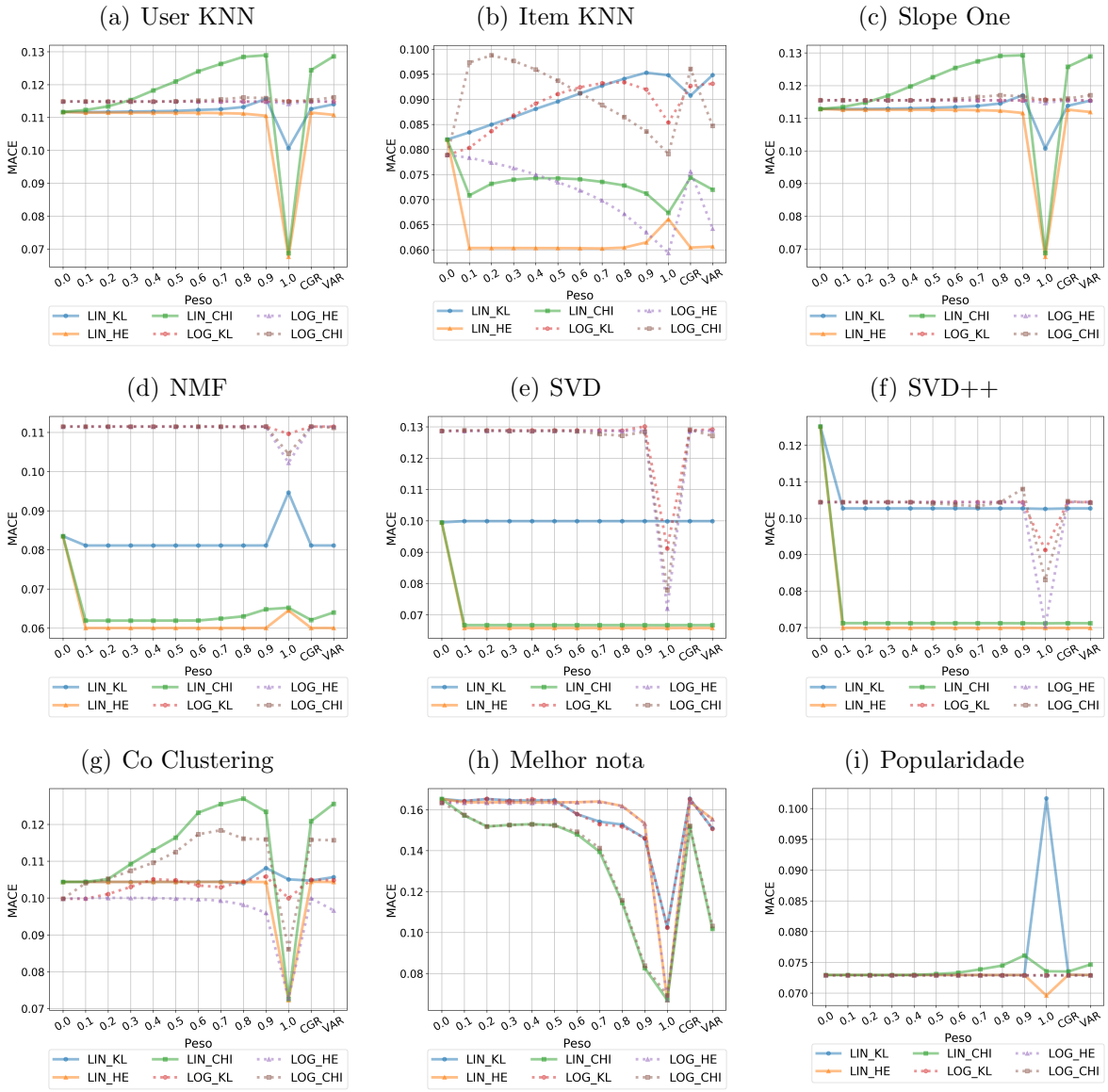
A aplicação do pós-processamento gera resultados imediatos com valores pequenos para o λ. Como é possível observar na figura, quando o peso do balanceamento começa a dar importância à calibragem a lista de recomendação é afetada de forma positiva e negativa, assim como apresentado nas métricas anteriores. Para o algoritmo Popularidade o peso $\lambda = 1.0$ e CGR produz resultados diferentes para a formulação LIN, afetando negativamente a formulação, entretanto a LOG obtém os melhores resultados. Para os algoritmos SVD++ e SVD, os pesos constantes obtiveram os melhores resultados. Os pesos personalizados ou mantiveram os resultados já obtidos pelos pesos constantes ou continuaram o comportamento de melhora nos resultados.

As medidas de divergência produzem diferentes resultados dependendo da formulação do balanceamento.Todos os piores resultados são produzidos pelo Kullback-Leibler (KL) e maioria deles juntamente com o balanceamento LIN. A medida HE obtém os melhores resultados, sem considerar as exceções. Percebe-se que o Hellinger melhora o desempenho em alguns casos e em outros mantém os valores abaixo das outras medidas, isto independente da formulação do balanceamento.

5.5.3.2 OMS Os resultados do OMS a partir da métrica MACE (Figura 5.12) mostram que os recomendadores possuem alguns comportamentos similares. Os melhores resultados são dos algoritmos NMF e Item-KNN, sendo que ambos possuem um diferente tipo de comportamento. Os algoritmos NMF, SVD e SVD++ possuem um comportamento similar em seus resultados. Para todos os fatoradores de matriz houve em maioria uma redução dos valores do MACE. Os algoritmos User-KNN, Slope One e Co Clustering possuem um comportamento similar, incluindo o intervalo de resultados. Os algoritmos Item-KNN, Melhor nota e popularidade possuem comportamentos singulares.

Ao observar os resultados a partir das formulações de balanceamento (LIN e LOG), vê-se que os melhores resultados estão com o LIN. Para o NMF, SVD e SVD++ o comportamento dos balanceamentos são similares. No algoritmo Melhor nota os resultados para ambas as formulações foram similares. Para o Popularidade também, entretanto a formulação LOG não é afetada pelo peso $\lambda = 1.0$. Para todos os fatoradores de matriz o que mais influi a lista é a formulação do balanceamento, havendo uma visível diferença nos resultados. Para todos os outros algoritmos o que influi no desempenho é a medida de

Figura 5.12 Resultados dos recomendadores usando o OMS avaliado com a métrica MACE.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

divergência usada, como é possível verificar na figura, que as medidas possuem resultados similares.

Os diversos pesos dos balanceamentos produzem diferentes comportamentos. Em especial o peso constante $\lambda = 1.0$, como debatido nas métricas anteriores, produz um resultado diferente em todos os algoritmos. É possível observar que quando o peso é aplicado os resultados são alterados, em alguns algoritmos a diminuição do valor do MACE é notória. Os pesos personalizados produzem melhoras no desempenho ou reproduzem comportamentos previamente obtidos com os pesos constantes.

Os melhores resultados obtidos a partir da observação pelas medidas de divergência

indicam que o LIN_HE obtém os melhores resultados, seguido do LIN_CHI. Para o algoritmo Co Clustering, o Hellinger obtém bons resultados juntamente com a formulação LOG. A medida de divergência KL, que é utilizada no estado-da-arte, não obteve bons resultados com nenhuma formulação do balanceamento.

5.5.4 MRMC

Os resultados obtidos através da métrica MRMC são apresentados nas Figuras 5.13 e 5.14, que são respectivamente os resultados do *Movielens* e OMS.

5.5.4.1 Movielens Os resultados do *Movielens* a partir da métrica MRMC (Figura 5.13) mostram que a maioria dos recomendadores possuem comportamentos similares, com algumas exceções. É possível observar que os valores do MRMC (eixo y), possuem uma grande amplitude quanto ao intervalo, entretanto um comportamento similar entre os algoritmos pode ser notado. Os resultados indicam que com o uso do pós-processamento, todos os algoritmos são afetados, diminuindo assim o nível de descalibragem e propiciando uma lista de recomendação que seja composta de todos os gêneros pertencentes às preferências do usuário.

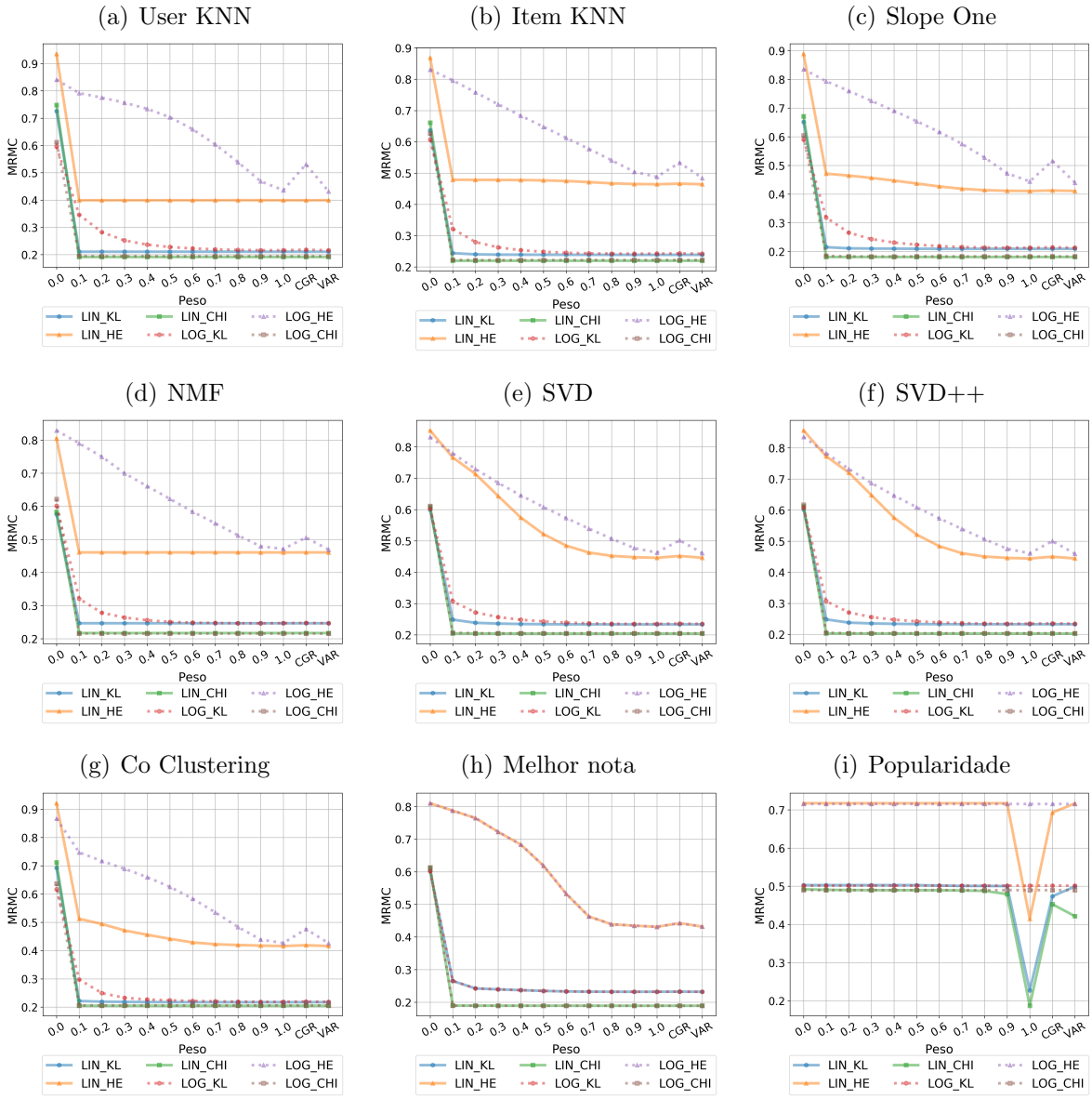
Todos os algoritmos, com exceção do Popularidade, obtêm em maioria resultados similares, quando observados os melhores desempenhos. Como alertado o recomendador Popularidade faz parte de uma análise diferente, devido a sua amplificação da popularidade de alguns itens, que como demonstrado nos resultados, reflete em uma lista de recomendação com o menor grau de calibragem, ou seja, não condiz com todas as preferências do usuário. A métrica permite detectar dois tipos de comportamentos entre os algoritmos: 1) manter o mesmo nível de descalibragem, como é possível ver no recomendador Popularidade; ou 2) uma imediata redução da descalibragem. Em alguns algoritmos, dependendo da combinação do balanceamento, medida e peso do balanceamento, a curva pode ser mais ou menos suave ao longo dos pesos.

Ao observar as formulações do balanceamento é possível perceber que, para o algoritmo Melhor nota, ambos os balanceamentos obtêm resultados similares. No algoritmo Popularidade, é possível observar o mesmo, entretanto, para os valores do peso $\lambda = 1.0$ e CGR os resultados são diferentes. Comportamentos estes que foram observados previamente em todas as métricas apresentadas, indicando que o MRMC, uma das propostas desta dissertação, captura as nuances que outras métricas conseguem observar.

Os melhores desempenhos em todos os algoritmos são similares. Os resultados indicam que a medida de divergência usada para gerar as recomendações é mais importante que o balanceamento, considerando que na figura as combinações: LOG e LIN junto com o Hellinger obtêm os piores resultados, mesmo havendo em alguns casos diferenças em seus valores; LOG e LIN junto com o KL obtêm a maioria dos resultados intermediários; e o LOG e LIN junto com o χ^2, proposta desta dissertação, obtêm os melhores desempenhos.

Conforme o valor do peso do balanceamento tende em direção a prover mais justiça na lista de recomendação, os valores do MRMC diminuem ou mantêm o desempenho. É possível observar que os pesos afetam a criação da lista, entretanto, a aplicação do pós-processamento com qualquer valor no peso (constante ou personalizado), produz uma

Figura 5.13 Resultados dos recomendadores usando o *Movielens* avaliado com a métrica MRMC.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

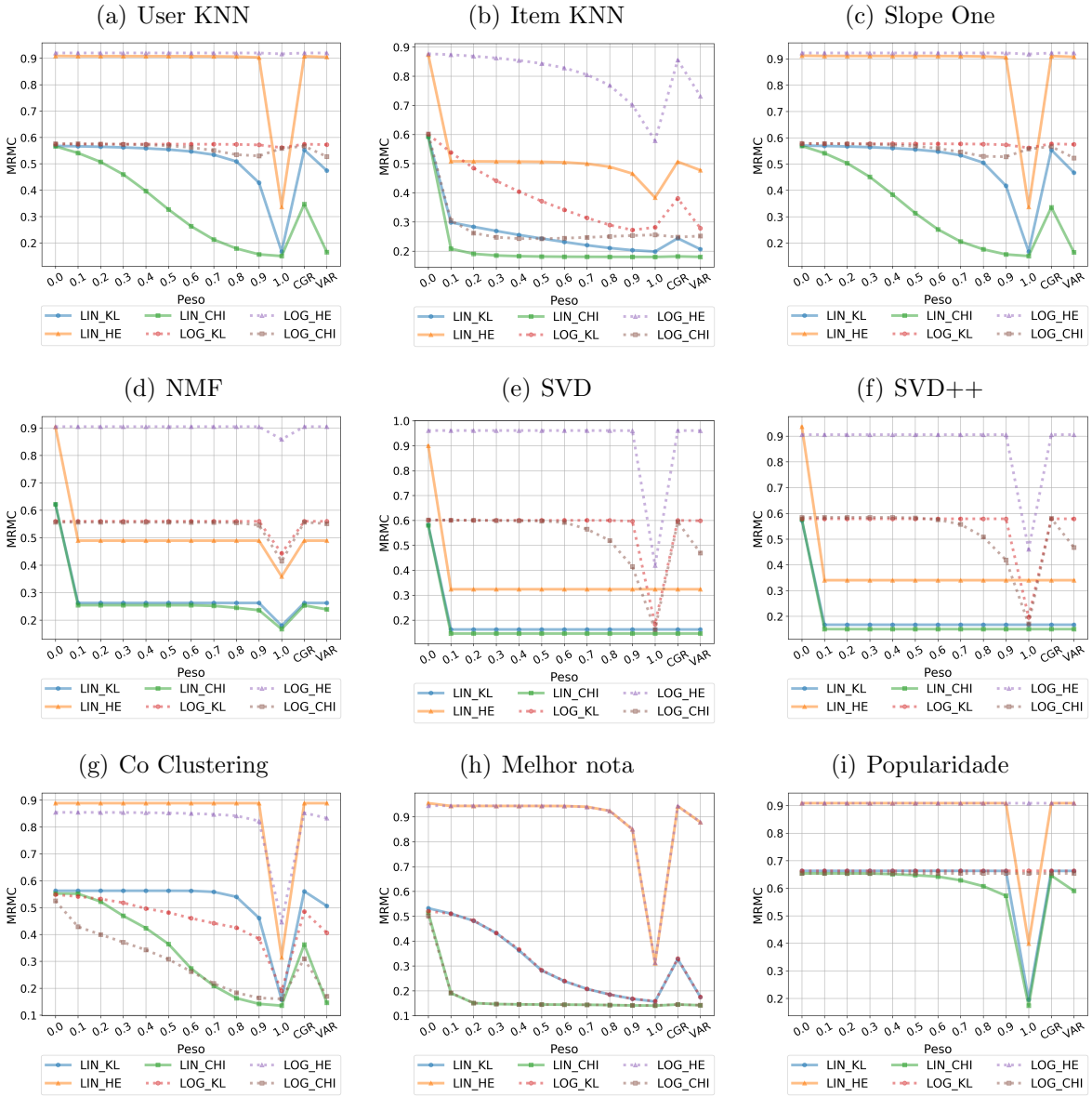
melhora significativa na maioria dos recomendadores.

5.5.4.2 OMS Os resultados do OMS a partir da métrica MRMC (Figura 5.14) mostram alguns comportamentos similares entre os algoritmos. Os recomendadores User-KNN, Slope One e Co Clustering possuem comportamentos similares. Os recomendadores NMF, SVD e SVD++ formam um outro tipo de comportamento. O Item-KNN, Melhor nota e Popularidade possuem comportamentos únicos. Entretanto, é possível observar um comportamento comum a todos os recomendadores, ou existe uma melhora no

desempenho do algoritmo ou o desempenho é mantido.

Similar ao *Movielens*, o Popularidade não obtém o melhor desempenho, reforçando que o recomendador não explora todas as preferências do usuário, apesar do algoritmo ter obtido bons desempenhos no MAP. Isto demonstra que recomendar os itens mais populares causa altos índices de descalibragem na lista de recomendação. Os melhores valores de desempenho são compartilhados entre diversos recomendadores, sendo possível verificar que todos os algoritmos obtêm melhoras do desempenho e o menor valor para todos é similar (≤ 0.2). Assim como demonstrado no *Movielens*, a aplicação do pós-processamento produz em todos os recomendadores um efeito positivo imediato.

Figura 5.14 Resultados dos recomendadores usando o OMS avaliado com a métrica MRMC.



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Ao observar as formulações do balanceamento torna-se possível afirmar que para a base de dados o LIN obtém os melhores resultados. Na maioria dos recomendadores o melhor resultado é obtido pela combinação LIN_CHI, demonstrando que a proposta de utilizar o χ^2 como medida de justiça pode gerar resultados promissores. Em outras métricas, por exemplo no MRR, a proposta do χ^2 também obteve melhora no desempenho. Em diversos momentos outras formulações acompanham e até ultrapassam o χ^2. Para o algoritmo Melhor nota, ambas as formulações obtêm os mesmos resultados. Similar ao *Movielens*, o LOG combinado com o Hellinger obtém os menores desempenhos. O LIN_HE na base de dados apresentou melhora no desempenho em outras métricas, como por exemplo na MAP, entretanto na MRMC não apresentou os mesmos desempenhos. Para o grupo de fatoradores de matriz o mais influente no resultado é a formulação do balanceamento e para todos os outros algoritmos o mais influente nos resultados é a medida de divergência.

A métrica MRMC capturou a nuance do peso $\lambda = 1.0$, que em todos os recomendadores propiciou melhora no desempenho, reduzindo em todos os casos a descalibragem. É possível observar, também, que os pesos constantes e personalizados ou mantêm o desempenho ou melhoram, em nenhum caso há piora no desempenho quando comparado com o valor $\lambda = 0.0$, que significa o não uso do pós-processamento, no balanceamento LIN.

5.5.5 MACEXMAP

Nesta seção é debatido o cruzamento dos resultados, buscando avaliar em cada base de dados a conexão entre as métricas MACE e MAP. As Figuras 5.15, 5.16 e 5.17 apresentam os cruzamentos para a base de dados do *Movielens*. As Tabelas 5.6, 5.7 e 5.8 apresentam o coeficiente de erro na calibragem para o *Movielens*. As Figuras 5.18, 5.19 e 5.20 apresentam os cruzamentos para a base de dados do OMS. As Tabelas 5.9, 5.10 e 5.11 apresentam o coeficiente de erro na calibragem para o OMS.

Para realizar a análise nas figuras torna-se necessário retirar os algoritmos Popularidade e Melhor nota. Como previamente debatido, ambos os recomendadores possuem resultados diferentes dos outros algoritmos, sendo analisados como uma exceção durante todas as seções anteriores. Os recomendadores Popularidade e Melhor nota causam uma distorção nos dados, gerando figuras nas quais uma análise visual não é possível, devido a possuir um alto valor de precisão mas também possuir um alto erro na calibragem.

5.5.5.1 Movielens Os resultados obtidos com a base de dados do *Movielens* a partir do cruzamento das métricas MACE e MAP (Figuras 5.15, 5.16 e 5.17 e Tabelas 5.6, 5.7 e 5.8) confirmam as observações realizadas nas seções anteriores.

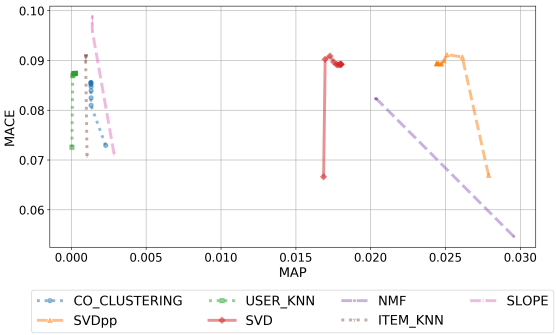
Tabela 5.6 *Movielens* - CCE - Resultados das formulações LIN e LOG com a medida de divergência KL.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	2.73	60.44	90.73	3.8	0.15	63.52	4.95	**3.52**	430.91
LOG	2.75	62.94	110.46	5.76	0.13	83.46	4.31	**3.05**	130.58

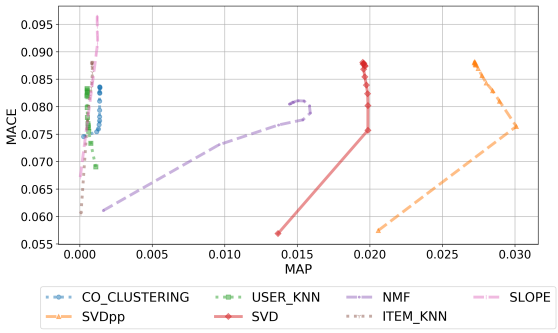
Fonte: Tabela elaborada pelo autor a partir dos resultados.

Figura 5.15 *Movielens* - MACE e MAP - Resultados das formulações *LIN* e *LOG* com a medida de divergência KL.

(a) *LIN* e Kullback-Leibler



(b) *LOG* e Kullback-Leibler



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

As Figuras 5.15(a) e (b) em conjunto com a Tabela 5.6 demonstram que, para o KL, a melhor formulação do balanceamento é a logarítmica (*LOG*), proposta desta dissertação. Esta combinação obteve os maiores valores do MAP e disputou os melhores valores do MACE. É possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre as duas métricas não são correlatadas. Resultados anteriores indicaram que alguns comportamentos podem ser observados por ambas as métricas, entretanto é possível afirmar que suas análises são diferentes. Ao observar o algoritmo recomendador é possível verificar que o melhor desempenho em relação ao MAP é do SVD++. O algoritmo SVD ocupa a segunda colocação entre os melhores desempenhos. O terceiro lugar é ocupado pelo NMF. Na formulação linear (*LIN*) o segundo lugar pertence ao NMF e o terceiro ao SVD. Ao observar os valores absolutos do MACE é possível afirmar que os fatoradores de matriz obtêm os melhores resultados, entretanto ao observar todos os valores é possível afirmar que não há um total melhor desempenho, sendo que todos variam similarmente. Assim, podemos afirmar que os fatoradores de matriz obtêm os melhores desempenhos para o MAP e MACE. Estas afirmações podem ser verificadas com o CCE na tabela.

Todos os outros recomendadores obtêm resultados inferiores aos fatoradores de matriz, quando observados a partir do MAP, entretanto ao verificar o MACE é possível observar que o intervalo de resultados são similares aos fatoradores. A Tabela 5.6 contém os coeficientes dos algoritmos Popularidade e Melhor nota, onde é possível verificar que, para estes dois recomendadores, os coeficientes são menores que o SVD++, demonstrando que ambos causam distorção nos resultados, reforçando a observação de retirá-los da figura.

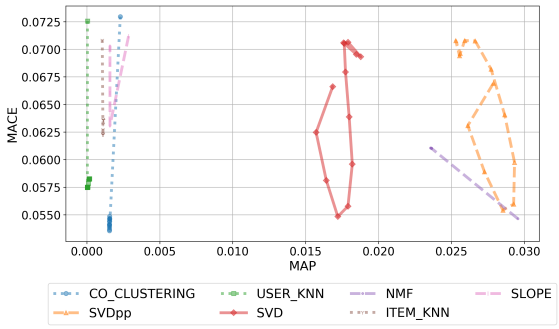
Tabela 5.7 *Movielens* - CCE - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	1.3	34.7	57.4	2.51	0.15	40.28	3.68	**2.38**	711.24
LOG	1.3	57.84	157.32	8.31	0.13	104.41	3.06	**2.08**	60.73

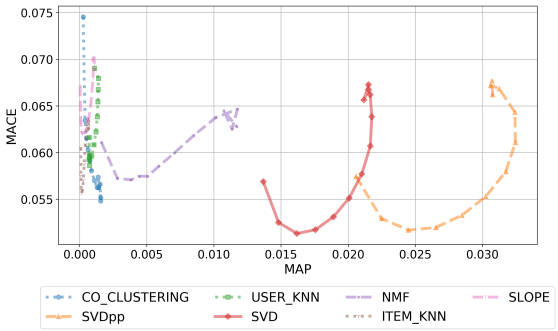
Fonte: Tabela elaborada pelo autor a partir dos resultados.

Figura 5.16 *Movielens* - MACE e MAP - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

(a) *LIN* e Hellinger



(b) *LOG* e Hellinger



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

As Figuras 5.16(a) e (b) em conjunto com a Tabela 5.7 demonstram que o Hellinger segue o comportamento encontrado no KL, em relação ao melhor desempenho ser do balanceamento logarítmico. É possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas por ambas não são correlatadas, assim como o encontrado na combinação das Figuras 5.15(a) e (b). O SVD++ obtém um desempenho superior aos outros algoritmos, assim como debatido previamente, tanto para precisão quanto para o erro na calibragem. A segunda e terceira posição também é disputada entre o SVD e o NMF, que para o *LIN* trocam de posição, assim como encontrado no KL.

Uma das diferenças entre as Figuras 5.15(KL) e 5.16(Hellinger) é que para a formulação linear do balanceamento o Co Clustering obtém um melhor desempenho do MACE quando comparado aos fatoradores de matriz, entretanto o recomendador não apresenta melhoras expressivas no MAP. Ao comparar a combinação do Hellinger com a do KL, é possível observar que os valores do MAP são iguais ou melhores para o Hellinger, este comportamento é verificável com o MACE. Esta afirmação pode ser confirmada pelos valores do coeficientes de cada recomendador. Analogamente ao KL, a Tabela 5.7 contém os coeficientes dos algoritmos Popularidade e Melhor nota, onde é possível verificar que, para estes dois recomendadores, os coeficientes são menores que o SVD++, demonstrando que ambos causam distorção nos resultados apresentados na figura.

Tabela 5.8 *Movielens* - CCE - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

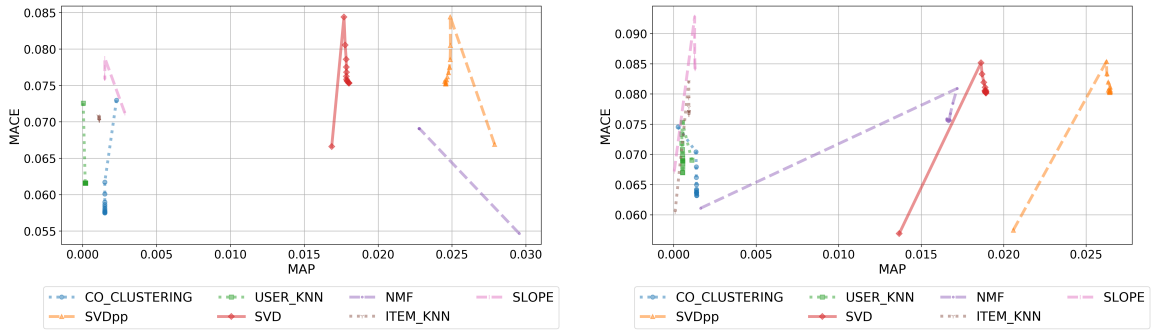
	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	2.57	38.26	63.33	**2.92**	0.15	47.41	4.29	3.06	364.44
LOG	2.59	50.63	88.66	4.83	0.13	71.43	4.3	**3.06**	120.71

Fonte: Tabela elaborada pelo autor a partir dos resultados.

As Figuras 5.17(a) e (b) em conjunto com a Tabela 5.8 demonstram que o Pearson Chi Square segue o mesmo comportamento encontrado no KL e Hellinger, quanto ao desempenho dos fatoradores de matriz, que obtêm resultados superiores aos outros al-

Figura 5.17 *Movielens* - MACE e MAP - Resultados das formulações LIN e LOG com a medida de divergência χ^2.

(a) LIN e Person (b) LOG e Person



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

goritmos, assim como debatido previamente, tanto para precisão quanto para o erro na calibragem. Entretanto, como é possível observar na Tabela 5.8, o recomendador NMF obtém o melhor coeficiente de erro na calibragem. Assim como é possível observar que a formulação LIN obtém os 7 dos 9 melhores desempenhos.

Para a formulação LOG o SVD++ obtém o melhor desempenho, que é o mesmo valor encontrado no LIN. Similarmente ao KL e Hellinger, os algoritmos Popularidade e Melhor nota obtêm os coeficientes menores que o SVD++. As figuras demonstram que há um comportamento visual diferente entre os resultados obtidos a partir da combinação do balanceamento com a medida. É possível verificar que os recomendadores obtêm resultados em intervalos diferentes, dependendo da combinação, tanto para o MAP quanto para o MACE. Como afirmado, é possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre ambas não são correlatadas. Os recomendadores tratados como exceção na análise obtêm os menores coeficientes. Como debatido, estas bases são formadas por itens, onde uma pequena parte possui a maior quantidade das interações com os usuários. Esta condição, em conjunto com o experimento *offline* e o funcionamento do algoritmo, cria uma bolha de recomendação, onde os itens mais populares compõem a lista e agradam a maioria.

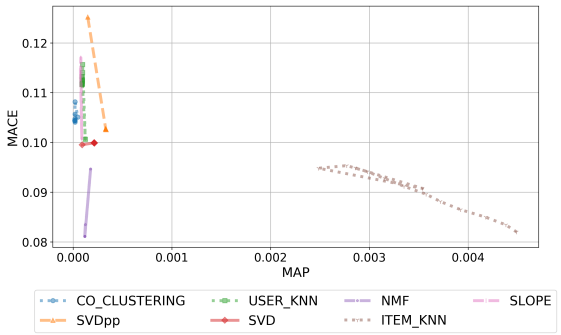
Os resultados apresentados nos permitem indicar o uso da formulação logarítmica em conjunto com o Hellinger e o recomendador SVD++, sendo que esta combinação obtém o melhor valor de coeficiente $CCE = 2.08$ (Tabela 5.7). Esta análise é restrita à base de dados do *Movielens*, considerando o erro na calibragem junto com a precisão como o mais importante na tomada da decisão.

5.5.5.2 OMS Os resultados obtidos com a base de dados do OMS a partir do cruzamento das métricas MACE e MAP (Figuras 5.18, 5.19 e 5.20 e Tabelas 5.9, 5.10 e 5.11) confirmam as observações realizadas tanto na seção do MAP quanto na seção do MACE.

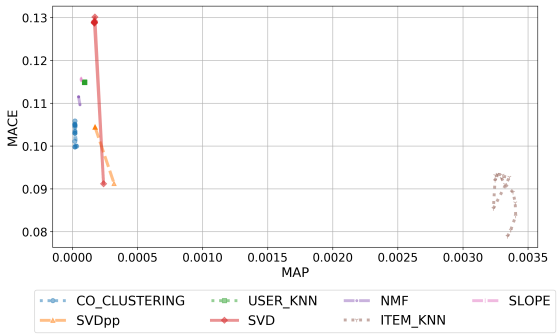
As Figuras 5.18(a) e (b) em conjunto com a Tabela 5.9 demonstram que, para o KL, a maioria dos melhores resultados encontram-se com a formulação do balanceamento linear (LIN), entretanto é possível observar que a variabilidade dos resultados do ba-

Figura 5.18 OMS - MAP e MACE - Resultados das formulações *LIN* e *LOG* com a medida de divergência KL.

(a) *LIN* e Kullback-Leibler



(b) *LOG* e Kullback-Leibler



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Tabela 5.9 OMS - CCE - Resultados das formulações *LIN* e *LOG* com a medida KL.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	1304.07	5093.61	**25.73**	667.67	1.56	1471.03	489.45	329.39	1175.62
LOG	1283.81	5199.86	**26.68**	2307.22	1.46	1714.39	721.64	559.37	1232.02

Fonte: Tabela elaborada pelo autor a partir dos resultados.

lanceamento logarítmico é menor que a do balanceamento linear, indicando uma melhor estabilidade. Apesar do Item-KNN no *LIN* tender a indicar uma correlação positiva decrescente entre o MAP e o MACE, assim como na análise anterior, não é possível verificar uma correlação entre as duas métricas. Diferentemente do *Movielens*, os resultados do OMS mostram que o Item-KNN obtém os melhores resultados. Todos os outros recomendadores obtêm resultados inferiores ao Item-KNN, quando observados a partir do MAP, entretanto ao verificar o MACE é possível observar que o intervalo de resultados é similar ao NMF, para a formulação *LIN*. Todos os outros algoritmos possuem resultados próximos quando observados os valores do MAP, entretanto o SVD++ possui um melhor desempenho que os outros. Estas afirmações podem ser verificadas na Tabela 5.9, onde os valores do coeficiente decisório são apresentados por algoritmo.

Analogamente ao *Movielens*, é possível verificar que o coeficiente do algoritmo Popularidade é o menor para ambas as formulações, demonstrando que o recomendador é uma exceção na análise, como já debatido. Entretanto o algoritmo Melhor nota possui um comportamento diferente e, neste caso, não obtém a segunda posição em relação ao desempenho. Este comportamento é influência do funcionamento do algoritmo, tendo em vista que o recomendador baseia-se em feedback explícito, sendo a base de dados do OMS composta por feedback implícito.

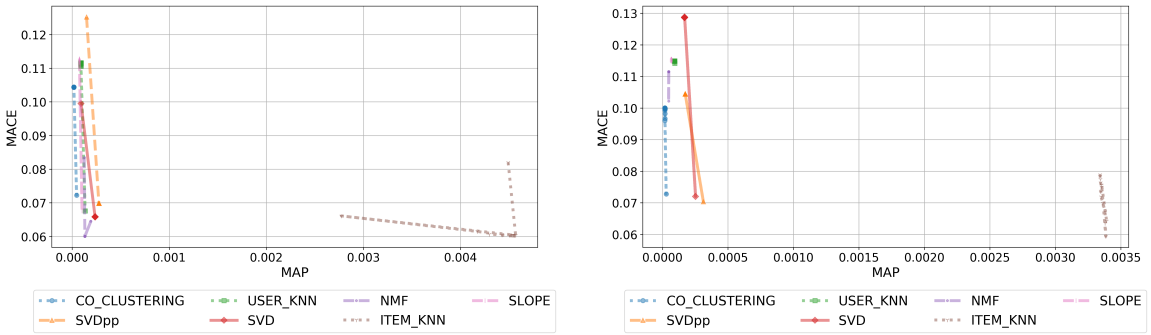
Tabela 5.10 OMS - CCE - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	1596.75	5072.75	**14.39**	459.35	1.51	1406.45	303.95	278.98	1148.17
LOG	1565.07	5057.98	**21.32**	2336.09	1.46	1709.8	709.28	552.41	1234.66

Fonte: Tabela elaborada pelo autor a partir dos resultados.

Figura 5.19 OMS - MAP e MACE - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

(a) *LIN* e Hellinger (b) *LOG* e Hellinger

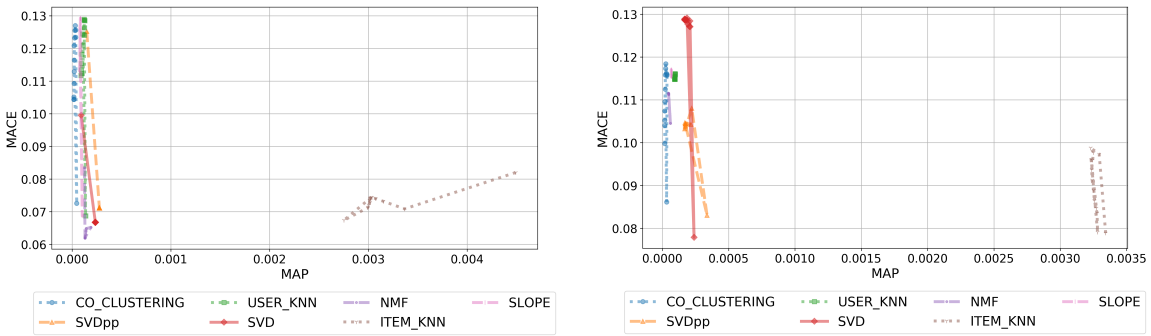


Fonte: Figuras elaboradas pelo autor a partir dos resultados.

As Figuras 5.19(a) e (b) em conjunto com a Tabela 5.10 demonstram que o Hellinger segue o mesmo comportamento encontrado no KL e que a melhor formulação do balanceamento é a linear, obtendo menores valores no MACE e maiores valores no MAP. O Item-KNN obteve um desempenho superior aos outros algoritmos, assim como debatido previamente. A segunda e terceira posição também são disputadas entre o SVD++ e o SVD. É possível verificar que, assim como no KL, o *LOG* causa um efeito de reduzir a variabilidade do Item-KNN quanto ao valor do MAP. Os recomendadores Popularidade e Melhor nota obtêm comportamentos similares ao encontrado no KL.

Figura 5.20 OMS - MAP e MACE - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

(a) *LIN* e Person (b) *LOG* e Person



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

As Figuras 5.20(a) e (b) em conjunto com a Tabela 5.11 demonstram que o Person Chi Square segue o mesmo comportamento encontrado no KL e Hellinger, quanto ao desempenho do Item-KNN, que obteve resultados superiores aos outros algoritmos, assim como debatido previamente. É observável também que há uma redução da variabilidade quando o *LOG* é usado no Item-KNN. Os fatoradores de matriz possuem um comportamento de obter melhores resultados que os outros recomendadores, assim como observado

Tabela 5.11 OMS - CCE - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	1047.6	4308.4	**23.29**	469.9	1.53	1438.43	308.4	283.38	1047.84
LOG	1048.7	4442.51	**27.64**	2287.83	1.46	1712.22	659.49	531.97	1223.63

Fonte: Tabela elaborada pelo autor a partir dos resultados.

anteriormente. Para o χ^2, o *LIN* obtém o melhor desempenho em ambas as métricas.

Todos os gráficos pertencentes às figuras desta seção demonstram que a maioria dos recomendadores obtêm um intervalo em comum de MAP e MACE, esses intervalos também são compartilhados entre as medidas de divergência e as formulações do balanceamento. É possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre as duas métricas são diferentes. Como debatido, estas bases são formadas por itens, onde uma pequena parte possui a maior quantidade das interações com os usuários. Esta condição, em conjunto com o experimento *offline* e o funcionamento do algoritmo Popularidade, cria uma bolha de recomendação, onde os itens mais populares sempre irão agradar. Este comportamento dos algoritmos é perceptível por diversas métricas.

Os resultados apresentados nos permitem indicar o uso da formulação linear em conjunto com o Hellinger e o recomendador Item-KNN, sendo que esta combinação obtém o melhor valor de coeficiente $CCE = 14.39$ (Tabela 5.10). Esta análise é restrita à base de dados do OMS, considerando o erro na calibragem junto com a precisão como o mais importante na tomada da decisão.

5.5.6 MAPXMRMC

Nesta seção é debatido o cruzamento dos resultados, buscando entender em cada base de dados como as métricas MAP e MRMC se comportam. As Figuras 5.21, 5.22 e 5.23 representam os cruzamentos dos resultados obtidos na base de dados do *Movielens*. As Tabelas 5.12, 5.13 e 5.14 apresentam o coeficiente de descalibragem do *Movielens*.

As Figuras 5.24, 5.25 e 5.26 representam os cruzamentos dos resultados obtidos na base de dados do OMS. As Tabelas 5.15, 5.16 e 5.17 apresentam o coeficiente de descalibragem do OMS. Para realizar a análise torna-se necessário retirar os algoritmos Popularidade e Melhor nota do cruzamento, assim como na seção anterior.

5.5.6.1 Movielens Os resultados da base de dados do *Movielens* a partir do cruzamento das métricas MAP e MRMC (Figuras 5.21, 5.22 e 5.23 e Tabelas 5.12, 5.13 e 5.14) confirmam as observações realizadas tanto na seção do MAP quanto na seção do MRMC.

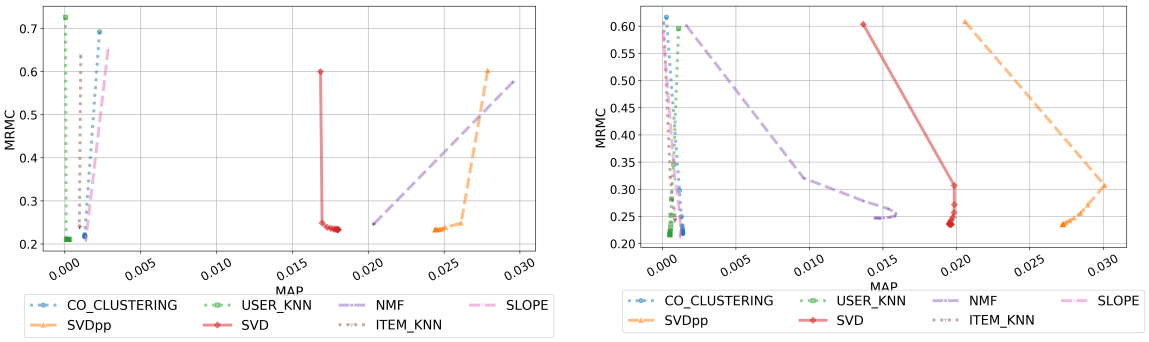
Tabela 5.12 *Movielens* - CMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência KL.

	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	2.73	60.44	90.73	3.8	0.15	63.52	4.95	**3.52**	430.91
LOG	2.75	62.94	110.46	5.76	0.13	83.46	4.31	**3.05**	130.58

Fonte: Tabela elaborada pelo autor a partir dos resultados.

Figura 5.21 *Movielens* - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência KL.

(a) *LIN* e Kullback-Leibler (b) *LOG* e Kullback-Leibler



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

As Figuras 5.21(a) e (b) em conjunto com a Tabela 5.12 demonstram que, para o KL, a melhor formulação do balanceamento é a logarítmica, proposta desta dissertação, seguindo o encontrado no MAPXMACE. Esta combinação obteve os maiores valores do MAP e disputou os melhores valores do MRMC, em conjunto com o recomendador SVD++. O algoritmo SVD, para formulação *LOG*, ocupa a segunda colocação entre os melhores desempenhos. O terceiro lugar é ocupado pelo NMF. Na formulação linear o segundo lugar pertence ao NMF e o terceiro ao SVD. Assim, podemos afirmar que os fatoradores de matriz obtêm os melhores desempenhos para o MAP e MRMC. Todos os outros recomendadores obtêm resultados inferiores aos fatoradores de matriz, quando observados a partir do MAP (eixo x), entretanto ao verificar o MRMC (eixo y) é possível observar que o intervalo de resultados são similares aos fatoradores. Estas afirmações podem ser verificadas com o CMC na tabela. A ordem dos resultados apresentados para ambas as formulações é igual à ordem dos resultados encontrados no MAPXMACE. Assim como é possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre as duas métricas são diferentes.

Resultados anteriores indicaram que alguns comportamentos podem ser observados por ambas as métricas, entretanto é possível afirmar que suas análises são diferentes. A Tabela 5.6 contém os coeficientes de descalibragem dos algoritmos Popularidade e Melhor nota, onde é possível verificar que para estes dois recomendadores os coeficientes são menores que o SVD++, demonstrando que ambos causam distorção nos resultados apresentados na figura, seguindo o encontrado no MAPXMACE.

Tabela 5.13 *Movielens* - CMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

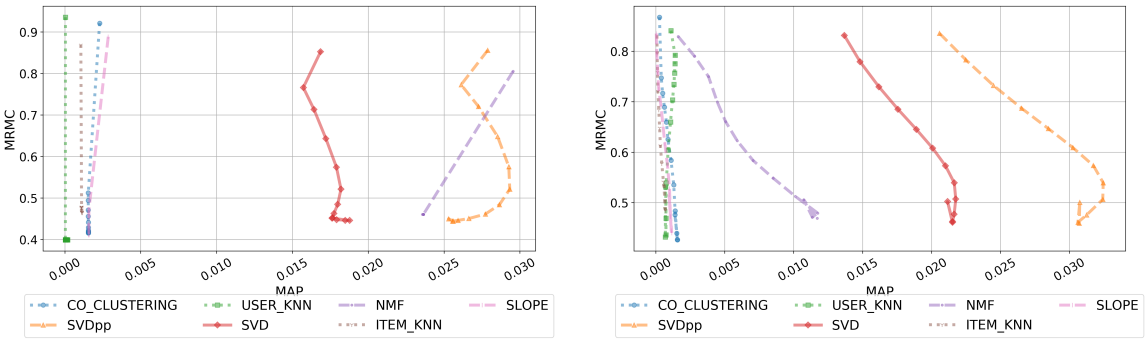
	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN	13.81	298.94	454.46	**20.22**	2.35	278.07	31.78	20.56	5319.06
LOG	13.9	576.33	1654.34	83.42	2.31	982.12	31.01	**20.95**	616.98

Fonte: Tabela elaborada pelo autor a partir dos resultados.

As Figuras 5.22(a) e (b) em conjunto com a Tabela 5.13 demonstram que o Hellinger

Figura 5.22 *Movielens* - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

(a) *LIN* e Hellinger (b) *LOG* e Hellinger

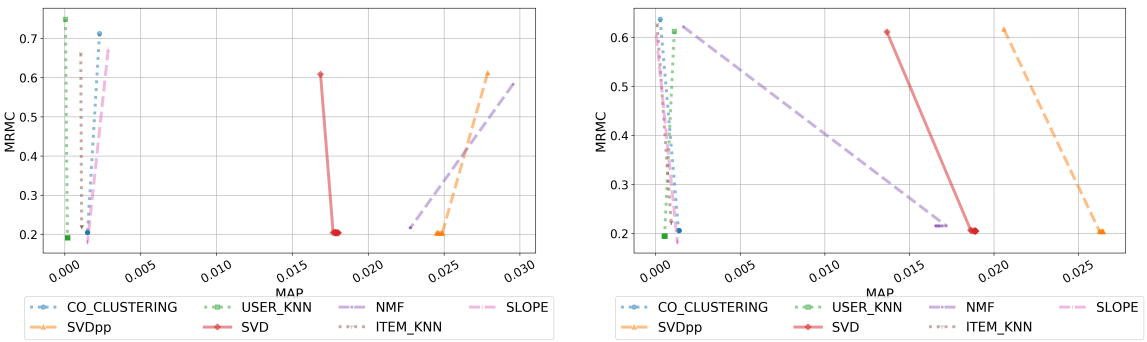


Fonte: Figuras elaboradas pelo autor a partir dos resultados.

obtém resultados diferentes do KL. O melhor desempenho é obtido através do uso do *LIN*. O NMF obteve um desempenho superior ao SVD++, entretanto a diferença entre os algoritmos é pequena. O terceiro lugar pertence ao SVD. Para o *LOG*, o melhor algoritmo é o SVD++, assim como encontrado no KL. A segunda e terceira posição também são disputadas entre o SVD e o NMF. É possível observar que o Hellinger, em valores absolutos, obtém melhores resultados que o KL, quando observado a partir do MAP. Entretanto o KL obtém melhores desempenhos quando os coeficientes são observados. Os resultados da formulação do *LOG* (Figura 5.22(b)) dão um indicativo de que existe correlação entre o crescimento positivo do MAP e o decaimento do valor da descalibragem (MRMC), entretanto esta correlação é fraca para ser afirmada. Analogamente ao KL, a Tabela 5.13 contém os coeficientes dos algoritmos Popularidade e Melhor nota, onde é possível verificar que, para estes dois recomendadores, os coeficientes são menores que os fatoradores de matriz, demonstrando que ambos causam distorção nos resultados apresentados na figura.

Figura 5.23 *Movielens* - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

(a) *LIN* e Person (b) *LOG* e Person



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Tabela 5.14 *Movielens* - CMC - Resultados das formulações LIN e LOG com a medida de divergência χ^2.

	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	7.6	156.53	228.18	10.52	1.55	136.21	13.21	**9.39**	1367.03
LOG	7.66	184.39	294.33	15.84	1.58	181.31	12.8	**9.08**	391.45

Fonte: Tabela elaborada pelo autor a partir dos resultados.

As Figuras 5.23(a) e (b) em conjunto com a Tabela 5.14 demonstram que o Person Chi Square também possui um comportamento diferente do encontrado no KL e Hellinger, quanto ao desempenho dos algoritmos. Como é possível verificar o LIN obtém a maioria dos melhores resultados, sendo o SVD++ o recomendador com o melhor desempenho. Os outros fatoradores de matriz ocupam o segundo e terceiro lugar, sendo respectivamente o NMF e o SVD. Para o LOG, o melhor desempenho ainda encontra-se com o SVD++, entretanto o SVD e o NMF trocam as posições. Assim como o encontrado previamente, os algoritmos tratados como exceção obtêm desempenhos melhores que os fatoradores de matriz, como demonstrado na Tabela 5.14.

As figuras demonstram que há um comportamento diferente entre os resultados obtidos a partir da combinação do balanceamento com a medida. É possível verificar que os recomendadores obtêm resultados em intervalos diferentes, dependendo da combinação. É possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre as duas métricas são diferentes. Os recomendadores tratados como exceção na análise obtêm os menores coeficientes. Como debatido, estas bases são formadas por itens, onde uma pequena parte possui a maior quantidade das interações com os usuários. Esta condição, em conjunto com o experimento *offline* e o funcionamento do algoritmo, cria uma bolha de recomendação, onde os itens mais populares compõem a lista e agradam a maioria.

Os resultados encontrados no cruzamento das métricas MAP e MRMC nos permitem indicar o uso da formulação logarítmica em conjunto com o KL, sendo que esta combinação obtém o melhor valor de coeficiente $CMC = 3.05$ (Tabela 5.12). Esta análise é restrita à base de dados do *Movielens*.

5.5.6.2 OMS Os resultados do OMS a partir do cruzamento das métricas MAP e MRMC (Figuras 5.24, 5.25 e 5.26 e as Tabelas 5.15, 5.16 e 5.17) confirmam as observações realizadas tanto na seção do MAP quanto na seção do MRMC.

Tabela 5.15 OMS - CMC - Resultados das formulações LIN e LOG com a medida de divergência KL.

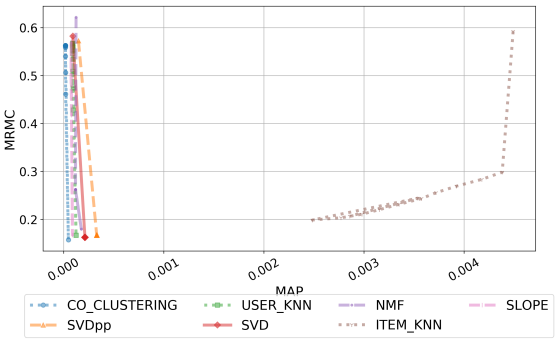
	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	2623.32	25124.25	**76.12**	2299.61	13.04	6598.41	953.96	624.99	5331.51
LOG	2588.21	22937.04	**115.76**	11416.07	13.3	8539.23	3251.54	2971.04	6144.5

Fonte: Tabela elaborada pelo autor a partir dos resultados.

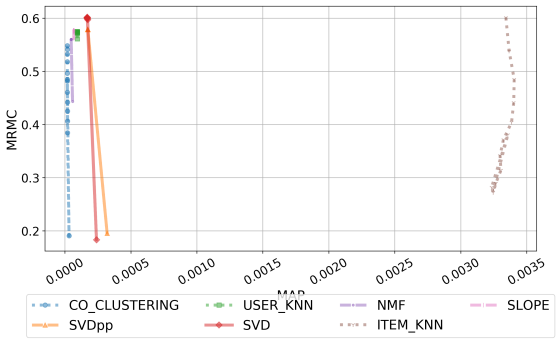
As Figuras 5.24(a) e (b) em conjunto com a Tabela 5.15 demonstram que, para o KL, a maioria dos melhores resultados encontram-se com a formulação do balanceamento linear, entretanto é possível observar que a variabilidade dos resultados do balanceamento logarítmico é menor que a do balanceamento linear, indicando uma melhor estabilidade.

Figura 5.24 OMS - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência KL.

(a) *LIN* e Kullback-Leibler



(b) *LOG* e Kullback-Leibler



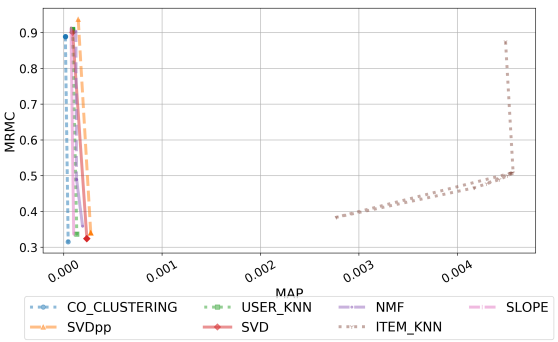
Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Diferentemente do *Movielens*, os resultados do OMS mostram que o Item-KNN obtém os melhores valores do MAP. Todos os outros recomendadores obtêm desempenhos inferiores ao Item-KNN, quando observados a partir do MAP, entretanto ao verificar o MRMC é possível observar que os intervalos de resultados são similares. Estes comportamentos também são encontradas no cruzamento MAPXMACE na base do OMS. Para o *LIN* e o *LOG*, o segundo e terceiro melhor desempenho pertencem ao SVD++ e o SVD.

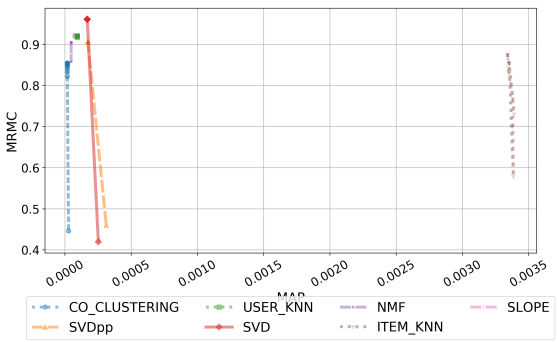
Analogamente ao *Movielens* é possível verificar que o coeficiente do algoritmo Popularidade é o menor para ambas as formulações, demonstrando que o recomendador é uma exceção na análise, como já debatido. Entretanto o algoritmo Melhor nota possui um comportamento diferente e, neste caso, não obtém a segunda posição em relação ao desempenho. Este comportamento é influência do funcionamento do algoritmo, tendo em vista que o recomendador baseia-se em feedback explícito, sendo a base de dados do OMS composta por feedback implícito. Este comportamento dos algoritmos, que fazem parte da exceção, é perceptível também nas outras medidas de divergência.

Figura 5.25 OMS - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

(a) *LIN* e Hellinger



(b) *LOG* e Hellinger



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Tabela 5.16 OMS - CMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência Hellinger.

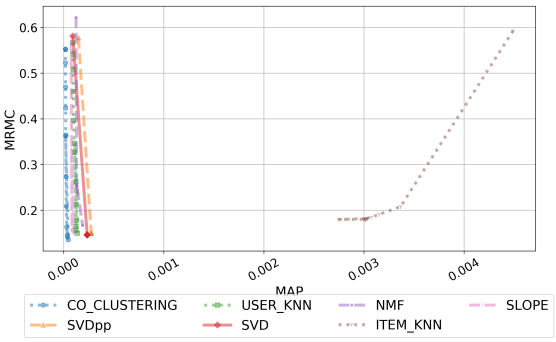
	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	9098.46	42028.72	**119.23**	3775.94	18.04	11175.3	1635.21	1452.92	9181.55
LOG	8932.42	42531.68	**239.19**	19002.44	18.23	13655.52	5243.78	4729.24	9891.66

Fonte: Tabela elaborada pelo autor a partir dos resultados.

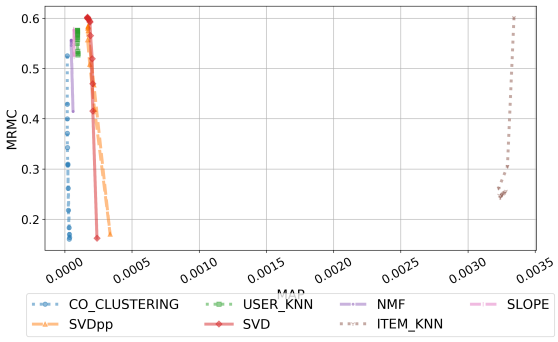
As Figuras 5.25(a) e (b) em conjunto com a Tabela 5.16 demonstram que o Hellinger segue o comportamento encontrado no KL e que a melhor formulação do balanceamento é a linear, tanto para o MAP quanto para o MRMC. Esta afirmação é confirmada por ambas as tabelas, onde o coeficiente decisório da calibragem é apresentado. O Item-KNN obteve desempenho superior aos outros algoritmos, assim como debatido previamente. A segunda e terceira posição também são disputadas entre o SVD++ e o SVD. Todos os outros algoritmos obtêm resultados similares. As observações sobre os recomendadores analisados como exceção são similares às apresentadas previamente.

Figura 5.26 OMS - MAP e MRMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

(a) *LIN* e Person



(b) *LOG* e Person



Fonte: Figuras elaboradas pelo autor a partir dos resultados.

Tabela 5.17 OMS - CMC - Resultados das formulações *LIN* e *LOG* com a medida de divergência χ^2.

	Melhor nota	Co Clustering	Item KNN	NMF	Popularidade	Slope One	SVD	SVD++	User KNN
LIN	1381.17	12640.08	**68.46**	1987.01	12.39	3937.01	798.85	687.41	2939.81
LOG	1370.89	11985.59	**86.13**	11219.71	13.11	8244.53	2818.63	2696.87	5937.93

Fonte: Tabela elaborada pelo autor a partir dos resultados.

As Figuras 5.26(a) e (b) em conjunto com a Tabela 5.17 demonstram que o Person Chi Square segue o mesmo comportamento encontrado no KL e Hellinger, quanto ao desempenho do Item-KNN, que obteve resultados superiores aos outros algoritmos, assim como debatido previamente. É observável também que há uma redução da variabilidade quando o *LOG* é usado no Item-KNN, assim como encontrado no cruzamento anterior. Os fatoradores de matriz possuem um comportamento que obtém melhores resultados que os outros recomendadores, entretanto essa diferença é pequena, assim como observado anteriormente. Para o χ^2, o balanceamento linear obtém o melhor desempenho.

Todos os gráficos das figuras acima demonstram que a maioria dos recomendadores obtêm um intervalo em comum de MAP e MRMC, esses intervalos também são compartilhados entre as medidas de divergência com as formulações do balanceamento, contudo, diferenças no maior e menor valores do intervalo de cada combinação acontecem. É possível observar que não há uma correlação direta e visual entre as duas métricas, indicando que as observações retratadas entre as duas métricas são diferentes.

Os resultados apresentados nos permitem indicar o uso da formulação linear em conjunto com o χ^2 e o recomendador Item-KNN, sendo que esta combinação obtém o melhor valor de coeficiente $CMC = 68.46$ (Tabela 5.11). Esta análise é restrita à base de dados do OMS, considerando as observações feitas sobre os cruzamentos dos resultados MAPXMRMC.

5.6 DECISÃO DO PROTOCOLO

Os resultados previamente apresentados permitem elucidar o desempenho de cada combinação de sistema calibrado individualmente. Entretanto não esclarece qual a combinação deve ser implementada para cada base de dados a partir do conjunto de sistemas gerados. Assim, como descrito no protocolo na Seção 4.8, é necessário realizar a verificação de qual é a melhor combinação para cada domínio.

O protocolo considera a soma do CCE com o CMC para saber o desempenho da combinação de sistema, levando em conta a precisão e a calibragem. Ao considerar duas métricas na decisão, a melhor combinação de sistema pode vir a ser diferente do avaliado individualmente. Vale ressaltar que esta pesquisa é focada em considerar formas de recomendar que vão além da precisão.

5.6.1 Movielens

A Tabela 5.18 apresenta os valores do desempenho de cada combinação de sistema calibrado implementado e testado no experimento para a base de dados do Movielens. Como é possível verificar, ao considerar o desempenho do sistema como $s = CCE + CMC$ tem-se que todos os melhores resultados pertencem à medida de divergência Pearson Chi Square, que é uma das propostas desta dissertação.

Tabela 5.18 Desempenho de cada combinação de sistema calibrado no Movielens.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN_KL	12,0	244,1	365,4	16,7	1,78	224,86	19,83	14,06	1682,22
LOG_KL	12,08	268,94	484,4	26,8	1,75	321,3	18,65	13,14	575,0
LIN_HE	15,11	333,64	511,86	22,73	2,5	318,35	35,46	22,94	6030,3
LOG_HE	15,2	634,17	1811,66	91,73	2,44	1086,53	34,07	23,03	677,71
LIN_CHI	**10,17**	**194,79**	**291,51**	**13,44**	**1,7**	**183,62**	17,5	12,45	1731,47
LOG_CHI	10,25	235,02	382,99	20,67	1,71	252,74	**17,1**	**12,14**	**512,16**

Fonte: Tabela elaborada pelo autor a partir dos resultados.

A observação dos algoritmos considerados como exceção demonstra que o balanceamento LIN com o χ^2 obtém os melhores resultados para ambos os recomendadores. Ao observar os outros algoritmos percebe-se que o SVD++ juntamente com o balanceamento LOG e o χ^2 obtém o melhor desempenho entre os recomendadores. Assim, a

implementação que o protocolo recomenda para a base de dados do Movielens, seguindo os passos aplicados nesta dissertação, é o SVD++ em conjunto com o LOG e o χ^2. Esta combinação de sistema é uma proposta desta dissertação, demonstrando que o protocolo proposto consegue encontrar o melhor sistema calibrado para o domínio.

5.6.2 OMS

A Tabela 5.19 apresenta os valores do desempenho de cada combinação de sistema calibrado implementado e testado no experimento para a base de dados do OMS. Como é possível verificar, ao considerar o desempenho do sistema como a soma do CCE com o CMC tem-se que os melhores resultados pertencem à medida de divergência Pearson Chi Square, que é uma das propostas desta dissertação. Entretanto, os fatoradores de matriz SVD++ e NMF obtêm os melhores resultados utilizando o KL.

Tabela 5.19 Desempenho de cada combinação de sistema calibrado no OMS.

	Melhor nota	**Co Clustering**	**Item KNN**	**NMF**	**Popularidade**	**Slope One**	**SVD**	**SVD++**	**User KNN**
LIN_KL	3927,39	30217,86	101,85	**2967,28**	14,6	8069,44	1443,41	**954,38**	6507,13
LOG_KL	3872,02	28136,9	142,44	13723,29	14,76	10253,62	3973,18	3530,41	7376,52
LIN_HE	10695,21	47101,47	133,62	4235,29	19,55	12581,75	1939,16	1731,9	10329,72
LOG_HE	10497,49	47589,66	260,51	21338,53	19,69	15365,32	5953,06	5281,65	11126,32
LIN_CHI	2428,77	16948,48	**91,75**	2456,91	**13,92**	**5375,44**	**1107,25**	970,79	**3987,65**
LOG_CHI	**2419,59**	**16428,1**	113,77	13507,54	14,57	9956,75	3478,12	3228,84	7161,56

Fonte: Tabela elaborada pelo autor a partir dos resultados.

A observação do algoritmo Popularidade considerado como exceção demonstra que o balanceamento LIN com o χ^2 obtém os melhores resultados para o recomendador, assim como o encontrado para a base de dados do Movielens. Ao observar o Item-KNN é possível verificar que o balanceamento LIN em conjunto com o χ^2 obtém o melhor desempenho. Assim, a implementação recomendada pelo protocolo para a base de dados do OMS, seguindo os passos aplicados nesta dissertação, é o Item-KNN em conjunto com o LIN e o χ^2. Esta combinação de sistema é uma mistura do proposto pelo estado-da-arte e a proposta desta dissertação, demonstrando que o protocolo proposto consegue encontrar o melhor sistema calibrado para o domínio independente de implementação.

5.7 DISCUSSÃO

A partir dos resultados obtidos algumas informações podem ser afirmadas. Para o *Movielens* a métrica MAP indica que a formulação do balanceamento logarítmico, proposto nesta dissertação, propicia um incremento nos desempenhos dos recomendadores SVD++, SVD e Popularidade. É possível afirmar também que a proposta de utilizar a medida de divergência χ^2 produz efeitos positivos nos desempenhos dos recomendadores. As propostas do peso do balanceamento produzem melhorias no desempenho ou, quando não produzem, reproduzem o desempenho e comportamento dos pesos atribuídos pelo especialista do sistema. Para o OMS a métrica MAP indica que ser popular é mais influente do que ser ouvido muitas vezes individualmente pelos usuários. Similar ao *Movielens*, os pesos personalizados do balanceamento, proposta desta pesquisa, produzem melhorias no recomendador ou reproduzem o comportamento e desempenho dos valores constantes,

assim como o uso da medida χ^2 também produz ou reproduz efeitos positivos similares às outras medidas usadas pelo estado-da-arte. Para ambas as bases, o pós-processamento produziu efeitos positivos no desempenho, quando avaliado pela métrica MAP.

Os resultados obtidos a partir da avaliação com o MRR seguem o mesmo comportamento do MAP para ambas as bases de dados e para todos os algoritmos. A partir das observações sobre as duas métricas de ranqueamento pode-se concluir que, ao criar a lista de recomendação, o pós-processamento traz o primeiro item mais relevante para mais perto do topo da lista, influenciando positivamente, assim, todo o processo de preenchimento das próximas posições.

Os resultados obtidos analisados a partir da métrica de avaliação proposta nesta dissertação, chamada de MACE, indica que os resultados de menor erro absoluto na calibragem dependem da base de dados e sua composição. Para o *Movielens* os melhores resultados são obtidos pela formulação do balanceamento logarítmico. Para o OMS o balanceamento linear obtém os melhores desempenhos. Os comportamentos observados em outras métricas de avaliação são capturados pela MACE também, como é possível verificar que o peso do balanceamento totalmente focado em justiça causa um efeito imediato na base do OMS. Esse comportamento pode ser atribuído à constituição das músicas, que possuem no mínimo um e no máximo dois gêneros, sendo que os filmes do *Movielens* possuem de um até cinco gêneros.

A métrica MRMC indica a possibilidade de reduzir a descalibragem em qualquer algoritmo recomendador, demonstrando que a aplicação do pós-processamento produz um efeito positivo na lista. É possível observar que a medida de divergência usada é diretamente influente no resultado obtido. A medida χ^2, proposta desta dissertação, obtém os melhores resultados.

A partir das análises obtidas com os cruzamentos das métricas e a decisão do protocolo é possível afirmar que, dependendo da base de dados utilizada, a melhor combinação de sistema muda. Para a base de dados do *Movielens* é recomendável utilizar a combinação de sistema: SVD++, a formulação logarítmica do balanceamento, a medida de divergência Pearson Chi Square e o peso personalizado VAR. Para a base de dados do OMS é recomendável utilizar a combinação de sistema com: Item-KNN, a formulação linear do balanceamento, a medida de divergência χ^2 e o peso personalizado CGR ou VAR.

5.8 COMPARAÇÃO COM OUTROS TRABALHOS DA LITERATURA

Steck (2018) e Kaya e Bridge (2019) afirmam que, para o *Movielens*, quanto maior o valor do peso no balanceamento relevância-divergência é esperado que a precisão caia, o que em parte é confirmado por este estudo, entretanto os algoritmos recomendadores baseados em fatoração de matrizes aumentam a precisão conforme o peso aumenta em direção à calibragem. Os resultados positivos também são sentidos pelos pesos personalizados, reforçando a melhora promovida pela etapa de pós-processamento. Os resultados de Kaya e Bridge (2019) sobre a base de dados do OMS indica um crescimento na precisão, o que é encontrado por esta dissertação, reforçando a afirmação positiva pelo uso do pós-processamento.

Lin et al. (2019) e Abdollahpouri et al. (2020) encontram que os fatoradores de matrizes possuem os menores níveis de enviesamento. Koren, Bell e Volinsky (2009) afirmam que o SVD++ produz uma melhora em comparação ao SVD, sendo o primeiro uma melhoria do segundo. Tal afirmação pode ser confirmada por esta dissertação. Os resultados deste estudo também permite afirmar que o uso do pós-processamento produz uma melhora na precisão do SVD++, assim como o auxilia na redução da descalibragem, criando uma lista de recomendação com precisão e calibragem.

5.9 RESPOSTAS PARA AS QUESTÕES DE PESQUISA

Diante dos problemas de pesquisa apontados na Seção 1.4, neste ponto da dissertação respostas já podem ser afirmadas. Assim, nesta seção cada uma das questões de pesquisa é respondida.

QP1: *Como encontrar o melhor sistema de recomendação calibrado?* Na Seção 4.8 um protocolo de decisão foi proposto, baseando-se no modelo de sistema calibrado também proposto nesta dissertação. Na Seção 5.5 os resultados foram apresentados, debatendo métrica por métrica. A Seção 5.6 foi apresentada a melhor combinação de sistema calibrado que o protocolo de decisão recomenda. Assim, esta dissertação demonstrou que é possível encontrar o melhor sistema calibrado dentre um conjunto de possibilidades.

QP2: *A base de dados utilizadas influencia no comportamento ou desempenho do sistema calibrado?* Neste capítulo foi apresentado o experimento para validar as propostas desta dissertação. No início duas bases de dados pertencentes a diferentes domínios foram apresentadas. Na Seção 5.5 foram apresentados os resultados do experimento, sendo possível verificar, ao decorrer do debate dos resultados, que a base de dados utilizada influencia no comportamento do sistema. Assim, para cada base de dados uma combinação de sistema calibrado diferente obteve o melhor desempenho. Para o Movielens foi o SVD++, LOG e χ^2. Para o OMS foi o Item-KNN, LIN e χ^2.

QP3: *É possível padronizar os sistemas calibrados de modo que auxilie o desenvolvimento?* Na Seção 4.1 foi apresentado um modelo de sistema calibrado, o qual pode ser modificado e adaptado a depender da necessidade da aplicação. Este modelo incorpora o protocolo de decisão proposto na Seção 4.8. Ambos foram testados neste capítulo através do experimento descrito previamente. Assim, pode-se afirmar que o modelo de sistema calibrado é uma padronização, i. e., um quadro pré-moldado que pode ser alterado gerando diferentes combinações de sistema.

QP4: *Quais são os efeitos que a calibragem produz nas listas de recomendação de cada algoritmo recomendador?* Na Seção 5.3 foram apresentados todos os algoritmos recomendadores utilizados no experimento desta dissertação, sendo ao todo nove. Na seção de resultados presente neste capítulo, viu-se que a depender do recomendador utilizado o efeito produzido pela calibragem na lista de recomendação muda. Alguns algoritmos obtiveram um acréscimo no MAP e outros, decréscimo, assim como para as outras métricas. Com isso, é possível afirmar que cada recomendador produzirá diferentes resultados e efeitos em conjunto com a calibragem.

QP5: *Utilizar pesos personalizados do balanceamento obtém melhorias ou mantém desempenho quando comparados com pesos constantes?* Na Seção 4.5.4 dois pesos perso-

nalizados do balanceamento foram propostos como parte desta dissertação. Na Seção 5.3 onze pesos constantes foram apresentados, estes que haviam sido utilizados pelo estado-da-arte. Na seção de resultados todos os treze pesos utilizados foram apresentados e debatidos a partir de cada métrica de avaliação. Foi possível observar que o desempenho da combinação de sistema é influenciado pelo peso utilizado. Entretanto, se a formulação da combinação de sistema tende a reduzir a precisão, os pesos utilizados apenas afetarão o resultado em maior ou menor escala, mas manterão o comportamento da combinação. Assim, a partir dos resultados é possível afirmar que o peso do balanceamento é um parâmetro otimizador, que por sua vez pode ser atribuído pelo especialista do sistema ou encontrado de forma personalizada. Ambas as formas do peso obtêm resultados similares, ou seja, o peso personalizado do balanceamento produz o comportamento da combinação, em alguns momentos demonstrando melhora no desempenho e em outros mantendo desempenhos previamente encontrados pelos pesos constantes.

QP6: *A medida de divergência utilizada influência nas listas de recomendação?* Na Seção 4.5.2 três medidas de divergência foram apresentadas, uma utilizada pelo estado-da-arte e duas propostas desta dissertação. Os resultados indicam que a medida influencia na lista de recomendação, gerando diferentes tipos de lista. Estas que por sua vez, trazem maior ou menor precisão, assim como produzem acréscimo ou decréscimo no desempenho da calibragem. Entretanto, para ambas as bases de dados a medida de divergência com o melhor desempenho foi o Pearson Chi Square, demonstrando que a medida pode trabalhar bem com sistemas calibrados.

QP7: *Ao considerar o viés do usuário na formulação do balanceamento é possível obter melhoria no desempenho?* Na Seção 4.5.5 foi apresentada uma nova proposta de formulação para o balanceamento de sistemas calibrados. Esta formulação considera o viés do usuário durante a criação da lista de recomendação. Na Seção 5.5 foram debatidos os resultados, comparando balanceamento linear, proposta do estado-da-arte, com o balanceamento logarítmico, proposta desta dissertação. Os resultados demonstraram que ao considerar o viés do usuário é possível obter melhora no desempenho, tanto da precisão como da calibragem. A depender da base de dados utilizada o viés do usuário pode ser positivo.

5.10 SUMÁRIO

Neste capítulo foram apresentadas as base de dados usadas, assim como a filtragem e seleção dos dados aplicados. Após a apresentação das bases foi realizada uma análise da composição dos dados e seus comportamentos, analisando os itens, os usuários, a popularidade e os gêneros. A metodologia de execução do sistema proposto também foi apresentada neste capítulo, descrevendo os algoritmos recomendadores e o processo de encontrar seus hiper parâmetros, assim como todos os valores usados. Os dados dos experimentos e o processamento de etapa a etapa foram demonstrados, assim como as máquinas usadas. Em seguida foram debatidas as métricas utilizadas para avaliar o sistema proposto. Os resultados obtidos foram apresentados e debatidos base-a-base e métrica-a-métrica. Ao fim, foi realizada uma discussão sobre os resultados, uma comparação com o estado-da-arte e um debate sobre as questões de pesquisa.

Capítulo 6

CONCLUSÃO

Este capítulo conclui o trabalho, apresentando uma visão geral dos resultados alcançados, as contribuições do estudo à comunidade científica, publicações de artigos, as limitações e orientações para trabalhos futuros. Para finalizar este trabalho considerações finais são realizadas.

O Capítulo 2 introduziu o referencial teórico da área de Sistemas de Recomendação. Trabalhos fundamentais foram apresentados, abordando diversos conteúdos que definem a área. Os conceitos básicos e tarefas para Sistemas de Recomendação foram descritos e debatidos. As formas de obter o feedback do usuário e como modelar esses dados obtidos foram descritas. Três técnicas de recomendação foram apresentadas, sendo a técnica de filtragem colaborativa descrita mais a fundo, assim como alguns dos algoritmos que aplicam a técnica. As métricas de avaliação comumente utilizadas em Sistemas de Recomendação foram apresentadas.

No Capítulo 3 abordou-se os conceitos de justiça e calibragem em Sistemas de Recomendação. Ao longo do capítulo foi definido o cenário de justiça em Sistemas de Recomendação, em especial o contexto de calibragem. Diversos trabalhos que servem como base de estudo foram apresentados, juntamente com um resumo dos conceitos e objetivos abordados por eles. Um *framework* conceitual foi apresentado em conjunto com os estudos envolvidos em Sistemas de Recomendação que prezam a justiça. O contexto da calibragem foi abordado em conjunto com estudos relacionados. Ao final, os trabalhos do estado-da-arte foram definidos para serem comparados com esta dissertação.

O Capítulo 4 apresentou a proposta desta dissertação, que é um modelo de sistema de recomendação baseado em filtragem colaborativa *C-Fairness* que em uma etapa de pós-processamento calibra a lista de recomendação baseando-se nos gêneros dos itens presentes nas preferências dos usuários. Assim, na seção foram abordados: o sistema e suas etapas, a notação formal, as modelagens dos dados, um pseudo algoritmo que exemplifica o funcionamento do sistema, as distribuições de gêneros, as medidas de divergência, a medida de ranque, os pesos do balanceamento relevância-divergência, as equações de balanceamento relevância-divergência, o algoritmo de ranqueamento, as métricas para

avaliar o erro na calibragem e o protocolo de decisão. Ao final, as principais contribuições desta dissertação foram destacadas.

No Capítulo 5 foram apresentadas: as bases de dados e a quantidade de dados dada como entrada para o sistema; em seguida as bases de dados foram analisadas a partir da popularidade e do gênero; a metodologia aplicada no sistema foi apresentada juntamente com os valores das variáveis do sistema e de seus algoritmos; as métricas de avaliação também foram escolhidas e debatidas; a seguir foram apresentados os resultados, demonstrando-os em forma de gráficos, para que assim sejam: analisados, debatidos e comparados com outros trabalhos no estado-da-arte.

6.1 VISÃO GERAL

Este trabalho teve como objetivo principal desenvolver um sistema de recomendação baseado em filtragem colaborativa que busca ser justo com as preferências dos usuários. Por exemplo, se as preferências musicais do usuário possuem 70% de Arrocha e 30% de Rock, é esperado que o sistema de recomendação proporcione uma lista de recomendação que seja semelhante à proporção nas preferências.

A relevância do problema abordado por este trabalho tem como base o respeito pelas preferências dos usuários, sendo que os algoritmos recomendadores da técnica de filtragem colaborativa sofrem com enviesamentos, principalmente com o viés de popularidade, que ignora as preferências dos usuários, recomendando-lhes itens mais populares. Isto causa uma desproporcionalidade nas preferências ou até um subjugamento de determinadas preferências do usuário.

A partir do modelo de sistema proposto foram implementados 1.404 sistemas diferentes, que foram avaliados por quatro métricas: Mean Average Precision (MAP), Mean Reciprocal Rank (MRR), Mean Average Calibration Error (MACE) e Mean Rank Miss Calibration (MRMC). Estes resultados diferentes vêm da combinação das formulações apresentadas ao longo deste trabalho: duas bases de dados, nove algoritmos recomendadores, três medidas de divergência, dois balanceamentos relevância-divergência, treze pesos de balanceamento (dois personalizados e onze fixos), uma medida de ranque e um algoritmo de ranqueamento.

Ao observar os resultados a partir da precisão, viu-se que o pós-processamento pode propiciar uma melhora no desempenho do sistema, sendo que a maioria dos resultados apontam para este fato. As propostas contidas nesta dissertação, como os balanceamentos e os pesos do balanceamento, apresentaram ou um incremento no desempenho do sistema ou mantiveram desempenhos já alcançados. A medida de justiça chamada Pearson Chi Square, proposta nesta dissertação para o uso como medida de justiça, apresentou um desempenho competitivo e em alguns momentos melhor que as outras medidas.

As métricas propostas nesta dissertação demonstram a sua efetividade em capturar os comportamentos do sistema e metrificar os resultados, auxiliando na leitura e interpretação. As análises apontam que as métricas conseguem captar comportamentos que outras métricas também capturam. Os coeficientes decisórios que compõem a proposta produzem um meio de afirmar qual a melhor combinação de sistema.

Os resultados indicaram que cada algoritmo recomendador, utilizado durante a etapa

de processamento, apresenta um comportamento diferente, existindo alguns comportamentos em comum. A base de dados utilizada também cria a possibilidade de comportamentos diferentes, assim, indicando que para cada base de dados a combinação de sistema que obterá o melhor desempenho pode vir a ser diferente, considerando que a avaliação seja para além da precisão e considere métricas de justiça na decisão da combinação de sistema que será usada.

Tendo em vista os aspectos observados, o escopo das áreas investigadas no desenvolvimento deste trabalho são: Sistemas de Recomendação, Justiça e Métricas de Avaliação. Pode-se ressaltar que as contribuições aqui obtidas são um acréscimo significativo para a área de justiça nas recomendações, bem como para áreas correlacionadas.

6.2 CONTRIBUIÇÕES DO TRABALHO

Ao longo da dissertação diversas propostas são apresentadas e posteriormente avaliadas. Os resultados demonstram que a pesquisa avançou com melhorias sobre o tema e contribuiu com a comunidade científica de Sistemas de Recomendação e Calibragem, explorando um tema recente e pouco desenvolvido. A seguir é apresentada uma lista com as contribuições inéditas desta pesquisa.

- **Um algoritmo de calibragem** é apresentado durante a descrição da proposta, este que é de base genérica e pode ser adaptado a qualquer nova medida de divergência, medida de relevância, pesos de balanceamento, dentre outros algoritmos apresentados;
- **Dois pesos do balanceamento relevância-divergência** são apresentados neste trabalho, destacando o ineditismo de pesos personalizados para uma calibragem mais respeitosa;
- Uma **formulação de balanceamento relevância-divergência** é apresentada durante esta dissertação, visando incrementar o desempenho do sistema a partir da influência do viés do usuário durante a criação da lista de recomendação;
- **Duas novas métricas de avaliação** para calcular o erro da calibragem são apresentadas neste trabalho, sendo esta pesquisa a primeira com uma proposta de nova classe de métricas, denominadas erro na calibragem;
- **Dois coeficientes decisórios** para auxiliar no entendimento do melhor resultado ao longo das combinações de sistemas;
- As **combinações de sistema** são das mais variadas, totalizando mais de 1.400 resultados possíveis;
- **Um protocolo decisório para sistemas de recomendação calibrado** é proposto no corpo desta dissertação, visando padronizar e descrever os componentes necessários para gerar um sistema de recomendação calibrado;
- **Um modelo de sistema de recomendação calibrado** é parte da proposta desta dissertação, visando facilitar implementações futuras.

6.3 SELO DE REPRODUTIBILIDADE, PUBLICAÇÕES E CÓDIGO

O sistema proposto foi concebido em duas partes. A primeira parte contendo as propostas do Capítulo 4, com exceção da formulação do balanceamento logarítmico. A segunda parte envolve todas as propostas apresentadas nesta pesquisa e analisadas no Capítulo 5.

A primeira parte do sistema proposto está publicada na revista Qualis A1, *Expert System With Applications* - (ESWA), no artigo de Silva, Manzato e Durão (2021a). Durante o processo da primeira parte, o código-fonte foi submetido para avaliação de reprodutibilidade dos experimentos, checagem realizada para saber se é possível reproduzir os passos dados para a obtenção dos resultados. Assim, ao término da avaliação o sistema proposto ganhou selo de reprodutibilidade e o código está em um contêiner[1] pronto e preparado para executar em Silva, Manzato e Durão (2021b).

A segunda parte da implementação é o sistema proposto completo. Os detalhes da implementação e os resultados alcançados estão presentes nesta dissertação. O código-fonte está disponível no repositório do Github[2].

6.4 TRABALHOS FUTUROS

O desenvolvimento e avaliação do sistema proporcionaram uma série de observações ao longo deste trabalho. Sendo assim, diante das várias mudanças que podem ser feitas, percebe-se que há espaço para sua continuação e melhoria. A seguir são apresentadas algumas sugestões de continuidade da pesquisa:

- A partida a frio pode ser estudada no contexto da calibragem. Os usuários que recém aderiram ao sistema ou itens da cauda longa podem ser estudados de forma a entender o efeito da calibragem sobre esses grupos. Os resultados apresentados nesta dissertação demonstraram a possibilidade de considerar a calibragem sem perder a precisão e em alguns casos obtendo melhor desempenho. Assim, pode-se realizar um estudo para entender se nos usuários que estão na partida a frio, a calibragem consegue traçar melhor o perfil do usuário, abrangendo todas as áreas de interesse sem perder precisão;

- Uso de algoritmos recomendadores baseados em redes neurais. Durante o desenvolvimento do sistema os algoritmos usados são baseados nos trabalhos relacionados. Como é possível verificar, nenhum algoritmo recomendador baseado em redes neurais foi utilizado. Assim é possível desenvolver um estudo em que todos os tópicos apresentados neste trabalho sejam utilizados em recomendadores baseados em redes neurais ou em redes neurais profundas. Durante este trabalho futuro, é possível estudar o grau de descalibragem das redes neurais sem aplicação do pós-processamento, ou seja, entender se, e o quanto, a lista de recomendação gerada por estes recomendadores é descalibrada. Em seguida, é possível aplicar o pós-processamento e obter o entendimento sobre o comportamento da precisão e da calibragem nas listas de recomendações. Esta pesquisa pode tomar como base duas

[1] https://doi.org/10.24433/CO.6790880.v1
[2] https://github.com/DiegoCorrea/masters-dissertation

diretrizes: 1) qual o grupo de redes neurais que naturalmente promove calibragem? e 2) qual o grupo possui o melhor desempenho com o pós-processamento?;

- Os recomendadores podem ser avaliados a partir dos grupos aos quais pertencem, por exemplo, pode-se avaliar as variações do K Nearest Neighbors (KNN) para o usuário e para os itens, ou explorar os resultados em outros fatoradores de matrizes e comparar seus resultados. Como é possível verificar o Item-KNN obteve o melhor desempenho na base de dados do One Million Songs (OMS) e o Singular Value Decomposition Plus Plus (SVD++) obteve o melhor desempenho na base de dados do *Movielens*. Ambos os recomendadores pertencem a diferentes grupos, o Item-KNN pertence ao grupo de algoritmos KNN e o SVD++ pertence ao grupo dos fatoradores de matriz. Cada um dos grupos possuem diversos algoritmos, assim é possível desenvolver uma pesquisa onde a calibragem seja estudada de forma a entender o comportamento de um grupo de recomendadores. Por exemplo, iniciar com a pergunta sobre o que levou o Item-KNN a obter melhor desempenho que o User-KNN. É possível que os KNN baseados na média obtenham melhor desempenho? E os KNNs baseados no índice Z, obtêm melhores resultados?;

- Como a calibragem é aplicada em uma etapa de pós-processamento é possível utilizar qualquer técnica de recomendação na etapa de processamento. Assim, um trabalho futuro é entender como, por exemplo, as técnicas de filtragem baseada em conteúdo ou a híbrida se comportam com o pós-processamento visando à calibragem. É possível desenvolver uma pesquisa que use diversos recomendadores de filtragem baseada em conteúdo, utilizando os gêneros como informação apenas no pós-processamento. Perguntas que podem guiar esta pesquisa são: 1) a aplicação do pós-processamento obtém resultados semelhantes ou diferentes da filtragem colaborativa? 2) qual o nível de descalibragem desses recomendadores baseados no conteúdo? e 3) o problema da superespecialização é diminuído ou amplificado com a calibragem?;

- A distribuição de gêneros apresentada trabalha com o peso do feedback e a probabilidade de cada gênero no item. Tal equação pode ser incrementada com outros pesos, por exemplo, baseado no tempo, assim como pode-se alterar de probabilidade para outros entendimentos sobre o valor do gênero no item. A depender do domínio da aplicação o momento temporal possui uma carga de informação útil para o sistema. A preferência do usuário pode ser modelada como uma série temporal, levando o tema da calibragem a considerar a evolução da distribuição dos gêneros ao longo do tempo. Nesta dissertação e no estado-da-arte usa-se a probabilidade do gênero no item na formulação da distribuição, em especial o trabalho de Kaya e Bridge (2019) trabalha com sub-perfis além da probabilidade do gênero no item. É possível utilizar outras formulações sobre o entendimento do gênero, como por exemplo, a entropia, assim como é possível utilizar uma probabilidade do gênero em todos os itens ou a probabilidade do gênero em todas as preferências (transações);

- As propostas de peso do balanceamento podem ser ampliadas e novos pesos podem ser utilizados, por exemplo, o uso da esperança sobre a distribuição de gêneros. A

partir da variância, proposta desta dissertação, é possível obter o desvio padrão, adicionando outra possibilidade de encontrar o peso personalizado. Quando um usuário foca em um gênero, este obtém um alto valor da distribuição, criando um ambiente onde os gêneros menos preferidos obtêm valores baixos da distribuição. A partir esta afirmação é possível utilizar a amplitude ou a média das amplitudes entre todos os itens. A possibilidade de diversos pesos personalizados amplia o entendimento sobre a calibragem, assim as seguintes perguntas podem direcionar esta pesquisa: 1) qual é o melhor peso personalizado? 2) a depender do domínio o melhor desempenho muda? e 3) existe algum peso personalizado que sempre obtenha melhor desempenho que os pesos constantes?;

- Outras medidas de divergência podem ser utilizadas, ampliando as possibilidades de obter um melhor desempenho. Existem mais de 50 medidas de divergência, que são classificadas em famílias, como explanado nesta dissertação. É possível desenvolver um estudo que selecione e explore as melhores medidas, melhorando o desempenho do sistema. É verificável nos resultados desta dissertação que o domínio influi nos desempenhos das medidas. Assim, perguntas que podem servir de diretriz a este estudo são: 1) em bases do mesmo domínio, as medidas de divergência mantêm desempenho semelhante? 2) Em cada domínio, qual é a melhor medida? 3) qual é a família de medidas que obtém o melhor desempenho? e 4) existem medidas não recomendáveis para o contexto da calibragem?;

- Outros algoritmos de ranqueamento podem ser utilizados. É possível desenvolver um estudo que filtre e selecione algoritmos de ranqueamento, buscando comparar os resultados obtidos. O problema do ranqueamento pode ser remodelado e entendido a partir de problemas provindos de algoritmos clássicos e grafos. Por exemplo, é possível utilizar o algoritmo *Cost Effective Lazy Forward* (CELF) ou a melhoria sobre o CELF, chamado de CELF++. Perguntas que podem guiar este estudo são: 1) é possível readequar o problema de gerar uma lista de recomendação justa em um problema clássico de algoritmo ou de grafos? 2) há melhores desempenhos que o *surrogate*? e 3) como estes algoritmos se comportam quando observados a partir do erro na calibragem?.

6.5 CONSIDERAÇÕES FINAIS

Os sistemas de recomendação modernos avaliam seus resultados para além da precisão, buscando apresentar múltiplos olhares do resultado. Assim, proporcionar recomendações mais justas e respeitosas com as preferências dos usuários é um tema importante à comunidade, devido à proposta de personalização aplicada pelos sistemas de recomendação modernos.

Este trabalho propôs, implementou e avaliou um sistema de recomendação que buscasse abranger e respeitar os interesses do usuário. Os resultados indicaram que a calibragem é um meio de promover justiça na lista de recomendação personalizada. As análises mostram que é possível melhorar o desempenho do sistema ao considerar a justiça através da calibragem.

Com isso, aos pesquisadores da área é adicionada mais uma visão sobre como construir um sistema de recomendação, esperando, assim, que o uso de algoritmos recomendadores sejam incrementados com um maior nível de justiça e respeito por aqueles aos quais desejamos agradar, os nossos usuários.

REFERÊNCIAS BIBLIOGRÁFICAS

ABDOLLAHPOURI, H.; BURKE, R. Multi-stakeholder recommendation and its connection to multi-sided fairness. *ArXiv*, abs/1907.13158, 2019. Disponível em: ⟨https://arxiv.org/abs/1907.13158⟩.

ABDOLLAHPOURI, H.; BURKE, R.; MOBASHER, B. Controlling popularity bias in learning-to-rank recommendation. In: *Proceedings of the Eleventh ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2017. (RecSys '17), p. 42–46. ISBN 978-1-4503-4652-8. Disponível em: ⟨http://doi.acm.org/10.1145/3109859.3109912⟩.

ABDOLLAHPOURI, H. et al. The impact of popularity bias on fairness and calibration in recommendation. *ArXiv*, abs/1910.05755, 10 2019. Disponível em: ⟨https://arxiv.org/abs/1910.05755⟩.

ABDOLLAHPOURI, H. et al. The unfairness of popularity bias in recommendation. *ArXiv*, abs/1907.13286, 2019. Disponível em: ⟨https://arxiv.org/abs/1907.13286⟩.

ABDOLLAHPOURI, H. et al. The connection between popularity bias, calibration, and fairness in recommendation. In: *Fourteenth ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2020. (RecSys '20), p. 726–731. ISBN 9781450375832. Disponível em: ⟨https://doi.org/10.1145/3383313.3418487⟩.

ADOMAVICIUS, G.; TUZHILIN, A. Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge & Data Engineering*, IEEE, v. 17, n. 6, p. 734–749, 07 2005. Disponível em: ⟨https://ieeexplore.ieee.org/document/1423975⟩.

BARJASTEH, I. et al. Cold-start item and user recommendation with decoupled completion and transduction. In: *Proceedings of the 9th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2015. (RecSys '15), p. 91–98. ISBN 9781450336925. Disponível em: ⟨https://doi.org/10.1145/2792838.2800196⟩.

BERTIN-MAHIEUX, T. et al. The million song dataset. In: KLAPURI, A.; LEIDER, C. (Ed.). *Proceedings of the 12th International Conference on Music Information Retrieval (ISMIR 2011)*. Miami, Florida, USA: University of Miami, 2011. p. 591–596. Disponível em: ⟨http://ismir2011.ismir.net/papers/OS6-1.pdf⟩.

BLEI, D. M.; NG, A. Y.; JORDAN, M. I. Latent dirichlet allocation. *J. Mach. Learn. Res.*, JMLR.org, v. 3, p. 993–1022, 3 2003. ISSN 1532-4435. Disponível em: ⟨https://dl.acm.org/doi/10.5555/944919.944937⟩.

BURKE, R. Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, Kluwer Academic Publishers, Hingham, MA, USA, v. 12, n. 4, p. 331–370, nov. 2002. ISSN 0924-1868. Disponível em: ⟨http://dx.doi.org/10.1023/A:1021240730564⟩.

BURKE, R. Multisided fairness for recommendation. *CoRR*, abs/1707.00093, 2017. Disponível em: ⟨http://arxiv.org/abs/1707.00093⟩.

CARBONELL, J.; GOLDSTEIN, J. The use of mmr, diversity-based reranking for reordering documents and producing summaries. In: *Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*. New York, NY, USA: ACM, 1998. (SIGIR '98), p. 335–336. ISBN 1-58113-015-5. Disponível em: ⟨http://doi.acm.org/10.1145/290941.291025⟩.

CELMA, Ò. *Music Recommendation and Discovery in the Long Tail*. 210 p. Tese (Doutorado) — Universitat Pompeu Fabra, Barcelona, 2009. Disponível em: ⟨http://hdl.handle.net/10803/7557⟩.

CHA, S.-H. Comprehensive survey on distance/similarity measures between probability density functions. *International Journal of Mathematical Models and Methods in Applied Sciences*, v. 1, n. 4, p. 300–307, 04 2007. Disponível em: ⟨http://www.gly.fsu.edu/~parker/geostats/Cha.pdf⟩.

CHENG, P. et al. Learning to recommend accurate and diverse items. In: *Proceedings of the 26th International Conference on World Wide Web*. Republic and Canton of Geneva, CHE: International World Wide Web Conferences Steering Committee, 2017. (WWW '17), p. 183–192. ISBN 9781450349130. Disponível em: ⟨https://doi.org/10.1145/3038912.3052585⟩.

DESROSIERS, C.; KARYPIS, G. A comprehensive survey of neighborhood-based recommendation methods. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. Springer, 2011. p. 107–144. ISBN 978-0-387-85820-3. Disponível em: ⟨https://doi.org/10.1007/978-0-387-85820-3_4⟩.

FIELDS, B. *Contextualize Your Listening : The Playlist as Recommendation Engine*. 164 p. Tese (Doutorado) — Department of Computing, Goldsmiths, University of London, London, 2011. Disponível em: ⟨http://research.gold.ac.uk/id/eprint/6477/⟩.

George, T.; Merugu, S. A scalable collaborative filtering framework based on co-clustering. In: *Fifth IEEE International Conference on Data Mining (ICDM'05)*. USA: IEEE Computer Society, 2005. (ICDM '05), p. 625–628. ISBN 0-7695-2278-5. Disponível em: ⟨https://doi.org/10.1109/ICDM.2005.14⟩.

HARDT, M.; PRICE, E.; SREBRO, N. Equality of opportunity in supervised learning. In: *Proceedings of the 30th International Conference on Neural Information Processing Systems*. USA: Curran Associates Inc., 2016. (NIPS'16), p. 3323–3331. ISBN 978-1-5108-3881-9. Disponível em: ⟨http://dl.acm.org/citation.cfm?id=3157382.3157469⟩.

HARPER, F. M.; KONSTAN, J. A. The movielens datasets: History and context. *ACM Trans. Interact. Intell. Syst.*, ACM, New York, NY, USA, v. 5, n. 4, p. 1–19, 12 2015. ISSN 2160-6455. Disponível em: ⟨http://doi.acm.org/10.1145/2827872⟩.

HELLINGER, E. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, v. 136, p. 210–271, 1909. Disponível em: ⟨http://eudml.org/doc/149313⟩.

HERLOCKER, J. L. et al. Evaluating collaborative filtering recommender systems. *ACM Trans. Inf. Syst.*, Association for Computing Machinery, New York, NY, USA, v. 22, n. 1, p. 5–53, jan 2004. ISSN 1046-8188. Disponível em: ⟨https://doi.org/10.1145/963770.963772⟩.

HIJIKATA, Y.; IWAHAMA, K.; NISHIDA, S. Content-based music filtering system with editable user profile. In: *Proceedings of the 2006 ACM Symposium on Applied Computing*. New York, NY, USA: ACM, 2006. (SAC '06), p. 1050–1057. ISBN 1-59593-108-2. Disponível em: ⟨http://doi.acm.org/10.1145/1141277.1141526⟩.

HUG, N. Surprise, a Python library for recommender systems. 2017.

ISINKAYE, F.; FOLAJIMI, Y.; OJOKOH, B. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, v. 16, n. 3, p. 261 – 273, 2015. ISSN 1110-8665. Disponível em: ⟨http://www.sciencedirect.com/science/article/pii/S1110866515000341⟩.

JOLAD, S. et al. A new family of bounded divergence measures and application to signal detection. *Proceedings of the 5th International Conference on Pattern Recognition Applications and Methods*, SCITEPRESS - Science and and Technology Publications, 2016. Disponível em: ⟨http://dx.doi.org/10.5220/0005695200720083⟩.

KAMISHIMA, T.; AKAHO, S.; ASOH, H. Enhancement of the neutrality in recommendation. In: *Proc. of the 2nd Workshop on Human Decision Making in Recommender Systems*. New York, NY, USA: ACM, 2012. p. 8–14. Disponível em: ⟨http://ceur-ws.org/Vol-893/paper2.pdf⟩.

KAYA, M.; BRIDGE, D. A comparison of calibrated and intent-aware recommendations. In: *Proceedings of the 13th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2019. (RecSys '19), p. 151–159. ISBN 9781450362436. Disponível em: ⟨https://doi.org/10.1145/3298689.3347045⟩.

KOKOSKA, S.; ZWILLINGER, D. *CRC Standard Probability and Statistics Tables and Formulae, Student Edition*. Boca Raton: CRC Press, 1999. ISBN 9780429181467.

KOREN, Y.; BELL, R. Advances in collaborative filtering. In: RICCI, F.; ROKACH, L.; SHAPIRA, B. (Ed.). *Recommender Systems Handbook*. Boston, MA: Springer US, 2015. p. 77–118. ISBN 978-1-4899-7637-6. Disponível em: ⟨https://doi.org/10.1007/978-1-4899-7637-6_3⟩.

Koren, Y.; Bell, R.; Volinsky, C. Matrix factorization techniques for recommender systems. *Computer*, v. 42, n. 8, p. 30–37, 2009. Disponível em: ⟨https://ieeexplore.ieee.org/document/5197422?arnumber=5197422⟩.

KULLBACK, S.; LEIBLER, R. A. On information and sufficiency. *Ann. Math. Statist.*, The Institute of Mathematical Statistics, v. 22, n. 1, p. 79–86, 03 1951. Disponível em: ⟨https://doi.org/10.1214/aoms/1177729694⟩.

LEMIRE, D.; MACLACHLAN, A. Slope one predictors for online rating-based collaborative filtering. *Proceedings of the 2005 SIAM International Conference on Data Mining*, v. 5, p. 471–475, 02 2007. Disponível em: ⟨https://epubs.siam.org/doi/abs/10.1137/1.9781611972757.43⟩.

LIANG, D. et al. Factorization meets the item embedding: Regularizing matrix factorization with item co-occurrence. In: *Proceedings of the 10th ACM Conference on Recommender Systems*. New York, NY, USA: Association for Computing Machinery, 2016. (RecSys '16), p. 59–66. ISBN 9781450340359. Disponível em: ⟨https://doi.org/10.1145/2959100.2959182⟩.

LIN, K. et al. Crank up the volume: preference bias amplification in collaborative recommendation. *CoRR*, abs/1909.06362, 2019. Disponível em: ⟨http://arxiv.org/abs/1909.06362⟩.

LIN, K. et al. Calibration in collaborative filtering recommender systems: A user-centered analysis. In: *Proceedings of the 31st ACM Conference on Hypertext and Social Media*. New York, NY, USA: Association for Computing Machinery, 2020. (HT '20), p. 197–206. ISBN 9781450370981. Disponível em: ⟨https://doi.org/10.1145/3372923.3404793⟩.

LIU, W.; BURKE, R. D. Personalizing fairness-aware re-ranking. *ArXiv*, abs/1809.02921, 2018. Disponível em: ⟨https://arxiv.org/abs/1809.02921⟩.

LOPS, P.; GEMMIS, M. de; SEMERARO, G. Content-based recommender systems: State of the art and trends. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. Springer US, 2011. p. 73–105. ISBN 978-0-387-85819-7. Disponível em: ⟨http://dx.doi.org/10.1007/978-0-387-85820-3_3⟩.

LUCIAN, R. Repensando o uso da escala likert: Tradição ou escolha técnica? *PMKT - Revista Brasileira de Pesquisa de Marketing, Opinião e Mídia.*, v. 18, p. 13–32, 04 2016. Disponível em: ⟨http://www.revistapmkt.com.br/pt-br/anteriores/anteriores-7.html?udt_863_param_detail=8640⟩.

Luo, X. et al. An efficient non-negative matrix-factorization-based approach to collaborative filtering for recommender systems. *IEEE Transactions on Industrial Informatics*, v. 10, n. 2, p. 1273–1284, 2014. Disponível em: ⟨https://ieeexplore.ieee.org/document/6748996⟩.

MANNING, C. D.; RAGHAVAN, P.; SCHüTZE, H. *Introduction to Information Retrieval*. New York, NY, USA: Cambridge University Press, 2008. ISBN 0521865719, 9780521865715. Disponível em: ⟨https://nlp.stanford.edu/IR-book/pdf/irbookonlinereading.pdf⟩.

MCFEE, B. et al. The million song dataset challenge. In: *Proceedings of the 21st International Conference on World Wide Web*. New York, NY, USA: Association for Computing Machinery, 2012. (WWW '12 Companion), p. 909–916. ISBN 9781450312301. Disponível em: ⟨https://doi.org/10.1145/2187980.2188222⟩.

MCNEE, S. M.; RIEDL, J.; KONSTAN, J. A. Being accurate is not enough: How accuracy metrics have hurt recommender systems. In: *CHI '06 Extended Abstracts on Human Factors in Computing Systems*. New York, NY, USA: Association for Computing Machinery, 2006. (CHI EA '06), p. 1097–1101. ISBN 1595932984. Disponível em: ⟨https://doi.org/10.1145/1125451.1125659⟩.

MEHROTRA, R. et al. Towards a fair marketplace: Counterfactual evaluation of the trade-off between relevance, fairness and satisfaction in recommendation systems. In: . New York, NY, USA: Association for Computing Machinery, 2018. (CIKM '18), p. 2243–2251. ISBN 9781450360142. Disponível em: ⟨https://doi.org/10.1145/3269206.3272027⟩.

MITTELSTADT, B.; RUSSELL, C.; WACHTER, S. Explaining explanations in ai. In: *Proceedings of the Conference on Fairness, Accountability, and Transparency*. New York, NY, USA: Association for Computing Machinery, 2019. (FAT* '19), p. 279–288. ISBN 9781450361255. Disponível em: ⟨https://doi.org/10.1145/3287560.3287574⟩.

NEMHAUSER, G. L.; WOLSEY, L. A.; FISHER, M. L. An analysis of approximations for maximizing submodular set functions–i. *Math. Program.*, Springer-Verlag New York, Inc., Secaucus, NJ, USA, v. 14, n. 1, p. 265–294, dez. 1978. ISSN 0025-5610. Disponível em: ⟨https://doi.org/10.1007/BF01588971⟩.

PARRA, D.; SAHEBI, S. Recommender systems: Sources of knowledge and evaluation metrics. In: VELáSQUEZ, J. D.; PALADE, V.; JAIN, L. C. (Ed.). *Advanced Techniques in Web Intelligence-2*. Springer Berlin Heidelberg, 2013, (Studies in Computational Intelligence, v. 452). p. 149–175. ISBN 978-3-642-33325-5. Disponível em: ⟨http://dx.doi.org/10.1007/978-3-642-33326-2_7⟩.

PESKA, L.; VOJTAS, P. Enhancing recommender system with linked open data. In: LARSEN, H. L. et al. (Ed.). *Flexible Query Answering Systems*. Berlin, Heidelberg: Springer Berlin Heidelberg, 2013. p. 483–494. ISBN 978-3-642-40769-7. Disponível em: ⟨https://link.springer.com/chapter/10.1007/978-3-642-40769-7_42⟩.

QIN, Y. *A Historical Survey of Music Recommendation Systems: Towards Evaluation*. Tese (Doutorado) — McGill University Libraries, 2013.

RESNICK, P.; VARIAN, H. R. Recommender systems. *Commun. ACM*, ACM, New York, NY, USA, v. 40, n. 3, p. 56–58, mar. 1997. ISSN 0001-0782. Disponível em: ⟨http://doi.acm.org/10.1145/245108.245121⟩.

RICCI, F.; ROKACH, L.; SHAPIRA, B. Introduction to recommender systems handbook. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. Boston, MA: Springer US, 2011. p. 1–35. ISBN 978-0-387-85820-3. Disponível em: ⟨https://doi.org/10.1007/978-0-387-85820-3_1⟩.

SACHARIDIS, D.; MOURATIDIS, K.; KLEFTOGIANNIS, D. A common approach for consumer and provider fairness in recommendations. In: *Proceedings of the 13th ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2019. (RecSys '19). Disponível em: ⟨http://ceur-ws.org/Vol-2431/paper1.pdf⟩.

SHANI, G.; GUNAWARDANA, A. Evaluating recommendation systems. In: RICCI, F. et al. (Ed.). *Recommender Systems Handbook*. Boston, MA: Springer US, 2011. p. 257–297. ISBN 978-0-387-85820-3. Disponível em: ⟨https://doi.org/10.1007/978-0-387-85820-3_8⟩.

SILVA, D. C. da; MANZATO, M. G.; DURãO, F. A. Exploiting personalized calibration and metrics for fairness recommendation. *Expert Systems with Applications*, p. 115112, 2021. ISSN 0957-4174. Disponível em: ⟨https://www.sciencedirect.com/science/article/pii/S0957417421005534⟩.

SILVA, D. C. da; MANZATO, M. G.; DURãO, F. A. Exploiting personalized calibration and metrics for fairness recommendation. *CodeOcean*, 2021. Disponível em: ⟨https://doi.org/10.24433/CO.6790880.v1⟩.

SILVEIRA, T. et al. How good your recommender system is? a survey on evaluations in recommendation. *International Journal of Machine Learning and Cybernetics*, v. 10, p. 813–831, 12 2017. Disponível em: ⟨https://doi.org/10.1007/s13042-017-0762-9⟩.

STECK, H. Calibrated recommendations. In: *Proceedings of the 12th ACM Conference on Recommender Systems*. New York, NY, USA: ACM, 2018. (RecSys '18), p. 154–162. ISBN 978-1-4503-5901-6. Disponível em: ⟨http://doi.acm.org/10.1145/3240323.3240372⟩.

SU, X.; KHOSHGOFTAAR, T. M. A survey of collaborative filtering techniques. *Adv. in Artif. Intell.*, Hindawi Limited, London, GBR, v. 2009, jan. 2009. ISSN 1687-7470. Disponível em: ⟨https://doi.org/10.1155/2009/421425⟩.

YANG, K.; STOYANOVICH, J. Measuring fairness in ranked outputs. In: *Proceedings of the 29th International Conference on Scientific and Statistical Database Management*. New York, NY, USA: ACM, 2017. (SSDBM '17), p. 22:1–22:6. ISBN 978-1-4503-5282-6. Disponível em: ⟨http://doi.acm.org/10.1145/3085504.3085526⟩.

ZHANG, S. et al. Learning from incomplete ratings using non-negative matrix factorization. In: SIAM. *In Proc. of the 6th SIAM Conference on Data Mining (SDM*. 2006. v. 6,

p. 549–553. ISBN 978-0-89871-611-5. Disponível em: ⟨https://archive.siam.org/meetings/sdm06/proceedings/059zhangs2.pdf⟩.

ZLIOBAITE, I. A survey on measuring indirect discrimination in machine learning. *arXiv e-prints*, p. arXiv:1511.00148, Oct 2015. Disponível em: ⟨https://arxiv.org/abs/1511.00148⟩.