Algoritmos de Recomendação: Propostas para Avaliação das Recomendações de Itens Únicos ou Lista de Itens

Guilherme B. Souto[1], **Renato M. Filho**[1], **Carlos A. Silva**[1]

[1]Departamento de Informática – Instituto Federal de Minas Gerais (IFMG)
CEP 34.590-390 – Sabará, MG – Brasil

guilhermeqbs@gmail.com,{renato.miranda,carlos.silva}@ifmg.edu.br

Abstract. *Recommendation algorithms play a role in improving and automating the task of making good recommendations. However, quantitatively measuring the quality of recommendations can be too complex. In this work, we propose two metrics to evaluate the result of recommendation algorithms: the Single Recommended Item Metric (SRIM) and List of Recommended Items Metric (LRIM). In the first, the purpose is to evaluate recommendation algorithms that return only one item, while the second has the objective of evaluating recommendations of multiple items. In this way, we apply the proposed metrics to evaluate such recommendation algorithms using a classic movie database. The results show that the AR2 recommendation algorithm provides more relevant recommendations than the AR2 algorithm according to the SRIM and LRIM metrics. In addition, we detected that the AR2 algorithm outperforms by just over 8% compared to AR1.*

Resumo. *Os algoritmos de recomendação desempenham o papel de aperfeiçoar e automatizar a tarefa de se fazer boas recomendações. No entanto, mensurar de forma quantitativa a qualidade das recomendações pode ser demasiadamente complexo. Neste trabalho, propomos duas métricas para avaliar o resultado de algoritmos de recomendação: a Single Recommended Item Metric (SRIM) e List of Recommended Items Metric (LRIM). Na primeira, o propósito é avaliar algoritmos de recomendação que retornam apenas um item, enquanto a segunda tem o objetivo de avaliar recomendações de múltiplos itens. Desta forma, nós aplicamos as métricas propostas para avaliar tais algoritmos de recomendação utilizando uma clássica base de dados de filmes. Os resultados mostram que o algoritmo de recomendação AR2 provê recomendações mais relevantes que o algoritmo AR1 de acordo com as métricas SRIM e LRIM. Além disso, no LRIM detectamos que o algoritmo AR2 supera em pouco mais de 8% em relação ao AR1.*

1. Introdução

O volume de informação disponível na internet tem crescido consideravelmente nos últimos anos, contribuindo com a ampliação do acesso ao conhecimento. Porém, tal fato, tem representado um fator de dificuldade aos usuários para definir suas escolhas de consumo, desde notícias até produtos. Neste contexto, os algoritmos de recomendação surgem com o papel de facilitar o processo de escolha a itens relevantes aos usuários [Marinho et al. 2019, Lorenção and Santos 2021].

Com a digitalização do consumo, empresas atuantes na *web* viram a necessidade de automatizar este processo de recomendação em grande escala, com o objetivo de maximizar o alcance de seus produtos. Presentes em nosso cotidiano, podemos citar a Netflix oferecendo filmes e séries, o Youtube com conteúdo de seus vídeos, a Amazon vendendo livros e produtos diversos ou o Facebook definindo qual postagem deve aparecer para certo usuário [de Oliveira Yamaguti and Coello 2013, Himel et al. 2017].

Em outras palavras, os algoritmos de recomendação buscam automatizar o processo de recomendação mediante a grande quantidade de informações. Esses algoritmos podem basicamente recomendar qualquer tipo de item para qualquer público que esteja na internet e possua dados suficientes dos mesmos [Marinho et al. 2019]. A recomendação de um item é feita a partir dos dados sobre o consumo do item ou a partir de dados do usuário. O item a ser recomendado por um algoritmo pode ser um produto, filme, música, livro, *post* em rede social, dentre outros. A recomendação pode ser feita de diversas formas, seja por avaliações dos usuários aos itens, pelas semelhanças entre as características dos itens ou entre os históricos de consumo dos usuários [Himel et al. 2017].

Há diversas formas de se fazer uma recomendação levando em conta o usuário/item. Neste sentido seguem as abordagens dos algoritmos de recomendação, sendo as mais tradicionais, a filtragem baseada em conteúdo, a filtragem colaborativa e a filtragem híbrida [Silva 2017].

A filtragem baseada em conteúdo funciona através das características e conteúdos de um item, como exemplo, um livro possui autor, descrição, gênero, ano de publicação, entre outros atributos. A recomendação é feita a partir da análise do conteúdo de cada item dentre um conjunto de itens, sendo definida a proximidade entre os itens que mais se assemelham. Neste caso, a recomendação é feita a partir da análise dos atributos de todos os livros para identificar quais são mais semelhantes. A expectativa é de que os itens semelhantes tenham a mesma relevância para um usuário. Ou seja, por meio das escolhas anteriores de um usuário é possível recomendar itens semelhantes além da possibilidade de traçar um perfil de escolhas, como o gênero de filme que agrada o usuário em questão [da Silva Costa 2017, Silva 2017].

A recomendação baseada na filtragem colaborativa é realizada levando em conta usuários com o histórico de consumo/avaliação semelhantes. Se o usuário 1 consumir os itens A, B e C e o usuário 2 consumir os itens A e C, possivelmente o item B será uma recomendação relevante para o usuário 2. Quanto maior a quantidade de dados sobre essas interações usuário/item, maior será o potencial de acerto da recomendação. Por isso, um fator negativo deste tipo de recomendação é o ponto de partida do algoritmo com poucos dados ou um novo usuário que não possui ainda um histórico [Silva 2017, Ramos 2010, de Oliveira Yamaguti and Coello 2013].

A filtragem híbrida une os dois métodos de recomendação, a filtragem colaborativa e a filtragem baseada em conteúdo, para que um método posssa complementar o outro. Além disso, há uma compensação dos pontos fracos que podem ocorrer nos dois métodos anteriores. Por exemplo, na filtragem colaborativa ocorre o problema do ponto de partida frio, correspondendo à pouca quantidade de dados de entrada para o algoritmo. Exemplos de hibridismo utilizando as duas filtragens podem ser encontrados em [Ramos 2010, Silva 2017].

Um questionamento pertinente é: como considerar que uma recomendação feita por esses algoritmos seja de fato relevante para o usuário? Existem indicadores que podem ser utilizados para aferir sobre o efeito de uma recomendação, como por exemplo, monitorar se o usuário interagiu com o item recomendado, o aumento na venda/pesquisa relacionada ao item que foi recomendado, avaliações positivas do item, dentre outros.

O objetivo deste trabalho é propor duas medidas quantitativas para avaliar a relevância das recomendações de diferentes tipos de algoritmos. A primeira medida é específica para algoritmos que recomendam apenas um item e seus resultados são expressos pelas métricas de acurácia, precisão, *recall* e *f1-score*. A segunda medida possibilita avaliar listas de recomendações de itens, ou seja, nas quais o algoritmo fornece mais de uma recomendação e seus resultados são expressos por uma mediana geral das listas únicas e suas medianas. Ambas são baseadas nas avaliações de usuários aos itens que serão recomendados e nas avaliações de itens que o algoritmo recomendou.

Este trabalho está organizado da seguinte forma. Na seção 2 é apresentado o referencial teórico descrevendo os principais trabalhos que serviram como base para o desenvolvimento desta investigação. A seção 3 descreve as duas medidas de avaliação propostas. Os resultados alcançados por meio da avaliação experimental apresentada são relatados na seção 4. Por fim, na seção 5 são apresentadas as conclusões finais e uma proposta de trabalho futuro.

2. Trabalhos relacionados

No trabalho desenvolvido por [Marinho et al. 2019] é apresentado o contexto histórico do surgimento dos algoritmos de recomendação. Assim, são mostrados os conceitos básicos dos algoritmos de recomendação e a lógica de funcionamento dos mesmos, por meio de exemplos de algoritmos aplicados na prática. Além disso, os autores abordam os três principais tipos de algoritmos de recomendação, sendo apresentados os três principais tipos: filtragem colaborativa, filtragem baseada em conteúdo e filtragem híbrida. Eles alerta sobre os eventuais problemas que podem ferir a privacidade dos dados de usuários e apresenta possíveis soluções para este problema. Por fim, o trabalho demonstra como implementar um algoritmo de recomendação utilizando o Google Colab em Python.

Em [Takahashi and Hirata Jr 2015] são propostos algoritmos de recomendação a fim de maximizar as vendas no setor de comércio eletrônico. Os algoritmos foram implementados de acordo com duas abordagens distintas, a filtragem colaborativa (*Large-scale Parallel Collaborative Filtering*) e a baseada no contexto com máquinas de fatoração (*Fast Context-aware Recommendations with Factorization Machines*). Os resultados foram avaliados utilizando uma campanha de e-mail marketing, no qual 22 grupos de usuários foram selecionados dentre um total de 10.000 usuários. Os melhores resultados foram obtidos usando a filtragem colaborativa, tendo em vista que esse algoritmo utiliza mais características implícitas do problema. Assim, são apresentados conceitos de diferentes abordagens de recomendação e proposta uma medida de avaliação para algoritmos de recomendação.

No trabalho de [Silva 2017] discutiu-se abordagens de recomendação e medidas para avaliar as recomendações. Foram implementados algoritmos de recomendação de filtragem baseada em conteúdo textual e filtragem colaborativa. A principal medida proposta, *Mean Average Precision* - MAP, avaliava a capacidade do algoritmo em apren-

der sobre as preferências do usuário, já a medida "Cobertura de Usuário" avaliava a capacidade dos algoritmos em apresentar uma quantidade significativa de itens para o usuário. Os resultados encontrados pelo MAP para a recomendação de filmes da MovieLens [Harper and Konstan 2006], demonstraram maior efetividade do algoritmo de filtragem colaborativa.

Em [de Oliveira Yamaguti and Coello 2013] são introduzidos conceitos de algoritmos de recomendação e suas variações. Os algoritmos de recomendação implementados no trabalho foram baseados em filtragem colaborativa consoante com os métodos KNN e algoritmo genético, de forma híbrida entre eles e também de forma isolada. A medida aplicada pelos algoritmos foi a *Mean Average Error* - MAE, em diferentes etapas. Foram consideradas cinco configurações de avaliações para teste, sendo que o algoritmo híbrido teve um menor MAE em todas as configurações em relação aos algoritmos isolados, pressupondo maior precisão na recomendação em relação aos demais.

Em [de Freitas Júnior et al. 2020] é utilizado o Juiz Online, um sistema que possui exercícios de programação e uma ferramenta para a autocorreção destes exercícios. Segundo os autores, o Juiz Online possui diversas opções de exercícios e de diferentes tipos, o que propicia uma dificuldade por parte dos usuários que utilizam o sistema em encontrar exercícios relevantes. Com isso, o trabalho propõe utilizar algoritmos para recomendar exercícios adequados dentro do sistema, considerando a hipótese de que recomendações possam ser feitas a partir de exercícios anteriores resolvidos por um usuário. Caso o usuário consiga resolver o exercício recomendado, é atribuído o nível de conhecimento deste usuário para a recomendação. Para validar a hipótese foram utilizados três métodos de recomendação, sendo eles baseados no enunciado, baseado no comportamento e com padrão estocástico. Foram realizados testes cegos dos métodos por três professores e 15 estudantes da área, os quais elencavam notas paras as recomendações, atingindo um total 324 recomendações. O método baseado no comportamento obteve as melhores avaliações positivas em ambos os grupos, sendo 63,3% de estudantes e 61,1% entre os professores. Outra análise constatou que os métodos baseado no comportamento e baseado no enunciado tiveram mais de 60% de seus exercícios recomendados resolvidos corretamente, já o método estocástico, apenas 25% de exercícios resolvidos.

Em [Oliveira et al. 2020] é analisada a classificação dos itens gerados por diferentes algoritmos de recomendação de filtragem colaborativa, sendo 15 no total, utilizando 7 bancos de dados e tendo os resultados combinados por diversos classificadores de agregação, sendo 19 no total. Os resultados demonstraram que a maioria dos itens relevantes estão no início da classificação e pelo estudo eles correspondem em média de 60% desses itens. A partir das classificações geradas pelos algoritmos de recomendação, os métodos de agregação de classificação foram divididos em três grupos principais, sendo o primeiro a sobreposição de classificação muito baixa e a entrada com classificações de baixa qualidade, em segundo, a sobreposição de classificação alta e a entrada com classificações de alta qualidade e por fim, a classificação de baixa sobreposição e classificações de entrada de alta qualidade. Os resultados da agregação demonstram que os grupos que possuem classificações de entrada de alta qualidade e alta diversidade, tendem a ter melhores resultados. Métodos não supervisionados não geram bons resultados quando utilizadas classificações de entrada de baixa qualidade, já métodos supervisionados e não supervisionados apresentam resultados parecidos em grupos de entrada com

classificações de alta qualidade, mas baixa diversidade. Os resultados indicam melhorias nas classificações recomendadas usando a agregação de classificação em todos os cenários incluídos pelo estudo.

As medidas propostas neste trabalho avaliam recomendações de forma quantitativa, diferente de outros trabalhos relacionados que usaram apenas abordagens qualitativas para mensurar a qualidade de explicações. Um exemplo de avaliação puramente qualitativa é realizado em [de Freitas Júnior et al. 2020] que usam avaliações de estudantes e professores da área. Métodos deste tipo tem um custo alto para agregar valor aos dados e consequentemente na maioria dos casos não são escaláveis, por exemplo, no estudo citado anteriormente apenas 324 recomendações foram avaliadas por 18 usuários. Além disso, avaliações qualitativas são mais suscetíveis ao erro devido a subjetividade humana, não sendo portanto apropriadas em cenários de avaliação de recomendações fornecidas por algoritmos.

3. Medidas de avaliação SRIM e LRIM

Este trabalho propõem duas medidas quantitativas para avaliar algoritmos de recomendação. A primeira medida, nomeada de SRIM (*Single Recommended Item Metric*), tem como objetivo mensurar a qualidade da recomendação de um único item e é baseada em métricas clássicas de avaliação em problemas de aprendizagem de máquina, tais como acurácia e precisão. A segunda medida, que recebe o nome de LRIM (*List of Recommended Items Metric*), avalia a qualidade de uma lista contendo vários itens recomendados. Nesta medida considera-se a utilização da mediana para cada lista, a fim de agregar o resultado em um único valor. E por fim é calculada a mediana geral com as medianas de cada lista única.

Como indicado na Figura 1, em ambas as propostas de medida de avaliação são necessárias como entrada o algoritmo de recomendação a ser avaliado e uma base de dados com avaliações de usuários. Neste processo alguns filtros são aplicados como, por exemplo, considerar apenas avaliações fornecidas por usuários engajados, ou seja, excluir os usuários esporádicos.

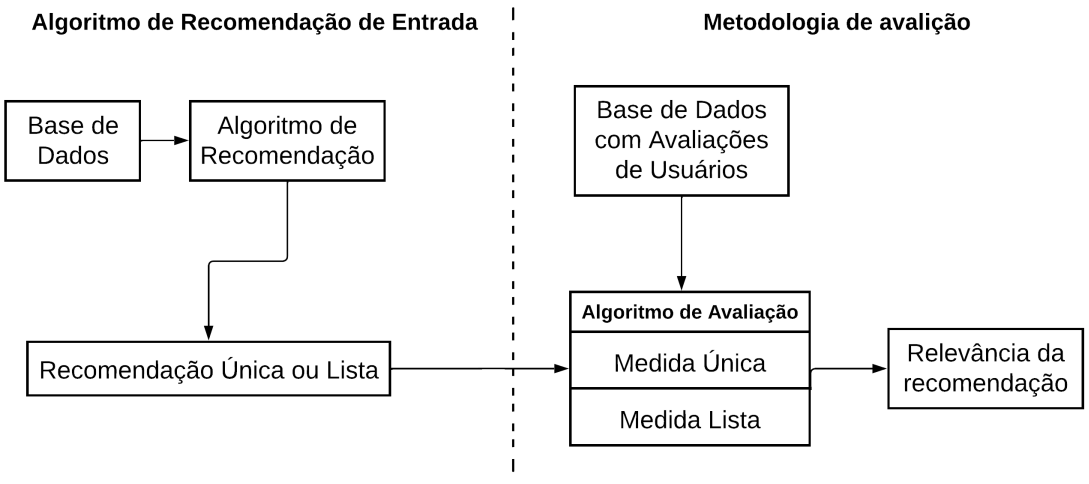


Figura 1. Fluxograma do funcionamento das Medidas

3.1. Medida SRIM - *Single Recommended Item Metric*

A medida SRIM avalia algoritmos de recomendação que a partir de um item, geram uma única recomendação. O processo dessa medida é especificado pelo Algoritmo 1, o qual recebe como entrada uma base de dados com avaliações de usuários dos respectivos itens e a função do próprio algoritmo de recomendação a ser avaliado. O resultado da medida é dado pelos cálculos das métricas acurácia, precisão, *recall* e *f1-score*.

Algoritmo 1 SRIM

```
1: for each usuário ∈ usuários do
2:     for each item ∈ usuário do
3:         itemClassificado ← classificaçãoDoItem(item)
4:         recomendação ← algoritmoDeRecomendação(itemClassificado)
5:         if recomendação ⊂ usuário.itens then
6:             coerência.append(classificaçãoItem ∧ classificaçãoRecomendação)
7:         end if
8:     end for
9: end for
10: return coerência
```

O cálculo da medida SRIM é iniciado pela inserção de cada item da base de dados no algoritmo de recomendação. Para cada item de cada usuário é realizada sua classificação (linha 3) em dois diferentes grupos: o de itens com nota alta e o de itens com nota baixa. A classificação das notas depende do padrão de notas dos itens caso a caso, por exemplo notas de 0 a 10 ou 0 a 5. Logo, no primeiro exemplo poderíamos definir que notas baixas seriam de 0 a 3 e notas altas seriam de 7 a 10.

O algoritmo de recomendação recebe o item classificado como entrada, gerando uma recomendação como saída, ou seja, um item recomendado (linha 4). Em seguida, conforme descrito na linha 5 do algoritmo SRIM, é verificado se o usuário em questão avaliou o item recomendado. Vale ressaltar que os itens recomendados não avaliados pelo usuário da base de dados são desconsiderados.

Na linha 6 é computada a coerência da recomendação feita pelo algoritmo, sendo a coerência definida pelo comparativo da nota do item de entrada e a recomendação gerada pelo algoritmo. A lógica da comparação é da seguinte maneira: se um item de entrada com nota alta, gerar uma recomendação com nota alta, significa que o resultado é considerado como coerente e é computada como um acerto. Caso o algoritmo retorne uma recomendação com nota baixa, a recomendação é considerada como incoerente e é computada com um erro. Se um item de entrada com nota baixa, gerar uma recomendação com nota baixa, a recomendação é considerada como coerente e é computada como um acerto, este caso é levado em conta para contextos onde o sistema possa vetar recomendações para o usuário. E caso o contrário, gerar uma recomendação com nota alta, a recomendação é considerada como incoerente e é computada com um erro, ou seja, basicamente temos a operação de um *and* lógico nesta verificação.

Por fim, dadas as classificações de erro/incoerência e acerto/coerência, são feitos os cálculos de acurácia, precisão, *recall* e *f1-score*, a fim de avaliar as previsões do algoritmo de recomendação. Para obter resultados mais precisos na métrica, de forma quantitavia, faz-se necessário utilizar uma base de dados com diversos usuários e suas avaliações/notas.

3.2. Medida LRIM - *List of Recommended Items Metric*

A medida LRIM avalia algoritmos de recomendação que a partir de um item geram mais de uma recomendação, formando uma lista de recomendação. Para uma mesma recomendação (lista) se tem diversas notas, e por este motivo a qualidade de cada lista recomendada é aferida pela mediana das notas.

O processo dessa medida é especificado pelo Algoritmo 2, o qual recebe como entrada uma base de dados com avaliações de usuários dos respectivos itens ($listaDeUsuários$). O resultado é dado por uma lista de medianas gerais.

Algoritmo 2 LRIM

```
1: for each usuário ∈ usuários do
2:     for each item ∈ usuário do
3:         itemFiltradoComNotaAlta ← filtragemDoItem(item)
4:         listaRecomendação ← algoritmoDeRecomendação(itemFiltradoComNotaAlta)
5:         if notas ← listaRecomendação.notas ⋂ usuário.itens then
6:             medianasDasListas[lista].append(mediana(notas))
7:         end if
8:     end for
9: end for
10: for each medianasDeÚnicaLista ∈ medianasDasListas.unique() do
11:     medianasGeraisDasListas.append(mediana(medianasDeÚnicaLista.medianas))
12: end for
13: return medianasGeraisDasListas
```

O cálculo da medida LRIM é iniciado pela inserção de cada item da base de dados no algoritmo de recomendação. Como na medida SRIM, cada item de cada usuário é filtrado baseado por sua nota, mas neste caso são considerados os itens com nota alta.

O item filtrado é colocado como entrada no algoritmo de recomendação, que por sua vez gera como saída uma lista de recomendação, contendo itens recomendados (linha 4). Em seguida é verificado se o usuário em questão avaliou ao menos um item da lista de recomendação (linha 5). Itens da lista de recomendação não avaliados pelo usuário são desconsiderados.

O resultado é dado em duas etapas, a primeira etapa representada pela linha 6, onde para cada lista de recomendação avaliada pelo usuário é calculada a mediana de suas notas. Neste caso pode-se ter diferentes listas de recomendação com um mesmo item que as gerou, a depender dos itens avaliados pelo usuário. Sendo assim, usuários diferentes podem ter listas de recomendação semelhantes ou iguais, a partir de um mesmo item de entrada.

Como um mesmo item gera diferentes listas de recomendação a depender de cada usuário, temos diversas medianas para uma mesma lista de recomendação gerada por um mesmo item. Para uma única lista de recomendação ter apenas uma mediana representando o seu resultado, na segunda etapa (linhas 10 e 11), é feita a mediana geral única de cada lista, utilizando suas medianas.

4. Simulação Computacional e Análise dos Resultados

A fim de validar as medidas propostas (SRIM e LRIM), realizamos uma avaliação experimental neste caso utilizando os algoritmos de recomendação por filtragem baseada em

conteúdo propostos em [Ahmed 2020]. Estes algoritmos foram aplicados à base de dados sobre filmes da TMDB Movies [TMDb 2017].

Em [Ahmed 2020] foram propostas duas versões do mesmo tipo de recomendação, onde a segunda versão considera mais características de conteúdo dos filmes, como relatado pela autora do trabalho, a segunda versão proporciona recomendações mais precisas do que a primeira versão do algoritmo. Para a avaliação experimental foi denominado a primeira versão como Algoritmo de Recomendação 1 (AR1) e a segunda versão como Algoritmo de Recomendação 2 (AR2).

A autora de [Ahmed 2020] provê uma avaliação qualitativa dos algoritmos propostos. A primeira versão de seu algoritmo gera boas recomendações no caso de filmes com descrições de enredo semelhantes. Para elucidar esse tipo de avaliação, foi utilizado como filme de entrada para o algoritmo, o filme "Batman - The Dark Knight Rises" dirigido por Christopher Nolan. O algoritmo gera como recomendações todos os filmes do Batman de diferentes diretores, e por consequência possuem estilos diversos de filmes que carregam um título semelhante. Neste caso, uma recomendação mais provável para um fã do filme de entrada, seja outro filme de Christopher Nolan com forte caracterização do diretor, além de considerar outros elementos do filme como principais atores e atrizes. Com isso, a autora fez uma segunda versão do algoritmo de recomendação levando em conta o acréscimo de filtros para seu algoritmo baseado em conteúdo, com o intuito de melhorar as recomendações em relação à primeira versão do algoritmo. Seu teste neste caso mostrou uma melhora considerável.

Neste trabalho, foram utilizadas as duas versões deste algoritmo, a fim de possibilitar a comparação entre os resultados das métricas nas propostas, uma vez que como referenciado pela a autora do algoritmo de recomendação, também esperamos um melhor resultado pela segunda versão. Para a avaliação experimental foi utilizada aplicação web Jupyter Notebook, e para desenvolver as medidas foram usadas as seguintes bibliotecas do Python: *pandas*, *numpy*, *statistics*, *seaborn*, *sklearn metrics* e *matplotlib*. A base de dados contendo as avaliações dos usuários foi a MovieLens do GroupLens [Harper and Konstan 2006], publicada em 2019 possuindo 25 milhões de avaliações distribuídas entre 162.000 usuários para 62.000 filmes. Os atributos da base foram "id do usuário", "título do filme" e "nota do usuário para o filme", sendo este último variando de 0 à 5 com passo de 0,5.

Devido uma grande quantidade de usuários com poucas avaliações aos filmes, podendo ser usuários esporádicos, conforme indicado na Figura 2, foi realizado um corte na base de dados, passando a considerar apenas usuários com 500 ou mais avaliações, a fim de manter apenas usuários mais relevantes. Após o corte, temos 9.712 usuários e 8.854.363 avaliações. Em seguida aplicamos o SRIM e LRIM para avaliar as recomendações fornecidas pelos algoritmos AR1 e AR2.

4.1. SRIM

Para a métrica de única recomendação os AR1 e AR2 foram modificados para retornarem somente um filme como recomendação. Além disso, deve-se definir as notas de corte para o intervalo que será considerado como nota baixa e para o intervalo que será considerado como nota alta baseado no contexto das notas da base de dados. No caso, como as notas vão de 0 a 5, foi considerado como nota baixa os valores entre 0 e 2, e como nota alta

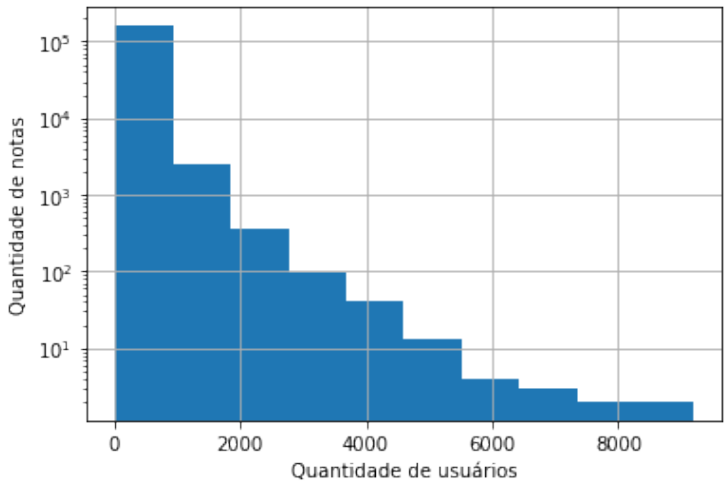


Figura 2. Gráfico de frequência acumulada.

os valores entre 4 e 5. Valores do intervalo maior que 2 e menor que 4 são considerados neutros para o gosto do usuário,ou seja, não é considerado que o usuário tenha gostado, tampouco que não tenha gostado.

Para a matriz de confusão do AR1 (Figura 3) e para a matriz de confusão do AR2 (Figura 4) o índice 0 representa filmes recomendados com notas baixas e o índice 1 representa filmes recomendados com notas altas. O *verdadeiro positivo* e o *verdadeiro negativo* podem ser considerados acertos dos algoritmos de recomendação. Em determinados contextos o verdadeiro negativo pode ser bem aproveitado para agregar em um sistema de recomendação mais robusto, como em um cenário onde um sistema tenha a necessidade de vetar recomendações para um usuário, levando em conta suas avaliações para os itens do sistema. Por exemplo, em uma campanha publicitária na internet de uma loja virtual de material esportivo, deve-se evitar recomendar para seu cliente material esportivo do clube rival de seu cliente, caso ocorra, pode afastá-lo de clicar na publicidade da loja. Tanto o AR1 quanto o AR2 possuem valores relativamente próximos a *verdadeiro negativo*, *falso positivo* e *verdadeiro positivo*. Em ambos os algoritmos o *verdadeiro positivo* se sobressai aos demais, ou seja, apresentam acertos quando o quesito são recomendações de notas altas tendo avaliação por parte dos usuários com notas altas. O AR1 se destaca em relação ao *verdadeiro negativo*, quando as recomendações de notas baixas tem avaliação por parte dos usuários com notas baixas.

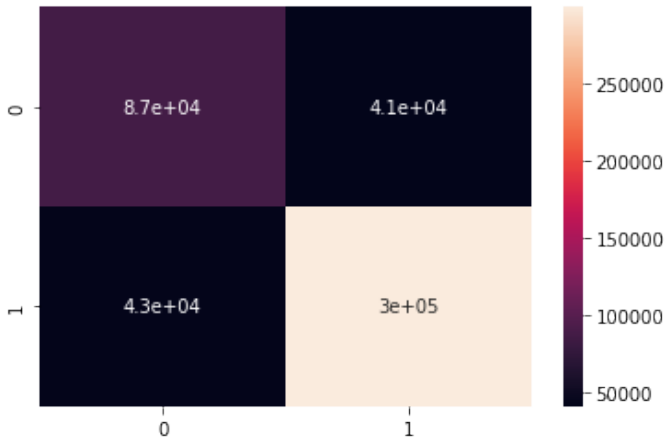


Figura 3. Matriz de confusão AR1.

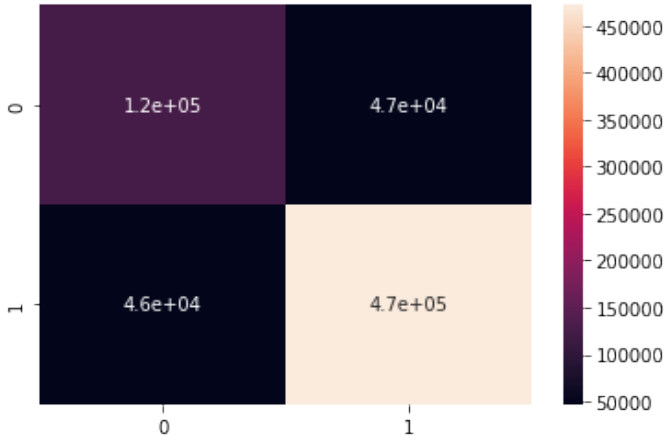


Figura 4. Matriz de confusão AR2.

Para o AR1, foi retornado 515.411 comparações com notas altas e 209.337 comparações com notas baixas. Ao passo que, para o AR2, temos 766.170 comparações com notas altas e 272.612 comparações com notas baixas. A Tabela 1 apresenta o resultado das métricas clássicas para os AR1 e AR2.

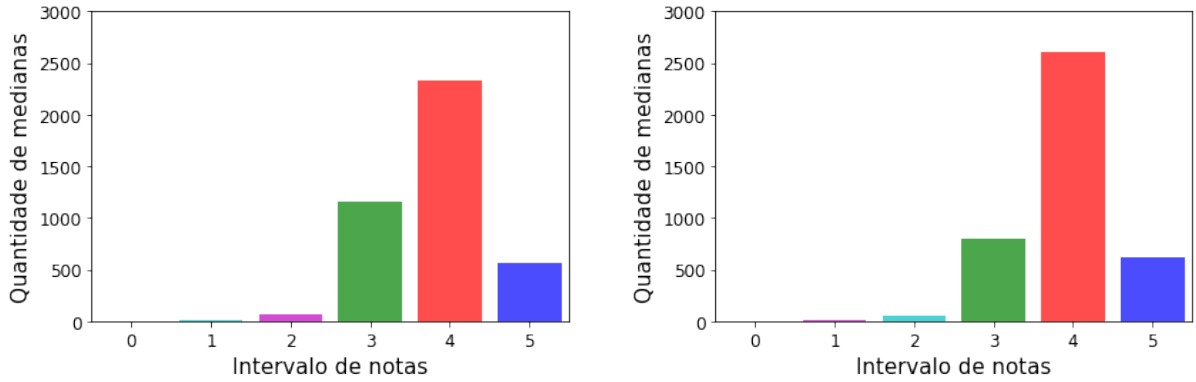
Tabela 1. Resultados gerais - SRIM

	Acurácia	**Precisão**	**Recall**	**F1-score**
AR1	0,82	0,88	0,88	0,88
AR2	0,86	0,91	0,91	0,91

Nota-se que o AR2 apresentou recomendações mais eficazes do que o AR1, tendo em vista a superioridade obtida em todas as métricas.

4.2. LRIM

Dado a mediana de cada lista de recomendação, é feita a mediana geral com as medianas de todas as listas geradas por um único filme de entrada ao algoritmo. O quantitativo de medianas gerais para o AR1 e AR2 é apresentado na Figura 5.



(a) AR1 (b) AR2

Figura 5. Resultados das medianas gerais das listas.

Para o AR1, foi retornado 363.021 listas de recomendação, sendo 4.120 de listas únicas. Dentre as listas únicas, 2.884 são de medianas altas, ou seja, medianas com valor maior ou igual a 4. Para o AR2, temos 387.132 listas de recomendação, sendo 4.086 de listas únicas. Dentre as listas únicas, 3.219 são de medianas altas.

Para auxiliar na determinação do algoritmo que tem a melhor performance é realizada uma análise da variância dos intervalos de quantidade das listas únicas representada pela Tabela 2, comparando as variâncias dos algoritmos.

Tabela 2. Variância dos intervalos de quantidade das listas únicas

Intervalos	**[0,1]**	**(1,2]**	**(2,3]**	**(3,4]**	**(4,5]**
AR1	0,069	0,068	0,037	0,058	0,091
AR2	0,038	0,048	0,049	0,054	0,085

Nota-se que o AR2 demonstrou ter recomendações mais eficazes do que o AR1, pois as listas únicas demonstraram ter proporcionalmente uma maior quantidade de medianas altas, no caso, um aumento de 70% do AR1 para 78,8% do AR2 entre as duas versões. E também como esperado, o AR2 demonstrou uma variação menor em quase todos os intervalos, exceto no intervalo maior que 2 e menor igual a 3, onde o AR1 demonstrou uma variação menor.

5. Conclusões e trabalhos futuros

O aumento considerável de dados gerados diariamente na Internet, tanto em plataformas *streaming* de vídeos quanto em compras *online* e redes sociais, fomentaram o desenvolvimento de algoritmos de recomendação. De forma geral, os algoritmos de recomendação buscam automatizar o processo de sugestão de um item de forma a melhorar a experiência do usuário. Avaliar as recomendações retornadas pelos algoritmos de forma quantitativa ainda é um grande desafio.

Neste contexto, este trabalho apresentou duas medidas capazes de avaliar recomendações. A primeira, denominada SRIM, teve como objetivo quantificar a qualidade da recomendação de um item quando é retornado uma única sugestão. A segunda medida

proposta, denominada LRIM, buscou avaliar a qualidade quando uma lista de sugestões era feita ao usuário.

A fim de verificar a assertividade das medidas propostas, foi realizada uma avaliação experimental avaliando utilizando dois algoritmos de recomendação de diferentes complexidades foi realizada. Esta avaliação experimental envolveu o cenário de recomendação de filmes. Como resultado, pode-se observar que as medidas propostas conseguiram identificar de forma correta o melhor dos algoritmos, tanto para o caso de sugestão única quanto para a lista de sugestões.

A partir dos itens recomendados por um algoritmo de recomendação e uma coleção de dados, onde esses itens possuem avaliações positivas e negativas, de diferentes usuários em grande quantidade, é viável aplicar as medidas SRIM e LRIM em outros contextos. Como trabalho futuro pretendemos utilizar outras bases de dados, como exemplo [Spachtholz 2017] que pode ser usada para avaliar as recomendações de um algoritmo que recomenda livros ou no caso dos diversos produtos vendidos pela Amazon podemos usar a base de dados divulgada por [Ni et al. 2019] armazenada em [Ni 2018] tendo avaliação de usuários em diversos contextos.

Referências

Ahmed, I. (2020). Getting started with a movie recommendation system. https://www.kaggle.com/code/ibtesama/getting-started-with-a-movie-recommendation-system/notebook. Acessado em: 29 de Março de 2022.

da Silva Costa, F. (2017). *Um estudo comparativo da acurácia de algoritmos de recomendação em sistemas de compras coletivas*. PhD thesis, Universidade Federal de Campina Grande, Campina Grande, PB.

de Freitas Júnior, H. B., Pereira, F. D., de Oliveira, E. H. T., de Oliveira, D. B. F., and de Carvalho, L. S. G. (2020). Recomendação automática de problemas em juízes online usando processamento de linguagem natural e análise dirigida aos dados. In *Anais do XXXI simpósio brasileiro de informática na educação*, pages 1152–1161. SBC.

de Oliveira Yamaguti, R. and Coello, J. M. A. (2013). Algoritmo híbrido para sistemas de recomendação utilizando filtragem colaborativa e algoritmo genético. In *Anais do XVIII Encontro de Iniciação Científica – ISSN 1982-0178*, pages 1–5. PUC-Minas.

Harper, M. and Konstan, J. (2006). Movielens data set. *ACM Trans Interact Intell Syst, 2016, 5: 1*, 19.

Himel, M. T., Uddin, M. N., Hossain, M. A., and Jang, Y. M. (2017). Weight based movie recommendation system using k-means algorithm. In *2017 International Conference on Information and Communication Technology Convergence (ICTC)*, pages 1302–1306. IEEE.

Lorenção, H. S. and Santos, R. V. (2021). Elaboração de um serviço de recomendação híbrido ponderado e misto implantado em webservice restful. In *Anais do VIII Encontro Nacional de Computação dos Institutos Federais*, pages 49–56. SBC.

Marinho, L. H., Campos, R., dos Santos, R. P., da Silva, M. F., and Oliveira, J. (2019). Conceitos, implementação e dados privados de algoritmos de recomendação. *Sociedade Brasileira de Computação*.

Ni, J. (2018). Amazon review dataset. https://nijianmo.github.io/amazon/index.html. Acessado em: 18 de Abril de 2022.

Ni, J., Li, J., and McAuley, J. (2019). Justifying recommendations using distantly-labeled reviews and fine-grained aspects. In *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing (EMNLP-IJCNLP)*, pages 188–197.

Oliveira, S. E., Diniz, V., Lacerda, A., Merschmanm, L., and Pappa, G. L. (2020). Is rank aggregation effective in recommender systems? an experimental analysis. *ACM Transactions on Intelligent Systems and Technology (TIST)*, 11(2):1–26.

Ramos, J. G. A. (2010). Algoritmos colaborativos para sistemas de recomendação. Master's thesis, Faculdade de Engenharia da Universidade do Porto, Porto, PT.

Silva, R. G. N. (2017). Sistema de recomendação baseado em conteúdo textual: avaliação e comparação. Master's thesis, Universidade Federal de Bahia, BA.

Spachtholz, P. (2017). Book recommender: Collaborative filtering, shiny. https://www.kaggle.com/code/philippsp/book-recommender-collaborative-filtering-shiny/data. Acessado em: 18 de Abril de 2022.

Takahashi, M. M. and Hirata Jr, R. (2015). Estudo comparativo de algoritmos de recomendação. Monografia (Bacharelado em Ciências da Computação), Instituto de Matemática e Estatística (IME), Universidade de São Paulo, SP.

TMDb (2017). Tmdb 5000 movie dataset. https://www.kaggle.com/datasets/tmdb/tmdb-movie-metadata. Acessado em: 29 de Março de 2022.