Universidade Federal do Paraná
Setor de Ciências Exatas
Departamento de Estatística
Programa de Especialização em *Data Science* e *Big Data*

Hugo Hiroshi Yamamura

Avaliação de sistemas de recomendação de conteúdo: uma abordagem offline

Curitiba
2024

Hugo Hiroshi Yamamura

Avaliação de sistemas de recomendação de conteúdo: uma abordagem offline

Monografia apresentada ao Programa de Especialização em *Data Science* e *Big Data* da Universidade Federal do Paraná como requisito parcial para a obtenção do grau de especialista.

Orientador: Prof. Marco Antônio Zanata Alves

Curitiba
2024

Avaliação de sistemas de recomendação de conteúdo: uma abordagem offline

Evaluation of Content Recommendation Systems: An Offline Approach

Hugo Hiroshi Yamamura[1], Marco Antonio Zanata Alves[2]

[1]Aluno do programa de Especialização em Data Science & Big Data, hugo.yamamura@ufpr.br
[2]Professor do Departamento de Informática - DINF/UFPR, mazalves@dinf.ufpr.br

Nesta monografia, exploramos o desenvolvimento e a avaliação de sistemas de recomendação de conteúdo usando o conjunto de dados *MovieLens 25M*. Duas abordagens principais são consideradas: filtragem baseada em conteúdo e filtragem colaborativa. O método de filtragem baseada em conteúdo utiliza distância por cosseno a partir dos gêneros dos filmes para fazer recomendações com base nos filmes já assistidos pelos usuários. Por outro lado, a filtragem colaborativa aproveita as avaliações de outros usuários em outras obras para identificar usuários semelhantes e recomendar filmes ainda não assistidos ou desconhecidos. Por fim, serão discutidas a eficácia e as limitações dos métodos, proporcionando estratégias para suas aplicações práticas e desempenho em cenários do mundo real.
Palavras-chave: Sistemas de recomendação, filtragem baseada em conteúdo, filtragem colaborativa, sistemas de recomendação de conteúdo

In this essay, we explore the development and evaluation of content recommendation systems using the *MovieLens 25M* dataset. Two main approaches are considered: content-based filtering and collaborative filtering. The content-based filtering method uses cosine similarity from movie genres to recommend films based on those already watched by users. On the other hand, collaborative filtering leverages ratings from other users on different works to identify similar users and recommend movies that have not yet been watched or are unknown. Finally, the effectiveness and limitations of the methods will be discussed, providing strategies for their practical applications and performance in real-world scenarios.
Keywords: Recommendation systems, content-based filtering, collaborative filtering, content recommendation systems

1. Introdução

Bilhões de pessoas criam, editam, assistem e compartilham vídeos no *YouTube*, assistem a séries e filmes em plataformas de *streaming* como *Netflix*, *Amazon Prime* ou *Disney+* e tantos outros curtem, comentam, enviam sua opinião ou dão uma nota ao conteúdo acessado. Ao analisar esses dados, os sistemas de recomendação podem prever as preferências dos usuários e recomendar conteúdos que provavelmente eles irão gostar. As empresas estão usando as redes sociais para expandir seus negócios, e a mineração de dados e análise são muito importantes hoje em dia. Para ajudar os usuários a descobrirem novas obras relevantes que se encaixem em suas preferências, é necessário o desenvolvimento e aprimoramento de sistemas de recomendação sofisticados buscando tanto uma boa acurácia como um uso equilibrado de recursos, como tempo, processamento e consumo de memória [1]. Entre as várias abordagens, a recomendação baseada em conteúdo (*Content-based Recommendation*) e a filtragem colaborativa (*Collaborative Filtering*) emergiram como duas das técnicas amplamente utilizadas, como demonstrado em [2], [3] e [4]. Neste trabalho, são examinados esses métodos no contexto do conjunto de dados *MovieLens 25M*, uma coleção em larga escala de avaliações de usuários e metadados de filmes. Nosso objetivo é comparar o desempenho das abordagens de filtragem baseada em conteúdo e filtragem colaborativa, destacando seus pontos fortes e fracos durante o processo.

Este artigo está dividido nas seguintes seções: a Seção 2, onde se descreve a base de dados utilizada e o tratamento realizado sobre ela, o funcionamento dos

métodos de decomposição em valores singulares (Seção 3) e de distância por cosseno (Seção 4), e a forma de avaliar o desempenho dessas duas abordagens. As seções 5 e 6 descrevem, respectivamente, os resultados obtidos e apresentam os aspectos positivos e negativos baseados nos resultados de cada método.

2. Materiais e métodos

O conjunto de dados usado foi o *MovieLens 25M* [5] . Lançado pelo laboratório de pesquisa *GroupLens*, ele é uma coleção abrangente de avaliações de filmes e metadados.

Ele contém 15 milhões de avaliações e 352 mil *tags* em 62.000 filmes, fornecidas por mais de 162 mil usuários. O conjunto de dados inclui gêneros dos filmes, datas de avaliação e *tags* geradas pelos usuários.

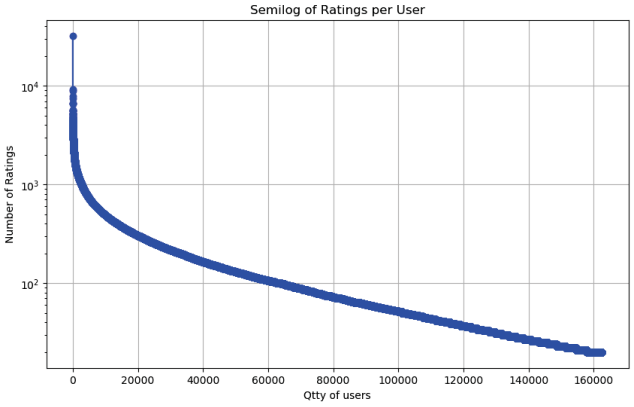

Figura 1: Distribuição de notas por usuário.

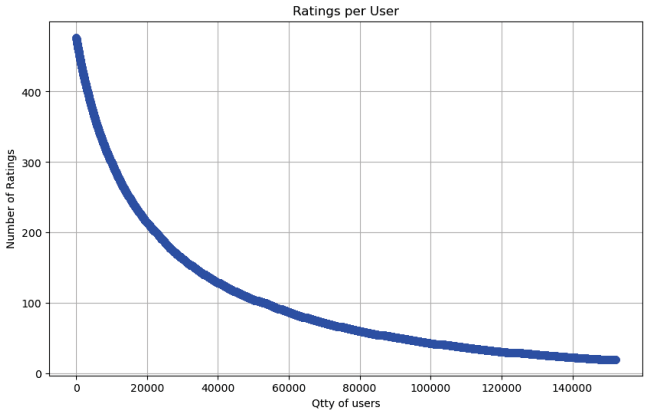

Figura 2: Distribuição de notas por usuário após Tukey's Fences de $k = 2,5$.

Pela Figura 1, observamos que poucos usuários opinam sobre mais de 1000 filmes. Na literatura, existem diversas estratégias para lidar com *outliers*. Uma que não depende da natureza da distribuição dos dados é o *Tukey's Fences* [6], que identifica os outliers a partir de uma janela baseada numa proporção k da *Inter Quartile Range (IQR)*. Dessa forma, foi adotado um corte dos *outliers* que não estejam dentro de $k = 2,5$. Logo, apenas usuários com até 477 filmes opinados foram considerados no estudo (Figura 2)

Após a conclusão da separação dos usuários entre treinamento e testes em uma proporção de 85:15, sobre o *dataset* de treinamento foi escolhido aplicar o modelo de decomposição em valores singulares ou *singular value decomposition (SVD)* para o *Collaborative Filtering*, cujos fundamentos serão desenvolvidos na seção 3.

As Figuras 3 e 4 mostram a distribuição de notas em cada *subset*.

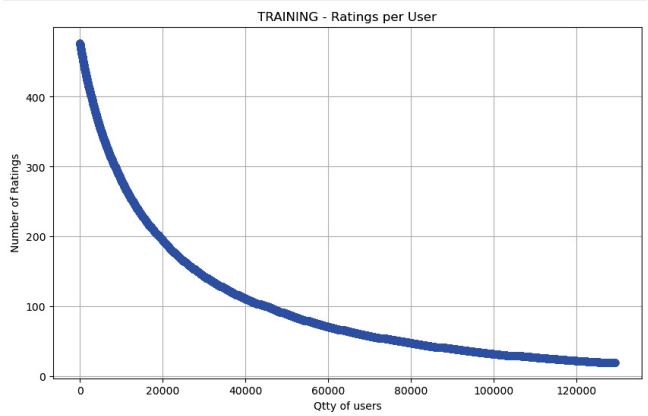

Figura 3: *Subset* de treinamento

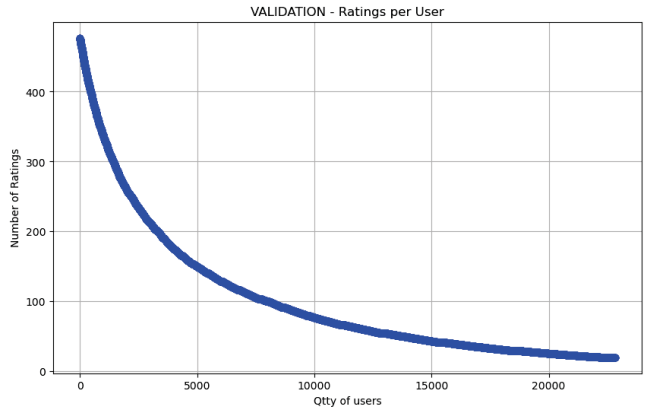

Figura 4: *Subset* de validação

Sobre os dados para treinamento, adotou-se a abordagem de *K-fold Cross Validation*, que divide o *dataset* em k *subsets* a fim de selecionar um deles para validação e o restante $(k-1)$ para o treinamento, revezando

a cada iteração entre todos os *subsets* para a validação. Escolheu-se $k = 5$.

3. *Singular Value Decomposition (SVD)*

A SVD decompõe uma matriz A_{mxn} em três outras matrizes conforme a Fórmula 1:

$$A_{mxn} = U_{mxm}\Sigma_{mxn}V_{nx}{}_n^T \quad (1)$$

- U é a matriz ortogonal dos vetores singulares à esquerda (representa usuários)
- Σ é uma matriz diagonal com valores singulares (representa a importância dos fatores latentes)
- V^T é a matriz ortogonal dos vetores singulares à direita (representa itens)

Por ser uma técnica de álgebra linear, ela tem diversas aplicações práticas por conta da sua propriedade de reduzir a dimensionalidade de um modelo, desde redução de ruído em processamento de sinais, compressão de imagem, e no caso de recomendação de conteúdo, determinar os fatores latentes de maior influência nas preferências do usuário.

Foi criada uma matriz esparsa A para obter a relação de títulos e as notas dadas pelos usuários, preenchendo os filmes sem avaliação com zeros e aplicando a SVD.

4. Distância por cosseno

Já para o contexto de **Filtragem Baseada em Conteúdo**, escolheu-se a distância por cosseno 4 para comparar a semelhança entre filmes. Por trabalhar com a similaridade por gêneros dos filmes, a distribuição das notas pelos usuários no *dataset* não afetará esse modelo.

Para tal, é criado um vetor binário com tamanho igual aos gêneros distintos existentes no *dataset* e o valor é 1 quando o filme está dentro desse gênero e 0 caso contrário.

A similaridade por cosseno entre dois vetores pode ser obtida pela fórmula 3 a seguir

$$\mathbf{A}\cdot\mathbf{B} = \|\mathbf{A}\|\|\mathbf{B}\|\cos(\theta)$$
$$\text{similaridade cosseno} = \frac{\mathbf{A}\cdot\mathbf{B}}{\|\mathbf{A}\|\|\mathbf{B}\|} \quad (3)$$

- A e B são vetores de gêneros de dois filmes
- $(A \cdot B)$ é o produto escalar entre A e B dado por $a_1b_1 + a_2b_2 + \cdots + a_nb_n = \sum_{i=1}^{n} a_ib_i$
- $||A||$ significa a magnitude ou módulo de um vetor A

Segundo a biblioteca *scipy* [7], a fórmula da distância é dada pela Fórmula 4:

$$\text{distância cosseno} = 1 - \text{similaridade cosseno} \quad (4)$$

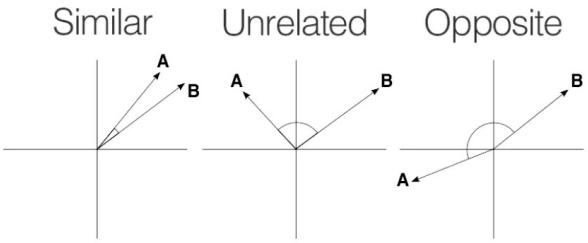

Figura 5: Representação da similaridade de dois vetores [8]

Conforme representado na Figura 5, quanto menor o ângulo θ, mais similares serão os vetores. Em outras palavras, quanto mais próximo de 0 for a distância por cosseno, mais semelhantes serão, enquanto que se o valor estiver próximo de 1 são não relacionados.

Para esta abordagem, serão quantificados quais filmes da lista de recomendações o usuário já assistiu e quais foram as notas atribuídas por ele.

Os arquivos para reproduzir a mesma configuração do ambiente e o código das duas abordagens pode ser encontrado dentro do repositório *GitHub* `https://github.com/YYHugo/contentRecommendation`.

O código foi executado em um servidor rodando *Ubuntu 23.10 (Mantic)* com *kernel 6.5.0-41-generic*, equipado com processador *Intel(R) Xeon(R) Silver 4114 CPU @ 2.20GHz* e 4 pentes de 32 GiB DIMM DDR4 em frequência 2666 MHz, totalizando 128 Gib. As bibliotecas para o carregamento e tratamento dos dados foram:

- Python (3.11.7)
- pandas (2.1.4)
- numpy (1.26.4)
- matplotlib (3.8.0)
- seaborn (0.12.2)
- scikit-learn (1.2.2)

5. Resultados

Para a abordagem de Filtragem Colaborativa adaptada a sugerir filmes, dos 2000 filmes sugeridos, 239

filmes (11,95%) foram realmente assistidos pelos usuários, porém a taxa de acertos não era homogênea para os usuários, conforme amostragem de usuários da Tabela 1:

Id do usuário	Proporção (%)
95097	8,1632
98058	17,1875
39268	12,1621
38344	13,6363
108710	1,5873
14000	4,6875
55214	17,0731
33933	8,3333
42501	6,2827
44920	25,6410
4966	17,5000
79304	13,7931
109187	12,9032
105336	4,7169
160117	8,0808
55309	0,9389
144854	13,2352
115272	2,5000
110187	8,9041
53677	36,8421

Tabela 1: Tabela de proporção de filmes assistidos e que foram mencionados nas sugestões de similaridade por SVD

No modelo que se baseia em similaridade de cossenos, de um total de 2000 filmes, apenas 3 (0,15%) estavam na lista de assistidos pelos usuários.

Embora existam filmes com exatamente o mesmo gênero, haverá filmes recomendados cuja nota dada pelo usuário é baixa. Por exemplo, dentre a lista de 100 filmes sugeridos por serem semelhantes ao filme #1378, o filme #2701 e o filme #2951 foram avaliados pelo usuário #112604, que graduou nota 4,0 para o último, mas nota 1,0 para o primeiro. Escolhendo aleatoriamente alguns dos usuários e calculando a proporção entre os filmes que o usuário assistiu e as obras sugeridas, observa-se uma fração muito pequena dos filmes que os usuários separados para a validação assistiriam, como pode ser visto na Tabela 2.

Id do usuário	Proporção (%)
1602	0,0000
18805	0,0000
42695	0,0000
21272	0,0000
32560	0,0000
2372	0,0000
109150	0,0000
83397	0,0000
33939	0,0000
44699	0,0000
10679	0,0000
50775	0,0000
90910	0,0000
7012	0,0000
43191	0,0000
40396	0,0000
135269	0,0000
96571	6,2500
143281	1,6129
122308	0,0000

Tabela 2: Tabela de proporção de filmes assistidos e que foram mencionados nas sugestões de similaridade por cosseno

Em contrapartida, percebe-se que a quantidade de memória utilizada na execução do método SVD foi de 70 GB ante os 11 GB consumidos para representar todos os vetores de gênero da lista de filmes.

6. Conclusões

A modelagem por SVD, por ponderar sobre a opinião dos usuários baseado em suas notas, teve mais filmes sugeridos que um usuário do *subset* de validação assistiria. A modelagem por cossenos se limitou à proximidade dos filmes quanto a suas categorias de gênero, deixando de lado uma possível ponderação de gostos dos usuários.

Em termos de consumo de recursos, a principal desvantagem foi o alto consumo de memória RAM para a representação do modelo SVD. Apesar de este estudo não considerar o tempo de execução de cada algoritmo, para modelos que ocupam mais memória, espera-se que o tempo de execução também seja maior. Portanto, o crescimento dessa base de dados pode tornar o método SVD não escalável com o aumento de usuários ou da coleção de filmes e por isso, em situações onde é necessário uma resposta rápida, pode ser necessário

adaptar o método, mesclando com outros algoritmos que tratem essa lentidão.

Nota-se que ambas as abordagens consideraram um perfil estático. Entretanto, o perfil dos usuários podem mudar ao longo do tempo. Uma mesma pessoa pode mudar de gostos e de interesse e por conta disso, os filmes e consequentemente a opinião sobre uma mesma obra poderão mudar ao longo do tempo. Assim, sugere-se incluir uma análise dentro de uma janela de tempo a fim de considerar a temporalidade das preferências dos usuários.

Adicionalmente, em ambos enfrenta-se o problema de *"cold start"*, que refere-se a um problema em que um sistema ou parte dele necessita de um valor prévio quando iniciar sua execução. Para a similaridade cosseno, o usuário precisa ter escolhido ao menos um filme para que o sistema possa sugerir outros baseados em seu gênero, a exemplo que a *Netflix* realiza quando solicita aos usuários escolherem algumas opções logo no primeiro acesso à plataforma. Para a sugestão baseada nas notas de outros usuários, são necessárias as notas de outros usuários sobre o mesmo filme e sobre outros filmes.

Referências

[1] Vinagre, J., Jorge, A. M., Gama, J. (2014). "Evaluation of recommender systems in streaming environments."https://www.researchgate.net/publication/265913731_Evaluation_of_recommender_systems_in_streaming_environments

[2] G. Sunandana, M. Reshma, Y. Pratyusha, M. Kommineni and S. Gogulamudi, "Movie recommendation system using enhanced content-based filtering algorithm based on user demographic data,"2021 6th International Conference on Communication and Electronics Systems (ICCES), Coimbatre, India, 2021, pp. 1-5, doi: 10.1109/ICCES51350.2021.9489125 https://www.overleaf.com/project/661400ca98316c9cc9aeb0d2

[3] Y. Chen, "A music recommendation system based on collaborative filtering and SVD,"2022 IEEE Conference on Telecommunications, Optics and Computer Science (TOCS), Dalian, China, 2022, pp. 1510-1513, doi: 10.1109/TOCS56154.2022.10016210 https://ieeexplore.ieee.org/document/10016210

[4] V. Paranjape, N. Nihalani and N. Mishra, "A Machine Learning Approach for Item Recommendation Using SVD Technique,"2021 IEEE 4th International Conference on Computing, Power and Communication Technologies (GUCON), Kuala Lumpur, Malaysia, 2021, pp. 1-7, doi: 10.1109/GUCON50781.2021.9574001 https://ieeexplore.ieee.org/document/9574001

[5] Harper, F. M., Konstan, J. A. (2015). "The MovieLens Datasets: History and Context". ACM Transactions on Interactive Intelligent Systems. https://doi.org/10.1145/2827872 Acessado em: 09/06/2024.

[6] Diachkov, D. "Outlier detection in R: Tukey Method or why you need “box and whiskers” (01/2024). Disponível em: https://medium.com/data-and-beyond/outlier-detection-in-r-tukey-method-or-why-you-need-box-and-whiskers-3c35d9ad8fb3 Acessado em: 06/08/2024.

[7] SciPy Project. "scipy.spatial.distance.cosine"s.d. Disponível em: https://docs.scipy.org/doc/scipy/reference/generated/scipy.spatial.distance.cosine.html Acessado em: 11/08/2024

[8] Shkhanukova, M.. "Cosine distance and cosine similarity"03/2023. Disponível em: https://medium.com/data-and-beyond/outlier-detection-in-r-tukey-method-or-why-you-need-box-and-whiskers-3c35d9ad8fb3 Acessado em: 06/08/2024