Universidade de Brasília
Departamento de Estatística

Algoritmos de Recomendação: Estudo aplicado a streaming de anime

Larissa Moreno Silva

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2021

Larissa Moreno Silva

Algoritmos de Recomendação: Estudo aplicado a streaming de anime

Orientador(a): Thais Carvalho Valadares Rodriguesr

Projeto apresentado para o Departamento de Estatística da Universidade de Brasília como parte dos requisitos necessários para obtenção do grau de Bacharel em Estatística.

Brasília
2021

Dedico esse trabalho a todos que me apoiaram durante a realização desse trabalho.

Resumo

Sistemas de recomendações são algoritmos que funcionam como filtros que identificam o que é e o que não é relevante para determinado usuário. Existem vários tipos de sistemas de recomendações, este trabalho de graduação implementa as técnicas de filtragem baseada em conteúdo, filtragem colaborativa e filtragem por popularidade para gerar recomendações de animes aos usuários utilizando informações sobre os animes como gênero e avaliações dos usuários. O método de classificação K-Nearest Neighbors (KNN) foi utilizado na aplicação da filtragem colaborativa e a técnica de agrupamento foi utilizada para a filtragem baseada em conteúdo. Além disso, este trabalho utilizou de diferentes métricas de avaliação para medir o desempenho de cada filtragem.

Palavras-chave: Sistemas de Recomendação; filtragem baseada em conteúdo; filtragem colaborativa; filtragem por popularidade; KNN; K-means; animes.

Lista de Figuras

Lista de Tabelas

Sumário

1 Introdução

O grande avanço da tecnologia na área de comunicação proporcionou um aumento do volume de informação disponível. Saber analisar esses dados é essencial para qualquer negócio se manter em um mercado competitivo, porém, processar essa enorme quantidade de informação não é fácil, e em muitas situações quase impossível. De acordo com a Associação Motion Picture, os serviços de streaming de vídeo ultrapassaram mais de um bilhão de assinaturas no ano de 2020. Em meio a esse crescimento, as companhias precisam melhorar a qualidade de como é fornecido seus produtos. Uma forma de personalizar seus serviços é usando os algoritmos de recomendação.

Os algoritmos de recomendações surgiram inicialmente na década de 90 e hoje são utilizados por grandes empresas, como a Netflix (GOWER, 2014), Amazon (LINDEN G., 2003), Soptify (MADATHIL, 2017), Facebook (BAATARJAV; PHITHAKKITNUKOON; DANTU, 2008), entre outras. Os sistemas de recomendações são bastante usados em comércio virtual e suas principais funções são indicar e auxiliar o usuário a encontrar conteúdos que podem ser de seu interesse e reduzir seu tempo de pesquisa.

Identificar o que é e o que não é relevante sob algum critério é o principal objetivo dos sistemas de recomendação e existem critérios para determinar isso. Veremos adiante que alguns algoritmos utilizam informações baseadas nos usuários ou na concordância e similaridade entre eles e outros dados sobre o conteúdo/características de seus itens. A utilização de algumas técnicas estatísticas é essencial para ajudar a classificar e estimar produtos que usuários possam se interessar.

Neste trabalho, focaremos nos serviços de streaming de animes. A quantidade de animes produzidos é aproximadamente de 150 a 200 por ano. Diante disso, procurar algo que seja de interesse de algum usuário em meio a grandes possibilidades de opções é uma tarefa difícil. Sendo assim, torna-se interessante criar algoritmos que recomendem programas que sejam mais interessantes ou que sejam mais preferíveis para cada usuário. Esse trabalho visa otimizar esse problema do tempo de procura por esses programas.

Muitos tipos de pesquisa sobre recomendações para streaming de anime foram conduzidos usando métodos de filtragem colaborativos e filtragem baseados em conteúdo. Por exemplo, Girsang et al. (2020) utilizou uma técnica de fatoração de matrizes chamada mínimos quadrados alternados (ALS, do inglês alternating least squares) para gerar recomendações para usuários utilizando filtragem colaborativa. Freitas (2018), em seu trabalho, utilizou uma técnica que procura vizinhos próximos chamada KNN (K nearest

neighbor) para encontrar usuários com preferências similares. Nesse trabalho, no entanto, utilizará métodos baseados em vizinhanças para filtragem colaborativa baseada tanto em usuário como em item, e também de técnica de clusterização para a filtragem baseada em conteúdo.

2 Objetivos

2.1 Objetivo Geral

Aplicar diferentes técnicas para construir algoritmos de recomendação para animes a partir de características dos animes e dos usuários que assistiram tais programas.

2.2 Objetivos Específicos

- Criar uma função que imprima uma lista de recomendações para cada algoritmo criado.
- Comparar, testar e validar os algoritmos criados.
- Aprender sobre os algoritmos de recomendações.
- Criar um aplicativo de web interativo.

3 Referencial Teórico

3.1 Medidas de Similaridade

Os sistemas de recomendações usam bastante o conceito de similaridade entre itens ou usuários. A escolha de qual medida usar é muito subjetiva, geralmente leva-se em consideração a natureza das variáveis, escalas de mensuração e o conhecimento sobre o assunto. (JOHNSON; WICHERN, 2007).

3.1.1 Distância Euclidiana

A distância euclidiana pode ser utilizada para medir a similaridade. A distância euclidiana entre dois pontos x e y é definida por

$$D(X,Y) = \sqrt{\sum_{i=1}^{n}(x_{ij} - y_{kj})^2}, \tag{3.1.1}$$

onde x_{ij} é o valor da variável j para o indivíduo i e y_{kj} é o valor da variável j para o indivíduo k.

Nesse trabalho, essa medida será usada na filtragem baseada em conteúdo para calcular a similaridade dos itens a partir dos atributos gênero de cada anime utilizando uma codificação "one hot", isto é, uma representação binária de uma matriz "dummy"e, em seguida, obter os clusters dos animes por gêneros. Na seção de resultados será explicado com será utilizada essa técnica. A seguir apresenta-se um exemplo de como a distância euclidiana é calculada.

Gênero	Ação	Drama	Terror	Fantasia	Aventura
anime 1	1	0	0	1	0
anime 2	0	1	1	0	0
anime 3	1	1	0	0	1
anime 4	0	0	0	1	1

Calculando a distância do anime 1 em relação ao anime 2 e para o anime 3 e 4 tem-se,

$$D(a_1, a_2) = \sqrt{(1-0)^2 + (0-1)^2 + (0-1)^2 + (1-0)^2 + (0-0)^2} = 2.$$

$$D(a_3, a_4) = \sqrt{(1-0)^2 + (1-0)^2 + (0-0)^2 + (0-1)^2 + (1-1)^2} = 1.73.$$

Repetindo o procedimento para calcular as outras distâncias. Obtém-se a matriz das distâncias a seguir.

$$\text{Item} \begin{array}{c} \\ I_1 \\ I_2 \\ I_3 \\ I_4 \end{array} \begin{array}{c} \begin{array}{cccc} I_1 & I_2 & I_3 & I_4 \end{array} \\ \begin{pmatrix} 0 & 2 & 1.73 & 1.41 \\ 2 & 0 & 1.73 & 2 \\ 1.73 & 1.73 & 0 & 1.73 \\ 1.41 & 2 & 1.73 & 0 \end{pmatrix} \end{array}$$

3.1.2 Correlação de Pearson

O Coeficiente de Correlação de Pearson mede a correlação linear entre duas variáveis, assumindo valores entre -1 e 1. Valores diferentes de 0 indicam que existe uma correlação entre as variáveis e valores iguais à 0 indica que não há associação entre as duas variáveis. Nesse trabalho, essa medida será utilizada na filtragem colaborativa baseada em usuários para calcular a similaridade entre dois usuários distintos u e v a partir da seguinte fórmula:

$$\rho_{u,v} = \frac{\sum_{i \in I}(r_{ui} - \bar{r_u})(r_{vi} - \bar{r_v})}{\sum_{i \in I}\sqrt{(r_{ui} - \bar{r_u})^2} * \sum_{i \in I}\sqrt{(r_{vi} - \bar{r_v})^2}}, \tag{3.1.2}$$

onde I representa o conjunto de itens avaliados pelos usuários u e v, enquanto $\bar{r_u}$ representa a média das avaliações do usuário u, os valores r_{ui}, que estão no conjunto I, representa na avaliação do usuário u para o item i. Por exemplo, considere a matriz usuário-item a seguir.

$$\begin{array}{c} \\ \text{Usuário} \end{array} \begin{array}{c} \text{Item} \\ \begin{array}{cccc} I_1 & I_2 & I_3 & \end{array} \\ \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{array} \begin{pmatrix} 4 & ? & 5 & 5 \\ 4 & 2 & 1 & ? \\ 3 & ? & 2 & 4 \\ 4 & 4 & ? & ? \\ 2 & 1 & 3 & 5 \end{pmatrix} \end{array}$$

Nesse exemplo, a similaridade do usuário 1 e 5 é igual à 0.756, pois, $\bar{r_1} = 14/3$ e $\bar{r_5} = 10/3$ então,

$$\rho_{15} = \frac{(4-14/3)\cdot(2-10/3)+(5-14/3)\cdot(3-10/3)+(5-14/3)\cdot(5-10/3)}{\sqrt{(4-14/3)^2+(5-14/3)^2+(5-14/3)^2}\cdot\sqrt{(2-10/3)^2+(3-10/3)^2+(5-10/3)^2}} \tag{3.1.3}$$

$\rho_{15} = 0.756$

Repetindo o processo para obter as outras correlações, tem-se a seguinte matriz:

$$\text{Usuário} \begin{array}{c} \\ U_1 \\ U_2 \\ U_3 \\ U_4 \\ U_5 \end{array} \begin{array}{c} \begin{array}{ccccc} U_1 & U_2 & U_3 & U_4 & U_5 \end{array} \\ \begin{pmatrix} - & & & & \\ -1 & - & & & \\ 0 & 1 & - & & \\ NA & NA & NA & - & \\ 0.756 & -0.327 & 0.654 & NA & - \end{pmatrix} \end{array}$$

Nota-se que há alguns valores que são dados como NA, isso acontece quando 2 usuários possuem apenas 1 classificação em comum, sendo insuficiente para oferecer recomendações. Já no caso das correlações do usuário 4, foi obtido NA, pois o desvio padrão desse usuário é igual à zero, não sendo possível calcular a correlação de pearson. Portanto, os valores dados como NA diferem de correlações iguais a 0.

Pelos resultados, pode-se inferir que o usuário 5 são mais similares ao usuário 1 que os usuários 2 e 3, por exemplo.

3.1.3 Cosseno de Vetor

O cosseno de vetor mede a similaridade entre dois vetores com base no cosseno do ângulo formado entre eles, em uma escala de +1 e -1, onde um valor alto e positivo sugere uma alta correlação, um alto valor negativo sugere o inverso, e valores iguais a zero representam nenhuma similaridade.

$$cos(\theta) = \frac{A \cdot B}{\|A\| \cdot \|B\|}, \tag{3.1.4}$$

onde A e B são vetores das notas dos usuários, $\|A\|$ é a norma do vetor A e θ é o ângulo formado entre os dois vetores. Nesse trabalho, essa medida será usada na filtragem baseada em item para calcular a similaridade entre os itens a partir das notas dadas pelos usuários.

Por exemplo, considere a matriz abaixo,

$$
\begin{array}{cc}
 & \begin{array}{ccc} \multicolumn{3}{c}{\text{Item}} \\ I_1 & I_2 & I_3 \end{array} \\
Usuário & \begin{array}{c} U_1 \\ U_2 \\ U_3 \\ U_4 \end{array} \left(\begin{array}{ccc} 2 & ? & 3 \\ 5 & 2 & ? \\ 3 & 3 & 1 \\ ? & 2 & 2 \end{array} \right)
\end{array}
$$

A similaridade entre os itens 1 e 2 seria igual à:

$cos(\theta) = (5 \times 2 + 3 \times 3)/(\sqrt{(25+9) \times (4+9)}) = 0.903$

Note que o cálculo considera apenas os usuários 2 e 3, pois eles avaliaram os dois itens. Da mesma forma, a similaridade entre os itens 1 e 3 e os itens 2 e 3 são, respectivamente, iguais 0.789 e 0.868. Dessa forma, o item 1 é mais similar ao item 2 do que ao item 3, por exemplo.

3.2 Sistema de Recomendação

Sistemas de recomendações são algoritmos criados para resolver problemas de sobrecarga de informação, auxiliando na busca por produtos e sugerindo lista de itens que podem ser relevantes através das informações sobre o usuário e dos itens (SILVA, 2014).

Os sistemas de recomendções funcionam principalmente de duas formas. A primeira é sugerindo produtos baseados na similaridade entre usuários, por exemplo, um indivíduo gostou dos filmes A e B, e outro indivíduo gostou dos filmes A, B e C, então recomenda-se ao primeiro usuário o filme C. A segunda forma é através das características do produto, sugerir itens baseados em um item semelhante, por exemplo, se alguém pesquisar no Youtube sobre receitas de bolos, o algoritmo irá recomendar outros vídeos relacionados a confeitaria de bolos.

Existem várias técnicas de recomendações, dentre elas destacam-se: filtragem baseada no conteúdo, filtragem colaborativa e filtragem por popularidade.

3.2.1 Filtragem baseada em popularidade

Funciona basicamente sugerindo produtos que estão em alta. Essa abordagem é caracterizada pela tendência. Esse sistema generaliza as recomendações para todos os

usuários, independentemente das suas características pessoais ou preferências. Basicamente, essa estratégia visa recomendar os k itens mais populares do domínio, sobre o pressuposto de que itens que interessam a um grande número de usuários podem cobrir as distintas preferências existentes (SILVA, 2018). A vantagem dessa filtragem é que não possui problema de recomendação para usuários novos, porém não consegue sugerir itens novos.

3.2.2 Filtragem baseada no conteúdo

Essa abordagem se baseia principalmente na similaridade dos itens, ou seja, um usuário gostará de itens semelhantes ao que ele consumiu. Essa técnica necessita de dados fornecidos pelos usuários, como as avaliações de produtos. Lembrando que o usuário não precisa necessariamente avaliar todos os produtos do catálogo. No entanto, quanto mais informação a pessoa fornecer, mais precisas serão as recomendações.

Portanto, essa abordagem não é muito precisa quando trata-se de usuários novos, justamente porque eles ainda não classificaram muitos produtos, o que torna uma desvantagem para usar essa técnica de filtragem baseada no conteúdo. Porém, diferentemente da filtragem baseada em popularidade, ela consegue aprender sobre as preferências de cada usuário, lidando com o problema de generalização de sugestões. Essa técnica também lida melhor com novos itens, conseguindo recomendar novos produtos antes mesmos deles serem avaliados.

Esse sistema traça um perfil dos usuários a partir das características dos itens avaliados por eles. O perfil, no caso, seria uma representação estruturada do interesse de cada usuário. Assim, o processo de recomendação consiste basicamente em encontrar itens que possuem características que combinam com as presentes no perfil do usuário alvo (JUNIOR et al., 2017).

Por exemplo, um sistema possui três categorias de gêneros de animes e cem animes diferentes, sendo 50 animes de ação, 20 de drama e 30 de terror. Suponha que, a média das avaliações de cada gênero de um determinado usuário alvo foi, respectivamente, 6, 9 e 2. Dessa forma, o sistema constrói o perfil desse usuário como drama e recomenda animes desse mesmo gênero, que são mais similares aos que o usuário já consumiu.

3.2.3 Filtragem colaborativa baseada em memória

As recomendações dessa filtragem baseada em memória, o termo memória refere-se ao fato de que se baseia no passado, enquanto o termo colaborativa refere-se ao fato de que usuários colaboram entre si para recomendar itens. Esse sistema utiliza técnicas estatísticas para encontrar grupos de usuários ou itens (vizinhos) que apresentem similaridade entre si para fazer previsões de avaliações. Em outras palavras, a filtragem baseada em memória, envolve o histórico de concordância, isto é, usuários que concordaram no passado também concordarão no futuro (MEDEIROS, 2013).

Baseada em usuário - UBCF

A filtragem colaborativa baseada em usuário, ou em inglês "user based collaborative filtering"(UBCF), consiste na similaridade entre usuários diferentes. Nesse procedimento, utiliza-se uma matriz $U \times I$, onde as linhas **U** representam os usuários e as colunas **I** os itens. A entrada A_{ij} é a nota que o usuário i atribuiu ao item j, conforme a Figura 1. A partir dessa matriz, compara-se o usuário-alvo à k outros usuários utilizando medidas de similaridade. Depois de detectar o grupo de usuários mais similares, avalia-se os itens comprados por esse grupo a partir da média, frequência ou por pesos das avaliações. Em seguida recomenda-se o top-N itens mais bem avaliados.

USER-ITEM Matrix

	I_1	I_2		I_j		I_{m-1}	I_m
U_1			...		...		
U_2			...		...		
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
U_i			...	A_{ij}	...		
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
	.	.	.	.	.	.	.
U_{n-1}			...		...		
U_n			...		...		

Figura 1: Matriz Usuário-Item

De acordo com Hahsler (2021), para predizer a nota que o usuário alvo "a"deu ao item j, aplica-se a fórmula a seguir:

$$\hat{r}_{aj} = \frac{1}{|N(a)|} \sum_{i \in N(a)} r_{ij}, \tag{3.2.1}$$

onde r_{ij} é a nota que o usuário i atribuiu ao item j e $N(a)$ é a vizinhança, grupo de usuários mais similares, do usuário alvo. Uma vez que os usuários da vizinhança são encontrados, suas classificações são agregadas para formar a classificação predita para o usuário ativo.

Na Figura 2 tem-se um exemplo de filtragem colaborativa baseada em usuário. A direita tem-se a matriz usuário-item, ao lado as medidas de similaridade do usuário alvo (u_a) com os demais usuários (s_a) e por último a formação da vizinhança do usuário alvo com k=3.

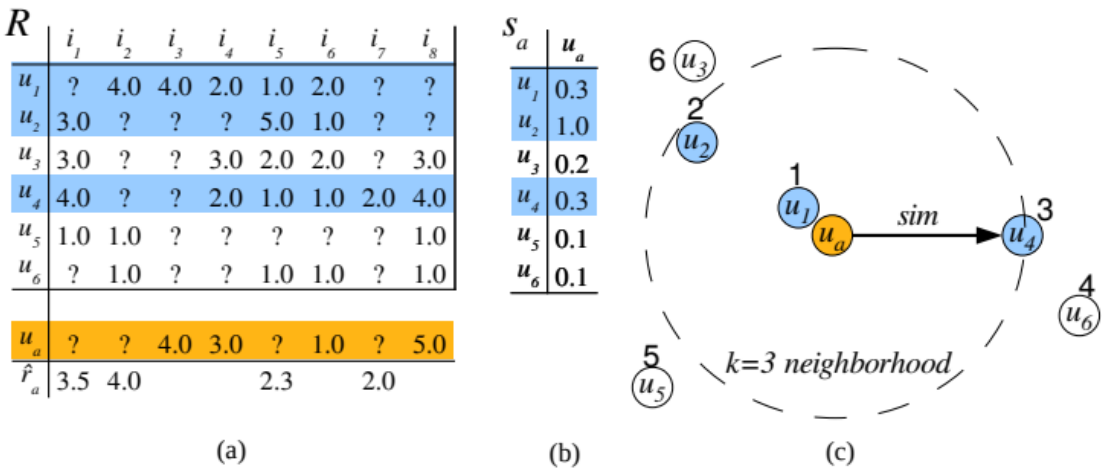

Figura 2: Exemplo UBCF (Fonte: Hahsler (2021))

A predição das notas do usuário alvo para os itens 1,2,5 e 7 seriam as seguintes:

$\hat{r}_{a1} = (3 + 4)/2 = 3.5.$

$\hat{r}_{a2} = (4)/1 = 4.$

$\hat{r}_{a5} = (1 + 5 + 1)/3 = 2.3.$

$\hat{r}_{a5} = (2)/1 = 2.$

Nota-se que o denominador muda a medida que a quantidade de usuários contidos na vizinhança que assistiram a um determinado item aumenta ou diminui. Por exemplo, o item 2 foi assistido apenas pelo usuário 1, então $|N(a)|$ é igual a um.

Há situações em que os k usuários semelhantes encontrados não são igualmente

semelhantes ao usuário alvo. Por exemplo, pode haver 3 usuários mais semelhantes ao usuário alvo do que os demais. Nesse caso, pode-se considerar uma abordagem utilizando uma média ponderada, em que a classificação do usuário mais semelhante seja mais importante do que o segundo usuário mais semelhante e assim por diante.

Na abordagem de média ponderada, multiplica-se cada classificação por uma medida de similaridade (que informa o quanto os usuários são semelhantes). Ao multiplicar pelo fator de similaridade, adicionam-se pesos às classificações. Quanto maior o peso, mais a classificação é relevante.

$$\hat{r}_{aj} = \frac{1}{\sum_{i \in N(a)} s_{ai}} \sum_{i \in N(a)} s_{aj} r_{aj}, \tag{3.2.2}$$

onde s_{ai} é a medida de similaridade entre o usuário alvo e o usuário u_i contido na vizinhança N(a).

Baseada em itens - IBCF

A filtragem colaborativa baseada em item ou em inglês "item based collaborative filtering"(IBCF), é um método que considera o conjunto de itens que o usuário-alvo avaliou no passado e calcula a semelhança entre cada item alvo "i"em relação aos demais itens disponíveis. Para cada dois itens, é medido o quão semelhantes eles são em termos de terem recebido classificações semelhantes por usuários semelhantes (vide Figura 3). Depois, para cada item, identifica-se os k-itens mais semelhantes, esses k itens são denotados como o conjunto S(i), o qual pode ser visto como a vizinhança de tamanho k do item i. Por fim, para cada usuário, identifica-se os itens que são mais semelhantes aos itens consumidos por ele e recomenda-se aqueles que o usuário ainda não viu (HAHSLER, 2021).

ITEM-ITEM SIMILARITY MATRIX

	I_1	I_2		I_j		I_{m-1}	I_m
I_1	1	Sim_{12}	...	Sim_{1j}	...		
I_2		1	...		...		
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
I_i		Simi2	...	Sim_{ij}	...		
	. . .	. . .	. . .	. . .	. . .	. . .	. . .
I_{m-1}			...		...	1	
I_m			...		...		1

Figura 3: Matriz de similaridade Item-Item

Para predizer a nota que o usuário alvo "a"deu ao item i, aplica-se a fórmula a seguir:

$$\hat{r}_{ai} = \frac{1}{\sum_{i \in S(i)} s_{ij}} \sum_{i \in S(i)} s_{ij} r_{aj}, \tag{3.2.3}$$

onde S(i) é a vizinhança de tamanho k do item i, $SIM_{ij} = s_{ij}$ é a similaridade do item i e o item j e r_{aj} é a nota que o usuário alvo fez ao item j, ou seja, caso o usuário não tenha avaliado o item j, r_{aj} é igual à zero. A seguir tem-se um exemplo de como essa fórmula é aplicada. A primeira matriz é a matriz usuário-item e a segunda é a matriz de similaridade item-item, utilizando a similaridade de cosseno.

$$\begin{array}{c} \\ Usuário \end{array} \begin{array}{c} \text{Item} \\ \begin{array}{cccc} & I_1 & I_2 & I_3 \end{array} \\ \begin{array}{c} U_1 \\ U_2 \\ U_3 \\ U_4 \end{array} \begin{pmatrix} 2 & ? & 3 \\ 5 & 2 & ? \\ 3 & 3 & 1 \\ ? & 2 & 2 \end{pmatrix} \end{array} \qquad \begin{array}{c} \\ \begin{array}{ccc} I_1 & I_2 & I_3 \end{array} \\ \begin{array}{c} I_1 \\ I_2 \\ I_3 \end{array} \begin{pmatrix} 1 & 0.9 & 0.78 \\ 0.9 & 1 & 0.86 \\ 0.78 & 0.86 & 1 \end{pmatrix} \end{array}$$

As similaridades foram calculadas da seguinte forma:

$s_{12} = (5 * 2 + 3 * 3)/(\sqrt{(25 + 9) * (4 + 9)}) = 0.9.$

$s_{13} = (2 * 3 + 3 * 1)/(\sqrt{(4 + 9) * (9 + 1)}) = 0.78.$

$s_{23} = (3 * 1 + 2 * 2)/(\sqrt{(9 + 4) * (1 + 4)}) = 0.86.$

Predizendo a nota do usuário 1 ao item 2 considerando k igual à 2 temos:

$\hat{r}_{12} = (s_{21} * r_{11} + s_{23} * r_{1,3})/(s_{21} + s_{23}).$

$\hat{r}_{12} = (2 * 0.9 + 3 * 0.86)/(0.9 + 0.86) = 2.48.$

Da mesma forma aplica-se para os usuários 2 e 4 aos itens 3 e 1, respectivamente.

$\hat{r}_{23} = (5 * 0.78 + 2 * 0.86)/(0.78 + 0.86) = 3.42.$

$\hat{r}_{41} = (2 * 0.9 + 2 * 0.78)/(0.9 + 0.78) = 2.$

Problemas da Filtragem colaborativa

A filtragem colaborativa possui alguns problemas quando existe um conjunto muito grande de usuários e itens, o que pode interferir na acurácia do algoritmo. Os dois principais problemas existentes em recomendação baseada em memória são os de esparsidade e escalabilidade (AZUIRSON, 2015).

- **Esparsidade:** quando existe uma grande quantidade de avaliações faltantes. Isso é muito comum pois nem todos os usuários avaliam itens e a quantidade de produtos é muito maior que a capacidade de indivíduos de avaliá-los. Isso pode dificultar na recomendação de itens novos ou itens que ainda não possuem avaliações, e na recomendação de itens para usuários novos. Os problemas de itens e usuários novos são chamados de problemas de inicialização. Para o caso do número de avaliações faltantes costuma-se utilizar as técnicas de fatoração de matrizes como a **Decomposição em Valores Singulares** para reduzir a dimensionalidade da matriz de usuários-itens.

- **Escalabilidade:** algoritmos baseados na memória são mais complexos computacionalmente à medida que aumenta-se o número de usuários e itens, dispondo de problemas de escala. Uma das técnicas para tratar desse problema é a clusterização.

De acordo com Barbosa (2014), quando o número de usuários é muito maior que o número de itens, é preferível uma recomendação colaborativa baseada em itens. Do mesmo modo, se o número de itens for maior que o de usuários, a recomendação colaborativa baseada em usuário é preferível pois demanda menos cálculos de similaridade.

Estudos indicam que o uso da correlação de Pearson apresenta desempenhos melhores para sistemas de filtragem colaborativa baseada em usuário, enquanto que a similaridade utilizando o cosseno de vetor possui uma boa perfomance nos algoritmos de filtragem colaborativa baseada em item. (GORAKALA; USUELLI, 2015).

3.3 Análise de Cluster

Análise de cluster é um método de aprendizagem não-supervisionado, isto é não possui uma variável resposta à ser predita, cujo objetivo de alocar em grupos objetos mais similares. Nesse trabalho, a análise de cluster será utilizada para criar um perfil para os usuários da seguinte forma, primeiro cria-se k clusters a partir da codificação "one hot"dos gêneros dos animes e, para cada, usuário descobre-se qual cluster é mais preferível. Então, os animes contidos nele serão recomendados para o usuário alvo. Na secção de resultados há um exemplo de como será feita a filtragem baseada em conteúdo junto à análise de cluster.

3.3.1 Método k-médias (K-means)

É um método não hierárquico de aglomeração com objetivo de alocar os elementos em k clusters (grupos), com o número k determinado como parte do procedimento ou, antecipadamente pelo responsável do estudo. Esse agrupamento, de acordo com Johnson e Wichern (2007), se baseia no isolamento dos clusters formados e nas similaridades internas.

MacQueen et al. (1967) propõe a técnica na qual cada elemento da amostra seja alocado ao cluster que contenha o centroide mais próximo (média). Tal técnica é constituída pelos próximo passos:

1. Escolher os k centroides (sementes) para iniciar o processo de partição;
2. Comparar cada um dos componentes com o centroide inicial por um tipo de distância euclidiana. Os componentes são alocados aos clusters pelo critério de menor distância;
3. Depois da alocação dos n componentes, recalcula-se os centroides para cada novo cluster formado, redefinindo o passo 2 com base nesses novos centroides;
4. Refazer os passos 2 e 3 até que todos os componentes estejam perfeitamente alocados em seus grupos, ou seja, até que não seja necessária mais alocações.

3.3.2 Método do Cotovelo (Elbow Method)

Um dos principais problemas a ser resolvido ao aplicar algoritmos de clusterização em conjuntos de dados é definir a quantidade ideal de clusters para se obter o melhor agrupamento. Esta decisão é peculiar a cada problema, variando conforme o conjunto de dados e o resultado esperado.

Segundo Kodinariya e Makwana (2013), um método bastante utilizado para determinar o melhor número de clusters é o *Elbow Method* ou Método do Cotovelo. Este é um método visual. Inicia-se com $k = 2$ (número de clusters) e a cada nova etapa soma-se 1 ao valor de k, calculando a variância dos dados em relação ao número de clusters. O algoritmo para quando a soma dos quadrados intra-clusters (ou do inglês, within-clusters sum-of-squares, comumente abreviado para wcss) é a menor possível.

Vale destacar que o método chama-se de curva de cotovelo, porque a partir do ponto em que seria o "cotovelo" não existe uma discrepância tão significativa em termos da soma dos quadrados intra-clusters. Dessa forma, a melhor quantidade de clusters K seria exatamente onde o cotovelo estaria.

Supondo que um site de streaming de anime contenha 100 animes classificados de forma binária por 5 gêneros, sendo que cada anime pode pertencer a um ou mais gêneros da seguinte forma:

Gênero	Ação	Drama	Terror	Fantasia	Aventura
anime 1	1	1	1	0	0
anime 2	0	1	0	1	0
anime 3	1	0	1	0	0
anime 4	1	0	1	0	1
...	...	...	...	...	...
anime 100	0	1	0	1	1

Para fazer a separação em cluster, aplica-se a distância euclidiana como no exemplo visto anteriormente na seção 3.2.1, e a partir da soma dos quadrados intra-clusters obtêm-se os seguintes resultados:

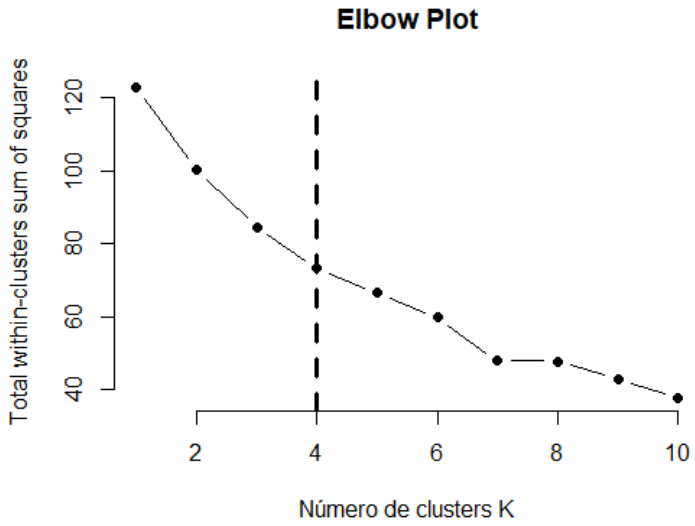

Figura 4: Exemplo Elbow

Nota-se que quando k é igual à 4 têm-se o que seria o "cotovelo", onde a diferença entre a soma dos quadrados intra-clusters não é muito significativa. Dessa forma, pode-se inferir que o número ótimo de cluster a serem formados é igual à 4.

3.3.3 K – Nearest Neighbors (KNN)

K-nearest-neighbor (KNN) é um método de classificação quando há pouco ou nenhum conhecimento sobre a distribuição dos dados, ou quando é muito difícil estimar a função de densidade. (PETERSON, 2009).

Esse classificador usa alguma medida de similaridade para determinar a proximidade entre o conjunto de treinamento e o de teste, isto é, ele compara a distância entre um elemento da amostra de teste aos elementos da amostra de treinamento, aqueles K elementos com as menores proximidades do elemento alvo da classificação são chamados de K-vizinhos mais próximos.

Na Figura 5 temos um exemplo de como o KNN funciona. Se olharmos para os três vizinhos mais próximos do ponto verde, um azul e dois vermelhos, o método KNN classificará como vermelho, pois, há mais triângulos vermelhos do que quadrados azuis. Mas se procurarmos pelos seis vizinhos mais próximos, então o ponto verde será classificado como azul.

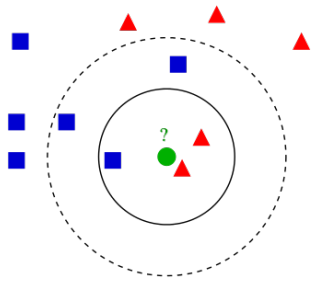

Figura 5: Exemplo KNN

A maior diferença entre os métodos de KNN e K-means é que o primeiro é um método supervisionado de classificação, porque usa os rótulos de classe dos dados de treinamento, enquanto os métodos de agrupamento não empregam os rótulos de classe dos dados de treinamento. (PETERSON, 2009)

O algoritmo KNN é comumente utilizado para os sistemas de recomendações baseados em memória, pois, nem sempre o usuário mais parecido, terá pontuado todos os itens do conjunto de dados, assim é importante definir uma certa quantidade de usuários similares. Nesse caso, onde o usuário mais similar encontrado não pontuou em todos os itens do conjunto de dados, é necessário o tratamento através do algoritmo KNN que tem por objetivo definir a vizinhança em ordem de similaridade, ou seja, o vizinho mais próximo é o mais similar e assim sucessivamente. E em tal situação, pode-se usar os dados de outros usuários similares (vizinhos) caso o mais similar (vizinho mais próximo) não tenha pontuado em todos os itens do conjunto de dados (FREITAS, 2018).

Um exemplo prático dessa técnica aos sistemas de recomendação pode ser visto na Figura 2.

3.4 Medidas de Avaliação de Desempenho

Para avaliar se o sistema de recomendação está ofertando itens que são realmente relevantes aos seus usuários, diversas métricas têm sido utilizadas. Entre elas, destacam-se as medidas de exatidão preditiva, que avalia o quão próximo dos valores reais são os valores previstos pelos sistemas de recomendação, como o erro médio absoluto (MAE) e o erro quadrático médio (MSE). E as medidas de exatidão de classificação, que avalia a frequência com a qual os sistemas de recomendação fazem recomendações corretas ou incorretas, como as matrizes de confusão e a curva roc, como visto em Barbosa (2014).

3.4.1 MAE

O erro médio absoluto (MAE) calcula a média de todas as diferenças de valor absoluto entre a avaliação verdadeira e a prevista. Quanto menor o MAE, melhor é a precisão do modelo. O cálculo do erro médio absoluto é dado por:

$$MAE = \frac{\sum_{i=1}^{n} |y_i - x_i|}{n} = \frac{\sum_{i=1}^{n} |e_i|}{n}. \tag{3.4.1}$$

3.4.2 MSE

O erro quadrático médio (MSE ou EQM) calcula o valor médio de todas as diferenças quadradas entre as avaliações verdadeiras e previstas. Essa medida fornece um erro de predição. Quanto maior é o MSE, maior é o afastamento entre a avaliação real e a predita, então o algoritmo está errando mais do que deveria. A fórmula do EQM é dada por:

$$MSE = E[(\hat{\theta}_i - \theta_i)^2] = \sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}. \tag{3.4.2}$$

Além disso, a raíz quadrada do erro quadrático (RMSE) médio é dada por:

$$RMSE = \sqrt{\sum_{i=1}^{n} \frac{(\hat{y}_i - y_i)^2}{n}}. \tag{3.4.3}$$

3.4.3 Matrizes de confusão

Uma matriz de confusão é construída usando os resultados de modelos de classificação, comparando os resultados preditos e os reais.

Verdadeiro Positivo (TP): classificação correta da classe Positivo.

Falso Positivo (FP): classificação errada, na qual o modelo previu a classe Negativo quando o valor real era classe Positivo.

Verdadeiro Negativo (TN): classificação errada, na qual erro o modelo previu a classe Positivo quando o valor real era classe Negativo.

Falso Negativo (FN): classificação correta da classe Negativo.

[1]https://diegonogare.net/2020/04/performance-de-machine-learning-matriz-de-confusao/

		Valor Predito	
		Sim	**Não**
Real	**Sim**	Verdadeiro Positivo (TP)	Falso Negativo (FN)
	Não	Falso Positivo (FP)	Verdadeiro Negativo (TN)

Figura 6: Matriz de Confusão[1]

Com os valores dessa matriz podemos, calcular métricas de avaliação para o modelo, como a precisão e a sensibilidade, que são dadas por:

$$Precisão = \frac{TP}{TP + FP} \tag{3.4.4}$$

$$Sensibilidade = \frac{TP}{TP + FN} \tag{3.4.5}$$

A precisão mostra como os modelos são sensíveis aos falsos positivos, isto é, o sistema recomenda um item com pouca probabilidade de ser comprado. Já a sensibilidade analisa o quão sensíveis os modelos são aos falsos negativos, ou seja, o sistema não sugere um item que é altamente provável comprado.

Curva Característica de Operação do Receptor (ROC Curve)

É uma representação gráfica de desempenho, que permite selecionar modelos possivelmente ideais e descartar aqueles que não são tão ótimos. No eixo x temos a taxa de falsos positivos (FP/(TN+FP)) e no eixo y a taxa de verdadeiros positivos (TP/(TP+FN)). Quanto mais a extrema esquerda estiver o ponto de corte escolhido, maior será a sensibilidade e especificidade (FN/(FN+TN)), que mede a capacidade de se predizer um item como não relevante de forma correta.

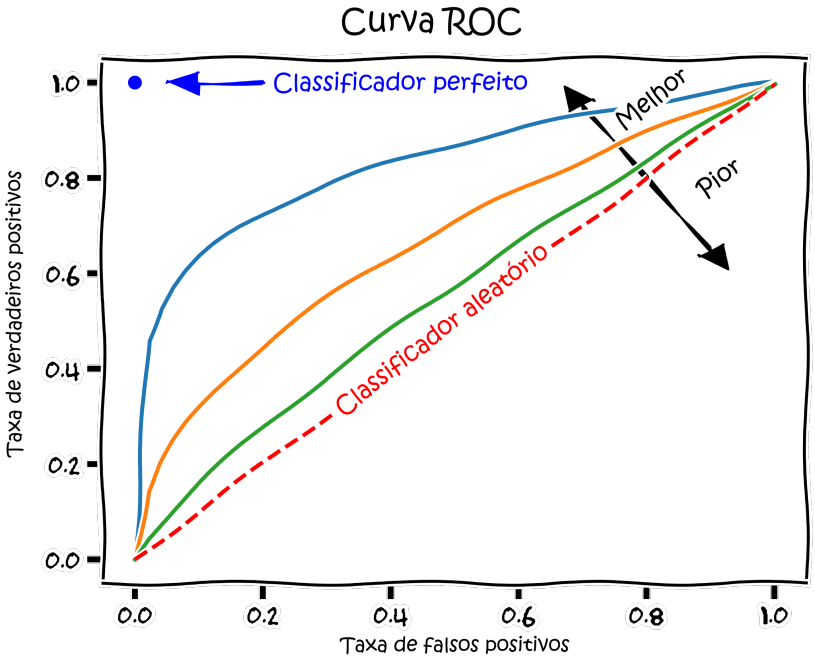

Figura 7: Curva Roc exemplo [2]

[2]https://pt.m.wikipedia.org/wiki/Ficheiro:Curva$_R OC.svg$

4 Materiais e Métodos

4.1 Materiais

Foram coletados dois conjuntos de dados: o primeiro (animes.csv) contendo uma lista de informações de diferentes animes com título, descrição, gênero, popularidade, etc, contendo 12.294 animes diferentes. O segundo (rating.csv) refere-se às avaliações que cada usuário fez de um determinado anime, juntamente com o id (identificador) do anime e do usuário, contendo um total de 73.516 usuários distintos e 69.600 avaliações. Os dados foram retirados por terceiros do site MyAnimeList.net por meio de web scraping e podem ser encontrados na plataforma online da comunidade de cientistas de dados Kaggle em formato CSV [1].

As avaliações dos usuários que avaliaram apenas alguns animes e as avaliações dos animes que foram vistos apenas algumas vezes podem ser tendenciosas. Sendo assim, para a criação dos modelos, foram utlizado apenas os usuários que avaliaram pelo menos 50 animes e apenas os animes que receberam pelo menos 100 avaliações, totalizando 9.927 animes, equivalente a quase 41% do total animes, e 32.833 usuários, equivalente a cerca de 47%.

4.2 Métodos

Nesse trabalho serão aplicados os sistemas de recomendações baseados em conteúdo, por popularidade e as filtragens colaborativas baseadas em usuário e item descritos na seção de referencial teórico. Os algoritmos serão implementados utilizando o software estatístico R (R Development Core Team, 2009) em conjunto com o pacote *recommenderlab* (HAHSLER, 2021).

Os métodos de agrupamento k-means e KNN serão empregados na criação de sistemas de recomendação baseados em conteúdo e na filtragem colaborativa, descritos na Seção 3.

Para a filtragem baseada em conteúdo funcionará da seguinte forma, primeiro a técnica de clusterização k-means será utilizada para agrupar os animes com base na informação de gênero de cada anime. Esse procedimento é necessário para decidir qual cluster de anime o usuário alvo prefere e assim criar um perfil de gêneros favorito de cada

[1] https://www.kaggle.com/CooperUnion/anime-recommendations-database

usuário. Em seguida, para cada usuário, ultiliza-se pesos para ponderar cada gênero do perfil do usuário e, a partir disso obter o possível nível de interesse do usuário dos animes ainda não vistos. Na seção dos resultados será explicado o método com um exemplo.

Para avaliar os modelos das filtragens por popularidade e colaborativa, a base de dados foi separada em 80% para treino e 20%. Assim, as notas dadas pelos usuários pertencentes à base de teste foram utilizadas para comparar com as notas previstas pelo modelo. Essa separação foi utilizada por dois motivos, primeiro para evitar viés de seleção entre modelos e evitar overfitting, e segundo porque o processamento de todo o conjunto de dados é computacionalmente muito caro. Já na filtragem baseada em conteúdo essa separação não foi utilizada já que não há necessidade das informações de outros usuários para gerar recomendações de um usuário alvo, sendo assim, computacionalmente mais viável a utilização de toda base de dados.

A performance dos algoritmos são avaliada por medidas de avaliação de desempenho apresentadas na Seção 3.

5 Resultados

5.1 Análise Descritiva

A matriz de usuário e item UxI, com as entradas a_{ij} representando a avaliação do usuário i para o item j, possui 32967 linhas, 4143 colunas e 5562654 elementos preenchidos, correspondendo a cerca de 4.07% das possíveis avaliações. Dessa forma, percebe-se que é uma matriz esparsa e que a maior parte dos elementos da matriz não possui valores.

A Figura 8 apresenta as distribuições do número de avaliações por usuário e por anime. A tabela resumo dessas distribuições está apresentada na Tabela 1.

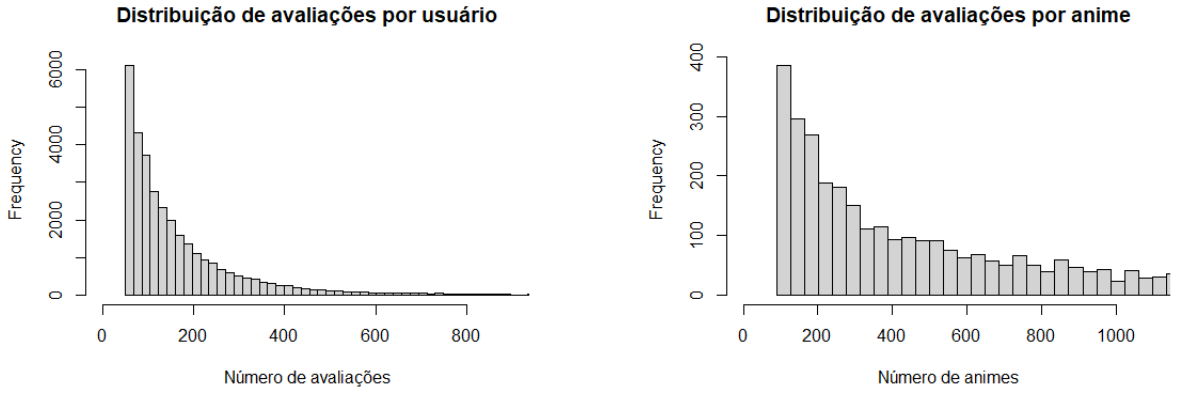

Figura 8: Distribuição do número de avaliações por usuário e por anime

Avaliações	Min	1º quartil	Mediana	Média	3º quartil	Max	Desvio Padrão
por usuário	50	77	120	169	204	2.790	150.47
por anime	100	223	549	1358	1498	22.448	2116.56

Tabela 1: Tabela do número de avaliações

A partir da Figura 8 e da Tabela 1, nota-se que, entre os usuários, 25% avaliaram 77 animes ou menos, nota-se que a maior parte (75%) avaliou até 204 animes. O desvio padrão e o desvio interquartílico do número de animes avaliados por usuário são iguais à 150.47 e 127 respectivamente, indicando uma grande variabilidade na quantidade de animes avaliados. Metade dos usuários avaliou pelo menos 120 animes, com média de, aproximadamente, 169 animes avaliados por usuário. Além disso, o número de avaliações por usuário apresenta uma assimetria à direita.

Para a distribuição do número de avaliações por anime, pode-se ver, pela Tabela 1, que 25% dos animes receberam 223 ou mais avaliações e que 50% dos animes receberam 549 ou menos avaliações, com média de 1358 avaliações por anime, ou seja, os valores no topo (moda) da distribuição estão muito distantes do centro (média). Além disso, o desvio interquartílico é 1275 e o desvio padrão foi, aproximadamente, 2116.56, indicando uma grande variabilidade na quantidade de avaliações por animes.

Na Figura 9, tem-se a distribuição da nota de todas as avaliações e a distribuição da nota média das avaliações por usuário.

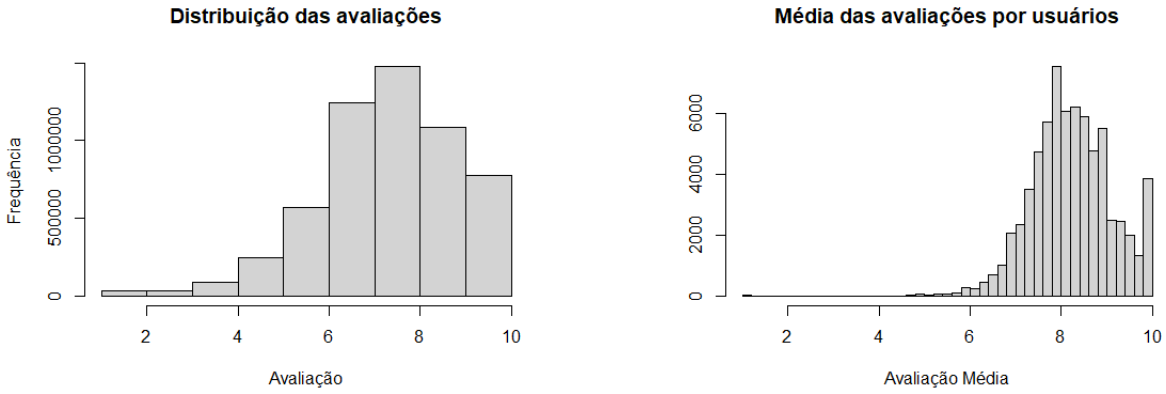

Figura 9: Distribuição das notas dos animes e da nota média por usuário

Nota-se, pela Figura 9, que a moda (valor mais frequente) das avaliações é igual a 8. A média obtida na distribuição das avaliações foi igual a 7.8 e mediana igual à 8. Pode-se notar pela Figura 9 que o número de avaliações iguais a 10 equivale à soma das avaliações com nota menor ou igual a 3. Na distribuição da média das avaliações por usuários, tem-se que a maioria das pessoas avalia que gostam dos programas, e que muitas pessoas dão apenas notas 10. Além disso, observa-se uma distribuição bimodal para a média das avaliações com moda em torno de 8,0 e 10.

A Tabela 2 contem os 10 animes mais avaliados, representando o número de usuários que assistiram o anime, o número de avaliações e o nome do anime a média das avaliações recebidas.

Rank	Nome do Anime	Número de Vezes Assistido	Número de Avaliações	Média
1	Death Note	22.947	22.448	8.83
2	Sword Art Online	19.813	19.164	8.14
3	Code Geass: Hangyaku no Lelouch	19.079	18.545	8.92
4	Shingeki no Kyojin	19.098	18.371	8.72
5	Angel Beats!	18.792	18.300	8.59
6	Elfen Lied	18.112	17.537	8.06
7	Toradora!	17.670	17.102	8.59
8	Code Geass: Hangyaku no Lelouch R2	17.197	16.697	9.05
9	Fullmetal Alchemist: Brotherhood	16.576	16.113	9.32
10	Highschool of the Dead	16.309	15.647	7.66

Tabela 2: Animes mais avaliados

Pode-se perceber pela Tabela 2 que nem todo usuário avalia os animes que assistiu, já que os números da primeira colunas são muitos maiores que os da segunda coluna. Além disso, a maioria desses animes receberão em média nota acima de 8, com exceção do último anime.

A Tabela 3 apresenta a quantidade de animes dos 10 principais gêneros e a Figura 10 é uma *wordcloud* com todos os gêneros existentes.

Rank	Gênero	Nº animes	Frequência Relativa
1	Comédia	1996	0.1222
2	Ação	1425	0.0875
3	Romance	997	0.0610
4	Drama	993	0.0608
5	Sci-fi	950	0.0581
6	Shounen	926	0.05670
7	Fantasia	925	0.0566
8	Aventura	920	0.0563
9	School	718	0.0439
10	Sobrenatural	642	0.0393

Tabela 3: Gêneros Assistidos e existentes

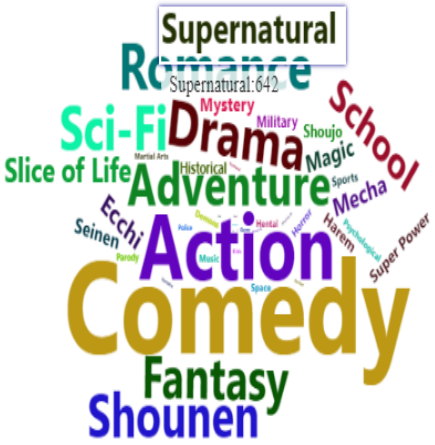

Figura 10: Gêneros existentes

Pode-se perceber que a maioria dos animes foi classificada com o gênero comédia, seguidos de ação, romance e drama, lembrando que cada anime pode ter sido descrito

com mais de um gênero, então a soma das frequências é maior que o número de animes. A distribuição do número de animes por gênero está bem equilibrada aumentando a precisão das recomendações feitas pela filtragem baseada em conteúdo, pois, se alguma das categorias tivesse mais de 80% dos animes, o sistema de recomendação por conteúdo iria apenas gerar recomendações de animes dessa categoria.

A Figura 11 apresenta a quantidade de animes por categorias de programa existentes.

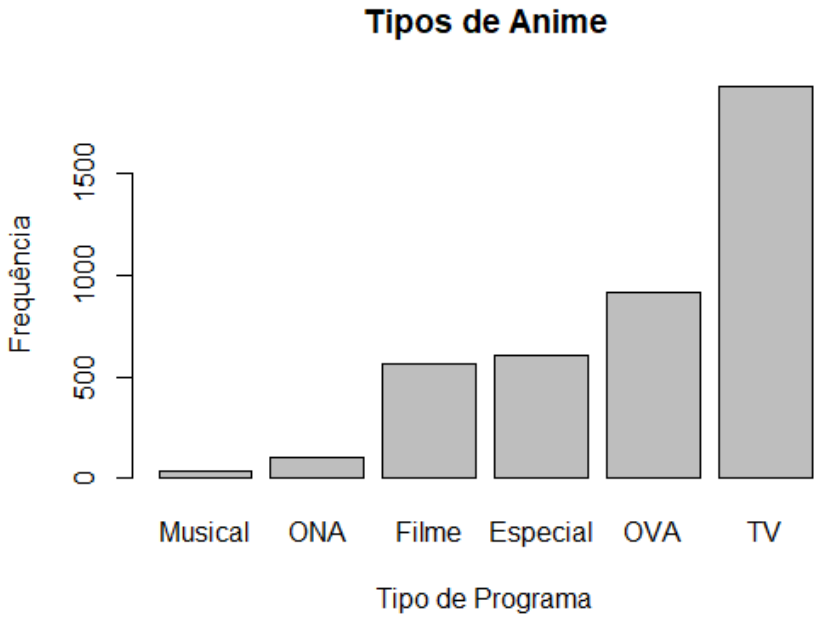

Figura 11: Distribuição dos Tipos de animes

Nota-se, pela Figura 11, que os seriados da TV são mais predominantes, em seguida os OVA (Original Video Animation), que são episódios lançados no mercado ou simplesmente direto em DVD e não na TV. Os filmes de animes e os especiais, que podem ser vistos na televisão ou em DVD, também possuem uma quantidade expressiva. Além disso, não existe uma quantidade grande de animes musicais, e nem de ONA (Original Network Animation), que são o tipo de anime que lança seus episódios pela internet.

A Figura 12 apresenta um mapa de calor dos 20 usuários que mais avaliaram os animes e os 17 animes mais assistidos. As colunas representam os animes e as linhas representam os usuários, e cada quadrado é a avaliação que o usuário j atribuiu ao item i. Quanto mais escuras as colunas forem, maior será a audiência do respectivo anime, e quanto mais escura a linha for, mais altas serão as avaliações dadas pelo respectivo

usuário. A célula com cor branca significa que o usuário ainda não avaliou o item.

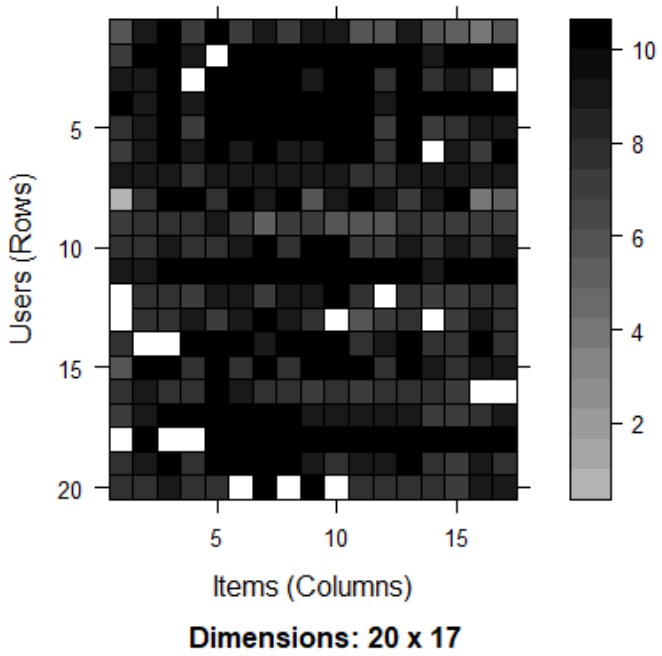

Figura 12: Heatmap para os top usuários e animes

Nota-se, pela Figura 12, que algumas linhas são mais escuras do que outras, isso significa que alguns usuários fornecem classificações mais altas a todos os animes. Ter usuários que dão classificações altas (ou baixas) a todos os animes pode distorcer os resultados. Por exemplo, alguns usuários são mais "tolerantes" e, portanto, suas classificações dos itens tendem a ser maiores do que a de outros usuários, mesmo que eles compartilhem muitos gostos semelhantes (JIN; SI, 2004). Esse viés é removido através da normalização dos dados, neste trabalho utilizará o método de normalização que subtrai de cada classificação disponível a média das classificações daquele usuário (linha).

5.2 Sistemas de Recomendações

Filtragem Baseada em Conteúdo

O método não hierárquico de agrupamento K-means foi utilizado para agrupar os animes por gênero utilizando a distância euclideana. Pelo gráfico do cotovelo (elbow plot), vide Figura 13, determinou-se que a quantidade ideal de clusters é 3.

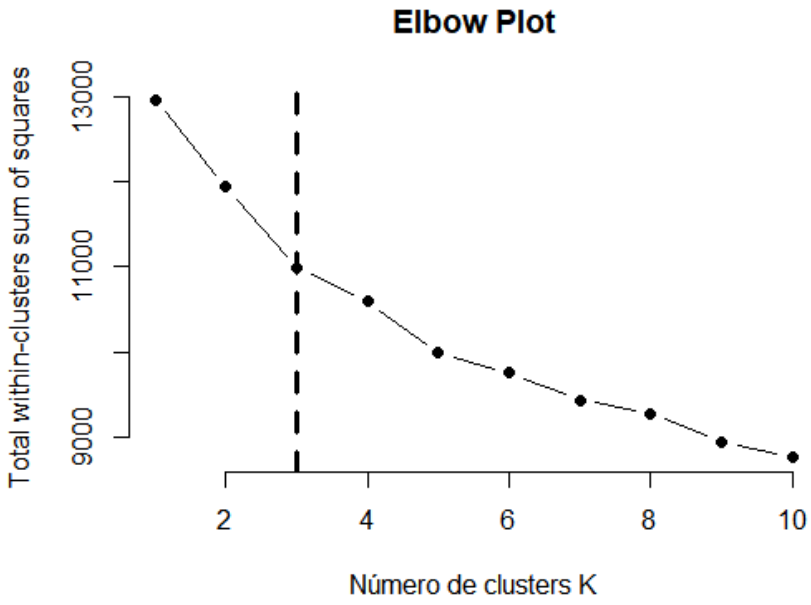

Figura 13: Elbow Plot

A Figura 14 apresenta os gêneros contidos em cada cluster criado. O primeiro cluster contém os animes com os gêneros: ação e Sci-Fi com maior frequência, enquanto no cluster 2, os gêneros que se destacam são comédia e drama e no cluster 3, os gêneros de comédia e romance. O primeiro cluster possui 7.243 (33.26%) animes, enquanto o segundo e terceiro possuem 2.229 (44.99%) e 2.822 (21.75%) animes, respectivamente. A distribuição dos animes nos clusters está bem equilibrada, favorecendo a filtragem baseada em conteúdo.

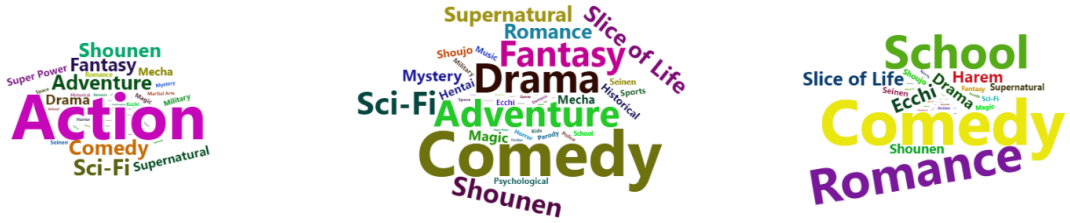

Figura 14: Gêneros contidos nos cluster 1, 2 e 3

O algoritmo de filtragem baseada em conteúdo será executado da seguinte maneira:

1. Encontra-se o cluster de anime mais preferível pelo usuário alvo através da média

das avaliações dadas pelo usuário alvo de cada cluster e em caso de empates, o cluster escolhido é aquele com maior quantidade de animes assistidos pelo usuário. O perfil do usuário será criado a partir dos gêneros de animes contidos no cluster selecionado.

2. Atribuem-se pesos a cada gênero do perfil do usuário da seguinte forma. Primeiro multiplica-se o vetor de avaliações do usuário pela matriz "dummy"de gênero dos animes que o usuário já assistiu (vide exemplo abaixo). Em seguida, soma as colunas da matriz de gêneros ponderados e divide os valores pelo total de peso, para que a soma dos peso seja igual à 1. Este seria o vetor de pesos do perfil do usuário.

3. Multiplica-se o vetor de pesos do perfil do usuário pela matriz "dummy" de gêneros de animes que o usuário ainda não assistiu, resultando na matriz de animes ponderada. Depois somando as linhas para obter os possíveis níveis de interesse do usuário dos animes não vistos.

4. Recomendam-se os animes com o nível de interesse mais alto.

Considere o exemplo a seguir, a esquerda tem-se o vetor de avaliações para os 5 animes que o usuário avaliou. Esse vetor está sendo multiplicado pela matriz dummy dos gêneros presentes no seu perfil.

$$\begin{pmatrix} 10 \\ 9 \\ 4 \\ 8 \\ 6 \end{pmatrix}^{T} \times \begin{pmatrix} 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 1 \\ 1 & 0 & 1 & 1 \\ 0 & 1 & 1 & 0 \\ 1 & 0 & 1 & 0 \end{pmatrix} = \begin{pmatrix} 10 & 10 & 0 & 10 \\ 0 & 9 & 0 & 9 \\ 4 & 0 & 4 & 4 \\ 0 & 8 & 8 & 0 \\ 6 & 0 & 6 & 0 \end{pmatrix}$$

O vetor do perfil do usuário é igual à

$$\begin{pmatrix} 20/88 & 27/88 & 18/88 & 23/88 \end{pmatrix} = \begin{pmatrix} 0.738 & 0.465 & 0.795 & 0.306 & 0.261 & 0.545 \end{pmatrix}$$

Considere ainda a seguinte matriz "dumy"de gêneros relativas a 6 animes que o usuário não assistiu

$$\begin{pmatrix} 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 \\ 1 & 1 & 0 & 1 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 1 \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

Multiplicando o vetor do perfil do usuário pela matriz " dumy" de gêneros de animes que o usuário ainda não assistiu tem-se

$$\begin{pmatrix} 20/88 & 27/88 & 18/88 & 0 \\ 0 & 0 & 18/88 & 23/88 \\ 20/88 & 27/88 & 0 & 23/88 \\ 0 & 27/88 & 0 & 0 \\ 0 & 0 & 0 & 23/88 \\ 20/88 & 0 & 18/88 & 0 \end{pmatrix}$$

Em seguida, soma-se as linhas para obter o possível nível de interesse do usuário dos animes não vistos. Por fim, multiplicá-se os níveis de interesse por 10 para estarem em um intervalo de 0 a 10, o que será considerado a predição da nota do usuário alvo.

$$\begin{pmatrix} 65/88 \\ 41/88 \\ 70/88 \\ 27/88 \\ 23/88 \\ 48/88 \end{pmatrix} = \begin{pmatrix} 0.738 \\ 0.465 \\ 0.795 \\ 0.306 \\ 0.261 \\ 0.545 \end{pmatrix} \times 10 = \begin{pmatrix} 7.38 \\ 4.65 \\ 7.95 \\ 3.06 \\ 2.61 \\ 5.45 \end{pmatrix}$$

No exemplo, o sistema de recomendação iria recomendar o terceiro anime, depois o primeiro, e assim por diante.

Filtragem Colaborativa Baseada em Item

Para o modelo, a matriz UxI foi normalizada, ou seja, subtraiu-se de cada classificação disponível a média das classificações daquele usuário (linha). Para a filtragem colaborativa baseada em item (IBCF) foi utilizado a similaridade de cosseno para encontrar os k itens mais similares a cada item previamente avaliado pelo usuário alvo. Neste trabalho foi utilizado k=30, visto que é o default das funções do pacote "recommenderlab".

Filtragem Colaborativa Baseada em Usuário

Para este modelo, a matriz UxI foi normalizada, ou seja, subtraiu-se de cada classificação disponível a média das classificações daquele usuário (linha). Para a filtragem colaborativa baseada em usuário (UBCF), foi utilizado o método de KNN para encontrar os k (no caso k=25) usuários mais semelhantes ao usuário alvo usando a similaridade de pearson. O valor de k também foi determinado pelo default do pacote "recommenderlab".

Filtragem por Popularidade

Para este modelos de filtragem por popularidade, a matriz UxI foi normalizada, ou seja, subtraindo de cada classificação disponível a média das classificações daquele usuário (linha). Foi utilizada uma abordagem onde a predição da nota é dada pela média das avaliações de cada item com base nas avaliações disponíveis, e prevê cada nota desconhecida como a nota média para o item. Ou seja, a filtragem por popularidade baseada na média vai recomendar os animes com maior média de avaliações e não os animes mais vistos. Portanto, para qualquer usuário os animes recomendados estão apresentados na Tabela 4.

Tabela 4: Recomendações por Popularidade

Rank	Animes Recomendados	Nota Prevista
1	Ginga Eiyuu Densetsu	9.09
2	GintamaÂ°	8.67
3	Kimi no Na wa.	8.49
4	Gintama'	8.46
5	Steins;Gate	8.45
6	Fullmetal Alchemist: Brotherhood	8.44
7	Gintama': Enchousen	8.42
8	Gintama Movie: Kanketsu-hen - Yorozuya yo Eien Nare	8.41
9	Gintama	8.40
10	Haikyuu!!: Karasuno Koukou VS Shiratorizawa Gakuen Koukou	8.38

Análise de Desempenho

Para a análise de desempenho, foi considerado que um usuário gostou de um anime quando ele deu uma nota 5 ou mais.

Na Tabela 5 tem-se medidas de desempenhos RMSE, MSE e MAE para cada modelo.

Sistema de Recomendação	RMSE	MSE	MAE
IBCF	**1.4383**	**2.0688**	**0.9619**
UBCF	1.6465	2.7112	1.2470
POPULARIDADE	1.5349	2.3561	1.1684
CONTEÚDO	1.5245	2.3243	1.1943

Tabela 5: Medidas de Avaliação de Desempenho

O modelo de filtragem baseada em item apresenta os menores erros dessa forma, o modelo apresenta a melhor performance. Além disso, a filtragem baseada em usuário é a que tem maiores valores nas suas medidas de desempenho, portanto apresenta pior desempenho. No entanto, a diferença dos valores das medidas de erros são bem próximas, não havendo evidências para rejeitar nenhum dos modelos.

A Figura 15 apresenta a curva ROC, que traça a taxa de verdadeiro positivo (TPR) contra a taxa de falso positivo (FPR), onde os pontos são o número de recomendações que são iguais à 10, 20, 30, ..., 100.

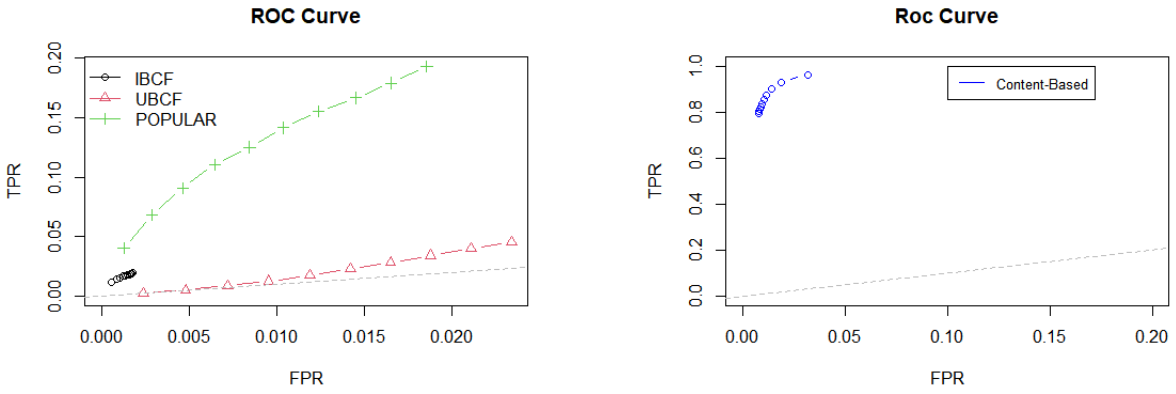

Figura 15: Gráfico Curva ROC

Os modelos de filtragem por popularidade e de filtragem baseada em conteúdo são os com melhores desempenhos, pois atingem o TPR mais alto para qualquer nível de FPR. Isso significa que o modelo está produzindo o maior número de recomendações relevantes (verdadeiros positivos) para o mesmo nível de recomendações não relevantes (falsos positivos). Pode-se perceber que, as curvas roc da filtragem baseada em usuário é pouco melhor que uma curva de um classificador aleatório, isto é, o sistema está recomendando a mesma quantidade de itens relevantes e não relevantes, sendo uma indicação de que essa filtragem não é a melhor para o problema de streaming de animes. Percebe-se também

que o TPR da filtragem baseada em item quase não muda em relação ao nível de FPR a medida que aumenta a quantidade de recomendações.

Na Figura 16 tem-se o gráfico de precisão-sensibilidade. A precisão mostra como os modelos são sensíveis aos falsos positivos (isto é, o sistema recomenda um item com pouca probabilidade de ser comprado), enquanto a sensibilidade analisa o quão sensíveis os modelos são aos falsos negativos (ou seja, o sistema não sugere um item que é altamente provável que seja comprado).

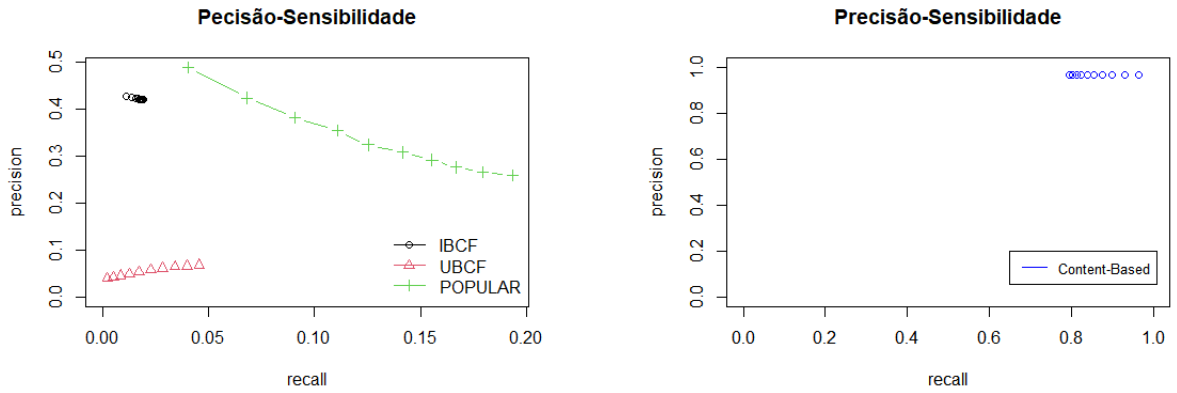

Figura 16: Gráfico Precisão-Sensibilidade

Normalmente, preocupa-se em prever com precisão quais itens são mais propensos a serem comprados, pois isso teria um impacto positivo nas vendas. Em outras palavras, deve-se maximizar o recall (sensibilidade), ou seja, minimizar a quantidade de falsos negativos (FN) para o mesmo nível de precisão. Nota-se que isso também é garantido em todos os níveis pelos modelos de filtragem por popularidade e baseada em conteúdo, apresentando assim um bom desempenho.

Shiny

Por fim, como parte do objetivo do trabalho, foi criado um aplicativos da web interativos utilizando o pacote Shiny do software R. Devido à limitação de memória, o aplicativo utilizou apenas os animes que receberam pelo menos 450 avaliações (equivalente à 2.339 animes) e os usuários que avaliaram pelo menos 500 animes (cerca de 2.5% dos usuários da base de dados).

Na Figura 16 tem-se o primeiro menu do aplicativo, contendo os animes disponíveis para avaliação com os números que serão as notas dadas pelo usuário aos respectivos animes.

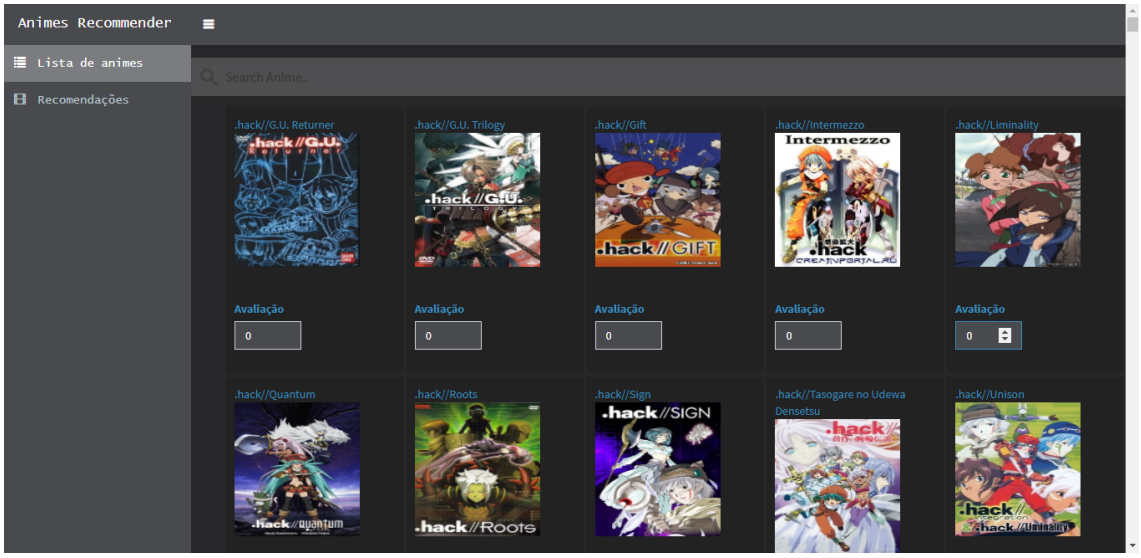

Figura 17: Shiny Primeira Página

Depois que o usuário termina de avaliar todos os animes que já assistiu e clica no botão de "Gerar Recomendações", o servidor cria uma matriz usuário-item e utiliza os modelos criados para predizer as notas dos animes e recomendar os animes ainda não assistidos, como mostra a Figura 17.

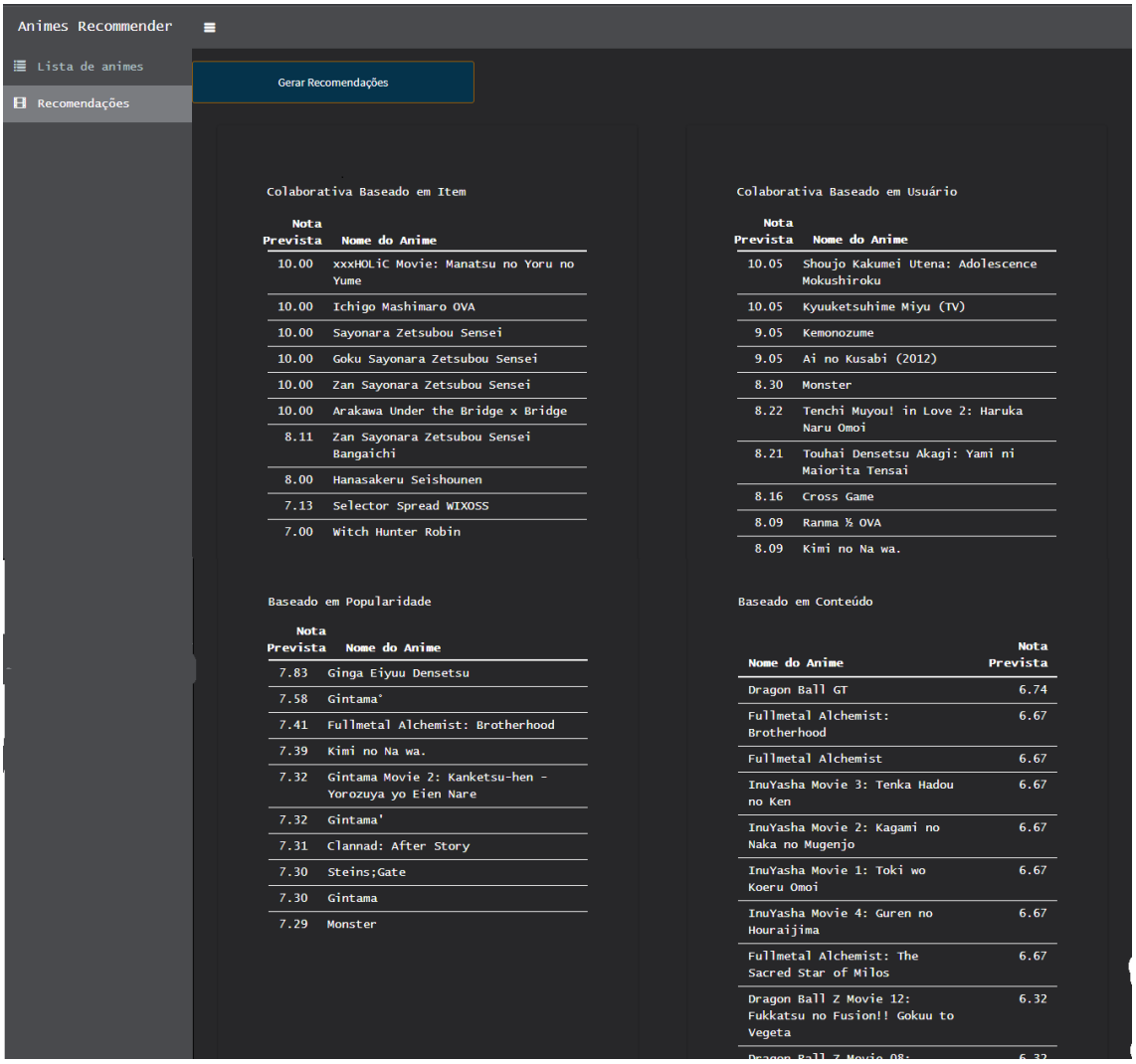

Figura 18: Shiny Segunda Página

O aplicativo pode ser encontrado pelo domínio ⟨https://larissamoreno.shinyapps.io/recommendation/⟩.

6 Conclusão

Para o conjunto de dados fornecido, os métodos de filtragem por popularidade e de filtragem baseada em conteúdo forneceram os melhores resultados. O algoritmo de UCBF foi descartado, o que não é surpreendente, pois o número de usuários é muito maior que o número de itens e, de acordo com Barbosa (2014) é um fator que prejudica o modelo de filtragem baseada em usuário.

A abordagem baseada em conteúdo tem melhor desempenho quando se trata de complexidade computacional do que os métodos de filtragem colaborativa. As filtragens colaborativas demoram mais de dez minuto para fazer os cálculos e exibir trinta animes recomendados, enquanto o método baseado em conteúdo pode fazer isso em menos de vinte segundos, pois o algoritmo baseado em conteúdo não requer informações sobre outros usuários, mas apenas sobre o item. Dessa forma, deve-se sempre considerar a compensação entre os modelos como a complexidade computacional e a eficácia de melhoria de desempenho.

Para concluir, os sistemas de recomendações oferecem uma personalização nos serviços de streaming de animes. Eles também ajudam a resolver os problemas de sobrecarga de informação, oferecendo produtos revelevantes aos seus usuários. Este projeto buscou entender mais os sistemas de recomendações e discutir as quatro técnicas de recomendação mencionadas, destacando seus pontos fortes e fracos.

Para futuros trabalhos, seria interessante construir outros modelos utilizando outros métodos disponíveis no pacote "recommenderlab", como o modelo aleatório (RANDOM) e a regra de associação (AR), além dos modelos que utilizam o método de fatoração de matriz como a decomposição em valores singulares (SVD) e o fuzz SVD. Já para a filtragem baseada em conteúdo seria interessante experimentar outras abordagens como a de Redes Neurais (OORD; DIELEMAN; SCHRAUWEN, 2013), Árvores de Decisões (AMIN et al., 2018) e naive Bayses (SHUXIAN; SEN, 2019).

Referências

AMIN, M. M. et al. A decision tree based recommender system for backpackers accommodations. *International Journal of Engineering & Technology*, v. 7, n. 2.15, p. 45–48, 2018.

AZUIRSON, G. d. A. V. *Investigação da combinação de filtragem colaborativa e recomendação baseada em confiança através de medidas de esparsidade*. Dissertação (Mestrado) — Universidade Federal de Pernambuco, 2015.

BAATARJAV, E.-A.; PHITHAKKITNUKOON, S.; DANTU, R. Group recommendation system for facebook. In: SPRINGER. *OTM Confederated International Conferences"On the Move to Meaningful Internet Systems"*. [S.l.], 2008.

BARBOSA, C. E. M. Estudo de técnicas de filtragem híbrida em sistemas de recomendação de produtos. *Monografia. Centro de Informática, Ciência da Computação, UFPE*, 2014.

FREITAS, D. W. Recomendação de animes utilizando machine learning, uma abordagem baseada em avaliações dos usuários. *Engenharia da Computação*, 2018.

GIRSANG, A. et al. Collaborative recommendation system in users of anime films. In: IOP PUBLISHING. *Journal of Physics: Conference Series*. [S.l.], 2020. v. 1566, n. 1, p. 012057.

GORAKALA, S. K.; USUELLI, M. *Building a Recommendation System with R.* [S.l.]: Packt Publishing Ltd, 2015.

GOWER, S. Netflix prize and svd. 2014.

HAHSLER, M. *recommenderlab: Lab for Developing and Testing Recommender Algorithms*. [S.l.], 2021. R package version 0.2-7. Disponível em: ⟨https://github.com/mhahsler/recommenderlab⟩.

JIN, R.; SI, L. A study of methods for normalizing user ratings in collaborative filtering. In: *Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval*. [S.l.: s.n.], 2004. p. 568–569.

JOHNSON, R. A.; WICHERN, D. *Applied multivariate statistical analysis, 2002*. [S.l.]: Prentice Hall, 2007.

JUNIOR, S. et al. *Recomendação de conteúdo baseada em informações semânticas extraídas de bases de conhecimento*. Tese (Doutorado) — Universidade de São Paulo, 2017.

KODINARIYA, T. M.; MAKWANA, P. R. Review on determining number of cluster in k-means clustering. *International Journal*, v. 1, 2013.

LINDEN G., S. B. Y. J. Amazon.com recommendations: Item-to-item collaborative filtering. *IEEE Internet Computing*, 2003.

MACQUEEN, J. et al. Some methods for classification and analysis of multivariate observations. In: OAKLAND, CA, USA. *Proceedings of the fifth Berkeley symposium on mathematical statistics and probability*. [S.l.], 1967. v. 1.

MADATHIL, M. Music recommendation system spotify-collaborative filtering. *Reports in Computer Music. Aachen University, Germany*, 2017.

MEDEIROS, I. Estudo sobre sistemas de recomendação colaborativos. 30 f. *Trabalho de Conclusão de Curso (Graduação em Ciência da Computação)—Centro de Informática, Universidade Federal de Pernambuco, Recife*, 2013.

OORD, A. Van den; DIELEMAN, S.; SCHRAUWEN, B. Deep content-based music recommendation. *Advances in neural information processing systems*, v. 26, 2013.

PETERSON, L. E. K-nearest neighbor. *Scholarpedia*, 2009.

R Development Core Team. *R: A Language and Environment for Statistical Computing*. Vienna, Austria, 2009. ISBN 3-900051-07-0. Disponível em: ⟨http://www.R-project.org⟩.

SHUXIAN, L.; SEN, F. Design and implementation of movie recommendation system based on naive bayes. *Journal of Physics: Conference Series*, v. 1345, p. 042042, 11 2019.

SILVA, N. de C. Sistemas de recomendação não-personalizados para atrair usuários novos. Universidade Federal de Minas Gerais, 2018.

SILVA, R. G. N. e. *Sistema de Recomendação baseado em conteũdo textual: avaliação e comparação*. Dissertação (Mestrado) — UFBA e UEFS, 2014.

Anexo

A Anexo

Todos os scripts executados para a obtenção desses resultados estão disponíveis no endereço https://github.com/LarissaMoreno/TCC.