3
Sistemas de Recomendação baseados em Filtragem Colaborativa

Os **sistemas de recomendação** tornaram-se uma importante área de pesquisa desde o surgimento dos primeiros trabalhos acadêmicos sobre **filtragem colaborativa**, em meados da década de 1990 (Hill et al., 1995; Resnick et al., 1994; Shardanand & Maes, 1995). Desde então, um grande esforço tem sido empregado, tanto pela indústria quanto pela academia, no desenvolvimento de novas abordagens para sistemas de recomendação. O interesse por este assunto continua elevado não apenas porque ele constitui uma área de pesquisa rica em problemas, mas também devido à abundância de aplicações práticas que ajudam os usuários a lidar com a sobrecarga de informação, fornecendo recomendações personalizadas de conteúdo e serviços. Exemplos deste tipo de aplicação incluem a recomendação de livros, CD's e outros produtos na Amazon.com (Linden et al., 2003), de filmes na MovieLens (Miller et al., 2003), e notícias na VERSIFI Technologies (anteriormente AdaptiveInfo.com) (Billsus et al., 2002). Além disso, alguns fornecedores de *software* têm incorporado funcionalidades de recomendação em seus servidores de comércio (Peddy & Armentrout, 2003).

Adomavicius & Tuzhilin (2005) argumentam que, apesar de as raízes dos sistemas de recomendação poderem ser relacionadas a trabalhos em ciência cognitiva (Rich, 1979), teoria da aproximação (Powell, 1981), Recuperação de Informação (Salton, 1989), teoria da previsão (Armstrong, 2001), ciência da gestão (Murthi & Sarkar, 2003) e modelagem da escolha dos consumidores em *marketing* (Lilien et al., 1992), os sistemas de recomendação só emergiram como uma área independente quando pesquisadores começaram a focar problemas que se baseiam explicitamente na estrutura de **notas**. Na sua formulação mais comum, o problema de recomendação é reduzido ao problema de estimar notas para os itens que não foram vistos por um usuário. Intuitivamente, esta estimativa é geralmente baseada nas notas dadas por este usuário a outros itens e em algumas outras informações. Uma vez que podemos estimar notas para os itens ainda sem

avaliação, podemos recomendar ao usuário o(s) item(s) com as notas estimadas mais altas.

Segundo Adomavicius & Tuzhilin (2005), o problema de recomendação pode ser formulado como segue: seja C o conjunto de todos os usuários e P o conjunto de todos os possíveis itens que podem ser recomendados, tais como livros, filmes ou restaurantes. O espaço P de possíveis itens pode ser muito grande, chegando, em algumas aplicações, a centenas de milhares ou até mesmo a milhões de itens, tais como recomendações de livros ou CD's. Da mesma forma, o espaço de usuários também pode ser muito grande – milhões, em alguns casos. Seja r uma função que mede a utilidade do item p para o usuário c, ou seja, $r : C \times P \rightarrow R$, em que R é um conjunto totalmente ordenado – por exemplo, inteiros não negativos ou números reais dentro de certo intervalo. Então, para cada usuário c, queremos escolher o item $p \in P$ que maximiza a utilidade para o usuário, conforme a eq. 4:

$$\forall c \in C, p'_c = \arg\max_{p \in P} r(c, p) \tag{4}$$

Em sistemas de recomendação, a utilidade de um item é normalmente representada por uma nota, que indica o quanto um usuário específico gostou de um determinado item – por exemplo, José da Silva deu ao filme "Tropa de Elite" a nota 7 em 10. No entanto, tal como indicado anteriormente, a utilidade, em geral, pode ser uma função arbitrária, incluindo uma função de lucro. Dependendo da aplicação, a utilidade r pode ser especificada pelo usuário, como ocorre muitas vezes em avaliações definidas pelo usuário, ou calculada pela aplicação, como pode ser o caso de uma função de utilidade baseada no lucro.

O problema central de sistemas de recomendação reside no fato de que a utilidade r geralmente não é definida em todo o espaço $C \times P$, mas apenas em algum subconjunto de tal espaço. Isto significa que r precisa ser extrapolada para todo o espaço $C \times P$. Geralmente, a utilidade é inicialmente definida apenas para os itens previamente avaliados pelos usuários. Por exemplo, em uma aplicação de recomendação de filmes tal como a MovieLens.org, os usuários inicialmente avaliam um subconjunto de filmes que já viram. Um exemplo de uma matriz de avaliação usuário-item para uma aplicação de recomendação de filmes é

apresentado na Tabela 3, na qual as notas são especificadas na escala de 1 a 5. O símbolo ∅ no lugar de algumas notas na Tabela 3 significa que os usuários não avaliaram os filmes correspondentes. Portanto, o sistema de recomendação deve ser capaz de estimar as notas das combinações usuário-filme não avaliadas e emitir recomendações adequadas, baseadas nestas previsões.

Tabela 3: Matriz de notas usuário-item para uma aplicação de recomendação de filmes.

	K-PAX	Matrix	Titanic	A Era do Gelo
Alice	4	3	2	4
Vitor	∅	4	5	5
Renata	2	2	4	∅
Davi	3	∅	5	2

A extrapolação de notas desconhecidas, baseada nas notas conhecidas, geralmente é feita:

- Especificando heurísticas que definem a função de utilidade e validando empiricamente o seu desempenho; e
- Estimando a função de utilidade que otimiza um determinado critério de desempenho, tal como o erro quadrático médio.

Uma vez que as notas desconhecidas são estimadas, a recomendação de um item para um usuário é feita selecionando a nota mais elevada dentre todas as notas estimadas para esse usuário, de acordo com a eq. 4. De forma alternativa, podem-se recomendar os *N* melhores itens a um usuário ou um conjunto de usuários para um item.

As novas notas, para os itens que ainda não foram avaliados, podem ser estimadas de muitas maneiras diferentes, utilizando métodos de Aprendizado de Máquina, teoria da aproximação, e várias heurísticas. Os sistemas de recomendação geralmente são classificados nas seguintes categorias, com base na forma como são feitas as recomendações (Balabanovic & Shoham, 1997):

- Recomendações baseadas em conteúdo: serão recomendados itens similares aos que o usuário preferiu no passado;

- Recomendações baseadas em filtragem colaborativa: serão recomendados itens dos quais pessoas com gostos e preferências semelhantes gostaram no passado; e
- Abordagens híbridas: combinam métodos baseados em conteúdo e em filtragem colaborativa.

O escopo deste trabalho se limita apenas a sistemas de recomendação baseados em filtragem colaborativa, ou simplesmente sistemas de recomendação colaborativa. Tais sistemas tentam prever a utilidade de itens para um determinado usuário baseado nos itens previamente avaliados por outros usuários. Formalmente, a utilidade $r(c, p)$ de um item p para um usuário c é estimada com base nas utilidades $r(c_j, p)$ atribuídas ao item p por aqueles usuários $c_j \in C$, que são "semelhantes" ao usuário c. Por exemplo, a fim de recomendar filmes ao usuário c, um sistema de recomendação colaborativa tenta encontrar outros usuários que possuem gostos semelhantes em filmes, ou seja, avaliam os mesmos filmes de forma similar. Em seguida, apenas os filmes que esses "amigos" do usuário c mais gostam seriam recomendados. Uma vez que os sistemas de recomendação colaborativos utilizam as recomendações, ou seja, as notas de outros usuários, eles podem lidar com qualquer tipo de conteúdo e recomendar quaisquer itens, mesmo os que não são similares aos observados no passado.

A seguir, a seção 3.1 descreve como utilizar sistemas de recomendação colaborativos na resolução do problema de predição do CTR.

3.1. Recomendação Colaborativa para Predição do CTR

Nos sistemas de recomendação colaborativa, os usuários indicam, através de notas, o quanto eles gostam de determinados itens, e, a partir de notas conhecidas para alguns pares usuário-item, os sistemas de recomendação se propõem a prever qual nota um usuário daria para um item que ele ainda não avaliou. Na publicidade direcionada, uma tarefa muito importante é predizer o CTR de um anúncio para uma determinada palavra-chave. Dessa forma, é possível fazer um paralelo no qual uma palavra-chave representa um usuário, já que os interesses ou necessidades de informação do usuário estão expressos pela palavra-chave; um

anúncio representa um item, pois, de fato, oferece um produto ou serviço; e, finalmente, o CTR representa uma nota que um usuário daria para um item, já que indica o quanto o anúncio é relevante para a palavra-chave.

Portanto, podemos utilizar os sistemas de recomendação para resolver o problema de predição de CTR's desconhecidos com base em CTR's conhecidos. Para realizar essa predição, este trabalho recorre à fatoração de matrizes, conforme apresentado na seção 3.2.

3.2. Fatoração de Matrizes

A idéia central da técnica de **Fatoração de Matrizes** (**FM**) (Takacs, 2007) é bastante simples. Suponha que queremos aproximar uma matriz $X_{I\times J}$ como o produto de duas matrizes $A_{I\times K}$ e $B_{K\times J}$, conforme a eq. 5:

$$X \approx AB \tag{5}$$

Esse modelo multiplicativo com dois fatores é bastante adequado para aplicação em sistemas de recomendação, já que estes possuem duas entidades de interesse: usuários e itens. Dessa forma, os valores a_{ik} e b_{kj} podem ser considerados, respectivamente, os pesos do *k*-ésimo atributo latente para o *i*-ésimo usuário e o *j*-ésimo item.

Ainda no contexto de sistemas de recomendação, *X* tem muitos elementos desconhecidos, que não podem ser tratados como zero. Para este caso, a tarefa de aproximação de *X* pode ser definida como se segue: sejam $A \in \Re^{I \times K}$, $B \in \Re^{K \times J}$ e *D* o conjunto dos pares (*i*, *j*) cujo valor em *X* é conhecido. Se a_{ik} denotam os elementos de *A*, e b_{kj} os elementos de *B*, então:

$$\hat{x}_{ij} = \sum_{k=1}^{K} a_{ik} b_{kj} \tag{6}$$

$$e_{ij} = x_{ij} - \hat{x}_{ij}, (i, j) \in D \tag{7}$$

$$SE = \sum_{(i,j)\in D} e_{ij}^2 \tag{8}$$

$$(A, B) = \arg\min_{(A,B)} SE \tag{9}$$

Na eq. 6, $\hat{x}_{ij}$ denota como o usuário *i* avaliaria o item *j* segundo o modelo preditivo; na eq. 7, e_{ij} denota o erro de treinamento no (*i*, *j*)-ésimo exemplo; e na eq. 8, *SE* denota o erro quadrático total do treinamento. A eq. 9 afirma que o par (*A*, *B*) ótimo minimiza a soma do erro quadrático apenas para os elementos conhecidos de *X*.

A fim de minimizar *SE*, aplicamos um método simples de descida por gradiente para encontrar um mínimo local. Dados um exemplo de treinamento x_{ij} e sua aproximação $\hat{x}_{ij}$, calculamos o gradiente de e_{ij}^2 conforme a eq. 10 e a eq. 11:

$$\frac{\partial}{\partial a_{ik}} e_{ij}^2 = -2e_{ij} \cdot b_{kj} \tag{10}$$

$$\frac{\partial}{\partial b_{kj}} e_{ij}^2 = -2e_{ij} \cdot a_{ik} \tag{11}$$

Em seguida, para reduzir o erro e obter uma aproximação melhor de x_{ij}, podemos atualizar os pesos em *A* e *B* no sentido oposto do gradiente, conforme a eq. 12 e a eq. 13:

$$a'_{ik} = a_{ik} + \eta \cdot 2e_{ij} \cdot b_{kj} \tag{12}$$

$$b'_{kj} = b_{kj} + \eta \cdot 2e_{ij} \cdot a_{ik} \tag{13}$$

Na eq. 12 e na eq. 13, η é a taxa de aprendizado. Visando obter uma melhor generalização em exemplos que não foram vistos no treinamento, aplicamos uma regularização com o fator λ, evitando assim pesos excessivamente grandes, conforme a eq. 14 e a eq. 15:

$$a'_{ik} = a_{ik} + \eta \cdot \left(2e_{ij} \cdot b_{kj} - \lambda \cdot a_{ik}\right) \tag{14}$$

$$b'_{kj} = b_{kj} + \eta \cdot \left(2e_{ij} \cdot a_{ik} - \lambda \cdot b_{kj}\right) \tag{15}$$

A fim de que o aprendizado dos pesos correspondentes ao *k*-ésimo atributo latente seja concluído antes do início do aprendizado do (*k* + 1)-ésimo atributo, definimos o cálculo de uma predição $\hat{x}_{ij}$ parcial, com base em um modelo que utiliza apenas os $l \leq K$ primeiros atributos, conforme a eq. 16:

$$\hat{x}_{ij} = \sum_{k=1}^{l} a_{ik} b_{kj} \tag{16}$$

Portanto, o algoritmo de aprendizado pode ser resumido pelo pseudocódigo apresentado no Quadro 1:

Quadro 1: Pseudocódigo do algoritmo de aprendizado da Fatoração de Matrizes.

1.	Definir valores positivos pequenos para η e λ.
2.	Inicializar os pesos em A e B aleatoriamente.
3.	Para l de 1 até K:
4.	Para t de 1 até T:
5.	Para cada elemento x_{ij} conhecido de X:
6.	Calcular o $\hat{x}_{ij}$ parcial, conforme eq. 16.
7.	Calcular e_{ij} conforme eq. 7.
8.	Atualizar os pesos a_{il} e b_{lj} conforme a eq. 14 e a eq. 15.

Note que o número de atributos latentes a serem aprendidos para cada usuário e para cada item é definido através do parâmetro K, e que a quantidade de épocas de treinamento utilizadas para o aprendizado de cada um desses atributos é definida pelo parâmetro T. Dessa forma, é possível controlar o nível de generalização do modelo gerado pelo algoritmo, através desses dois parâmetros.

Observe também que o algoritmo de aprendizado é sensível à ordem em que os elementos conhecidos de X são processados pelo laço das linhas 5 a 8. Por esse motivo, é necessário que tal processamento seja executado em ordem aleatória, a fim de evitar que os últimos elementos processados sejam sempre os mesmos e exerçam maior influência sobre o modelo gerado.