Leonardo Filgueira

Sistemas de recomendação usando o software R

Niterói - RJ, Brasil

Universidade Federal Fluminense

Leonardo Filgueira

Sistemas de recomendação usando o software R

Trabalho de Conclusão de Curso

Monografia apresentada para obtenção do grau de Bacharel em Estatística pela Universidade Federal Fluminense.

Orientador: Prof. Luciane Ferreira Alcoforado

Niterói - RJ, Brasil

Universidade Federal Fluminense

Leonardo Filgueira

Sistemas de recomendação usando o software R

Monografia de Projeto Final de Graduação sob o título *"Sistemas de recomendação usando o software R"*, defendida por Leonardo Filgueira e aprovada em , na cidade de Niterói, no Estado do Rio de Janeiro, pela banca examinadora constituída pelos professores:

Profa. Dra. Luciane Ferreira Alcoforado
Departamento de Estatística – UFF

Prof. Dr. Steven Dutt Ross
UNIRIO

Prof. Dr. Nome do 2o membro da banca
Instituicao do 2o membro da banca

Niterói,

Sumário

Lista de Figuras

Lista de Tabelas

1 Introdução

A partir do aumento de informação disponível com a popularização da Internet e com a possibilidade de armazenar essas informações, surge o desafio de lidar com este grande conjunto de dados[1]. Este aumento de informações desafia o site, como lojas on-line, que recebe todas as informações dos usuários que visitam o endereço, mas também pode se tornar um problema para o usuário que, diante da grande quantidade de produtos disponíveis para compra, pode levar muito tempo para achar o produto desejado[2].

Sistemas de recomendação são técnicas de *machine learning* (aprendizado de máquina) que filtram um grande conjunto de dados, tendo como base informações dos usuários e itens[3]. A partir dessas técnicas são previstas as notas que os usuários dariam a determinados itens, que podem ser dos mais variados tipos, e, para um indivíduo, recomenda-se o(s) item(ns) que obtiveram uma nota prevista maior[4]. Os sistemas de recomendação têm como objetivo recomendar itens que interessariam aos usuários[5], beneficiando o usuário e a loja, pois eles aumentam o desempenho da loja, fazendo-a vender uma quantidade maior de produtos, e também facilitam a procura do usuário fazendo-o achar produto(s) desejados em um menor tempo[1].

O primeiro sistema de recomendação foi criado na década de 90 e tinha como nome "filtragem colaborativa", pois o sistema funcionava com base na colaboração entre os grupos de pessoas interessados. Contudo, o termo "sistemas de recomendação" é mais usado por ser mais geral, não sendo realizada, necessariamente, nenhuma colaboração entre pessoas[6]. Já em 1996 o *Yahoo* utilizou sistemas de recomendação em uma de suas páginas, aplicando em larga escala[6], coisa que hoje é feita comumente por diversos sites e serviços.

É facilmente perceptível no cotidiano o uso de sistemas de recomendação em ambientes on-line. Ao usar a *Netflix*, sugestões para o usuário são oferecidas, baseadas nas atrações já assistidas e/ou avaliadas. Sites de compras como a *Amazon* também oferecem sugestões de produtos ao usuário baseado em visitas à página dos produtos ou no comportamento de

outros usuários que compraram um mesmo produto. Também em redes sociais, como no *YouTube*, são sugeridos vídeos baseados no histórico do internauta e nas suas avaliações, ou então no *Facebook*, que recomenda lista de pessoas que o usuário pode conhecer[7].

Em geral, sistemas de recomendação utilizam como informação a avaliação (*rating*) dada pelos usuários aos itens, podendo a avaliação estar expressa de diferentes maneiras[4]:

- Avaliações numéricas: O usuário avalia um item numa escala numérica, como no site da *Amazon*, onde o usuário dá uma nota de até 5 estrelas.
- Avaliações qualitativas: A avaliação é dada por frases definidas, como: "Concordo totalmente", "Concordo parcialmente", ...
- Avaliações binárias: O usuário seleciona se gostou ou não gostou do item, como a *Netflix*, atualmente, recebe as avaliações.
- Avaliação unária: A indicação se refere a se o usuário visualizou, comprou ou então avaliou o item positivamente.

1.1 Técnicas de recomendação

Existem diferentes categorias de sistemas de recomendação, que podem ser classificados em: Filtragem baseada em conteúdo (*Content-based filtering*), filtragem colaborativa (*Collaborative filtering*) e sistemas de recomendação híbridos (*Hybrid Recommender Systems*)[5].

1.1.1 Filtragem baseada em conteúdo

Os sistemas nesta categoria recomendam itens similares aos que o usuário gostou no passado[7]. Para isto é necessário utilizar informações das características de um produto[4] e comparar com o perfil do usuário, de acordo com itens já conhecidos pelo usuário. Considerando filmes como itens, se um usuário avaliou positivamente filmes do gênero de ação, então o sistema recomendará a este usuário filmes de ação. Por outro lado, a filtragem baseada em conteúdo não leva em conta a similaridade de preferência entre os usuários, mas apenas o histórico do usuário e as características dos itens[7].

Algumas das técnicas utilizadas neste tipo de filtragem são: TF/IDF (*Term Frequency Inverse Document*), *naive Bayes Classifier*, árvores de decisão ou redes neurais[1].

1.1.2 Filtragem colaborativa

Na filtragem colaborativa são recomendados itens de acordo com as avaliações de todos os usuários[5]. Existem duas maneiras principais de realizar essa filtragem: baseado em memória ou em modelo[8]. Nos algoritmos baseados em memória, verifica-se a similaridade entre usuários ou entre itens (vizinhança), de acordo com suas avaliações passadas. Essa técnica é a mais utilizada para realizar recomendações[4]. Um exemplo simples seria: Se o usuário 1 comprou o item A, B e C, e o usuário 2 comprou os itens A e C, então recomenda-se o item B para o usuário 2.

Os algoritmos de filtragem colaborativa utilizam uma matriz, chamada de matriz de avaliações (*ratings matrix*), usualmente representada desta forma:

Tabela 1: Típica matriz **R** de avaliações

	Item 1	Item 2	$\cdots$	Item m
Usuário 1	$r_{(1,1)}$		$\cdots$	
Usuário 2		$r_{(2,2)}$	$\cdots$	$r_{(2,m)}$
$\vdots$	$\vdots$	$\vdots$	$\ddots$	$\vdots$
Usuário n			$\cdots$	$r_{(n,m)}$

Onde $r_{(i,j)}$ é a avaliação (*rating*) do usuário i dado ao item j. Em geral, os usuários não tiveram contato com todos os itens, então os itens não recebem avaliações de todos os usuários, produzindo então uma matriz esparsa (com grande quantidade de valores faltantes). Os algoritmos buscam, então, preencher a matriz de avaliações com previsões para os valores faltantes.

À medida, porém, que os números de usuários e items aumentam, podem surgir problemas ao realizar a filtragem, como o aumento do tempo necessário, além de recursos computacionais, para executar o algoritmo, chamado de problema de escalabilidade[8]. Além disso, existe o problema da esparsidade, pois um usuário, em geral, não avaliou uma grande quantidade de itens, mas apenas uma pequena quantidade, o que pode causar a impossibilidade do cálculo de medidas de similaridade (pois itens precisam ter sido avaliados por dois usuários), ou então pode levar, pela pequena quantidade de informação utilizada no cálculo da medida, a uma medida que não represente bem a real similaridade entre os usuários[8].

Buscando reduzir o tempo de processamento e melhores medidas de acurácia podem ser utilizados métodos de agrupamento (cluster)[9]. Uma possibilidade é agrupar usuários, de acordo com alguma informação disponível em k clusters e, para cada um dos grupos

de usuário, aplicar a técnica de recomendação.

1.1.3 Sistemas de recomendação híbridos

Os sistemas híbridos são uma combinação da filtragem baseada em conteúdo e filtragem colaborativa, buscando aproveitar as vantagens e eliminar as desvantagens das técnicas[4]. Cada uma das técnicas podem ser aplicadas de maneira separada, combinando os resultados, mas também pode ser construído um modelo com as duas abordagens unificadas[3].

2 Objetivos

Este trabalho tem os seguintes objetivos:

2.1 Objetivo geral

Comparar a acurácia das recomendações utilizando filtragem colaborativa para todo o conjunto de dados com as recomendações utilizando filtragem colaborativa para cada cluster de usuários.

2.2 Objetivos específicos

3 Materiais e Métodos

3.1 Conjunto de dados

Será utilizado um *dataset* disponível no site *grouplens*, disponível em `https://grouplens.org/datasets/movielens/1m/`. O conjunto de dados possui 1 000 209 avaliações de 3900 filmes dados por 6040 usuários[10], que se cadastraram no site *MovieLens* no ano de 2000. De acordo com o próprio site, pessoas podem se inscrever para avaliar filmes e receber recomendações de filmes para assistir.

Os usuários são representados pelo seu ID, que varia entre 1 e 6040 e os filmes possuem ID entre 1 e 3952. As avaliações têm formato numérico, de até 5 estrelas, com estrelas completas, tendo cada usuário avaliado ao menos 20 filmes. O conjunto de dados também apresenta o gênero dos filmes.

A base de dados será dividida em duas, treino e teste, na proporção de 70% para treinar o modelo e 30% que serão usados para que o modelo preveja as notas a fim de comparar com a nota real.

3.2 Metodologia

Haverão um conjunto de usuários $U = \{u_1, u_2, \ldots, u_n\}$ e um conjunto de itens $I = \{i_1, i_2, \ldots, i_m\}$, assim como as notas dos usuários aos itens, que serão armazenadas na matriz $\boldsymbol{R}_{n \times m}$ de avaliações[11]. Logo, cada linha da matriz $\boldsymbol{R}$ representa um usuário e cada coluna, um item. Os algoritmos buscarão preencher os valores faltantes desta matriz, com valores na mesma escala das avaliações presentes na matriz[3].

3.2.1 Filtragem colaborativa baseada no item (item-based)

Este algoritmo busca recomendar itens similares aos bem avaliados pelo usuário. Desta forma será verificado, para cada par de itens, a sua similaridade, e a partir desta medida

é prevista a avaliação do usuário para o item. A similaridade entre dois itens i e j pode ser medida pelo coeficiente de correlação de Pearson, definido da seguinte maneira[5]:

$$w_{i,j} = \frac{\sum_{u \in U} (r_{u,i} - \overline{r}_i)(r_{u,j} - \overline{r}_j)}{\sqrt{\sum_{u \in U} (r_{u,i} - \overline{r}_i)^2 \sum_{u \in U} (r_{u,j} - \overline{r}_j)^2}} \tag{3.1}$$

Sendo U o conjunto de usuários que avaliaram os dois itens, i e j,$r_{u,i}$ o rating dado pelo usuário u ao item i e $\overline{r}_i$ o rating médio recebido pelo item i dado por todos os usuários que o avaliaram.

Alternativamente, a similaridade entre os itens i e j pode ser medida considerando os ratings recebidos pelos dois itens como vetores e calcular o cosseno entre estes vetores[12]:

$$w_{i,j} = \cos(\vec{r_i}, \vec{r_j}) = \frac{\vec{r_i} \cdot \vec{r_j}}{\|\vec{r_i}\| \times \|\vec{r_i}\|} = \frac{\sum_{u=1}^{n} r_{u,i} r_{u,j}}{\sqrt{\sum_{u=1}^{n} r_{u,i}^2 \sum_{u=1}^{n} r_{u,j}^2}} \tag{3.2}$$

A seguir, o *rating* do item i pelo usuário a pode ser previsto da seguinte forma[5]:

$$p_{a,i} = \frac{\sum_{j \in k} r_{a,i} - w_{i,j}}{\sum_{j \in k} |w_{i,j}|} \tag{3.3}$$

Sendo k o conjunto de itens avaliados pelo usuário a que são mais similares ao item i.

3.2.2 Filtragem colaborativa baseada no usuário (user-based)

Este algoritmo assume que usuários com preferência similar no passado terão preferências similares no futuro. Então os *ratings* não observados serão previstos a partir das avaliações de uma vizinhança e usuários com gostos similares[11]. São então encontrados os k vizinhos mais próximos de um usuário ou então todos os usuários que tenham pelo menos uma dada similaridade. O coeficiente de correlação de Pearson pode ser utilizado como medida de similaridade entre dois usuários a e u, definida da seguinte maneira[5]:

$$w_{a,u} = \frac{\sum_{i \in I} (r_{a,i} - \overline{r}_a)(r_{u,i} - \overline{r}_u)}{\sqrt{\sum_{i \in I} (r_{a,i} - \overline{r}_a)^2 \sum_{i \in I} (r_{u,i} - \overline{r}_u)^2}} \tag{3.4}$$

Sendo I o conjunto de itens avaliados pelos dois usuários, $r_{u,i}$ é o rating dado pelo usuário u ao item i e $\overline{r}_u$ é o rating médio do usuário u a todos os itens por ele avaliados.

Uma outra maneira de calcular a similaridade entre dois usuários é considerar os ratings de dois usuários como vetores num espaço m-dimensional, para, assim, encontrar

o cosseno do ângulo entre estes vetores[5]:

$$w_{a,u} = \cos(\vec{r}_a, \vec{r}_u) = \frac{\vec{r}_a \cdot \vec{r}_u}{\|\vec{r}_a\| \times \|\vec{r}_u\|} = \frac{\sum_{i=1}^{m} r_{a,i} r_{u,i}}{\sqrt{\sum_{i=1}^{m} r_{a,i}^2 \sum_{i=1}^{m} r_{u,i}^2}} \tag{3.5}$$

Por fim, a predição da nota dada ao item i pelo usuário a é dada por:

$$p_{a,i} = \overline{r}_a + \frac{\sum_{u \in k} (r_{u,i} - \overline{r}_u) w_{a,u}}{\sum_{u \in k} |w_{a,u}|} \tag{3.6}$$

Sendo k a vizinhança do usuário a.

3.2.3 *PAM* (Partitioning Around Medoids)

O algoritmo de agrupamento *PAM* é baseado na definição de *medoide*, que é o ponto com menor distância, em média, de todos os outros elementos do cluster. O algoritmo, para obter k clusters, é executado da seguinte maneira[13]:

1. Definir aleatoriamente k medoides.
2. Associar cada um dos elementos restantes a um cluster, sendo pertencente ao grupo de medoide mais próximo.
3. Calcular a dissimilaridade entre um elemento x_i e todos os outros do cluster, e a dissimilaridade entre o medoide e os outros elementos do cluster.
4. Caso a distância considerando x_i seja menor que a distância do medoide, passe a considerar x_i como medoide daquele cluster.
5. Repetir os passos 2 a 4 até não haver troca de medoides.

Uma desvantagem desse método é a ineficiência ao ser aplicado para um grande conjunto de dados[14].

3.2.4 *CLARA* (Clustering Large Applications)

Essa técnica foi proposta, em 1990, de forma a aplicar o PAM, solucionando o problema de escalabilidade, ao utilizar amostragem para a aplicação do PAM[14]. O método, então, seleciona aleatoriamente uma parte da base de dados e aplica o algoritmo PAM

nesta amostra. Em seguida é calculada a função de custo, que é uma média da similaridade entre os medoides e os outros elementos da base[15]. A função de custo é definida da seguinte maneira:

$$C(m, D) = \frac{\sum_{i=1}^{n} d(x_i, cl(m, x_i))}{n} \tag{3.7}$$

Onde:

- m são os medoides encontrados.
- $cl(m, x_i)$ é o medoide mais próximo de um ponto x_i.
- $d(x_i, cl(m, x_i))$ é uma medida de similaridade entre x_i e seu medoide mais próximo.
- n é o número de observações na base de dados D.

Todo o processo é repetido um número determinado de vezes e o resultado que obtiver menor função de custo é definido, então como o melhor e é retornado[15].

3.2.5 *K-Means*

K-means é uma técnica que particiona elementos em k clusters utilziando-se de centroides, que são os elementos representativos de cada cluster. Este método busca minimizar a soma das distâncias dos elementos de um mesmo cluster. Dados então, uma matriz D, de dimensão $m \times n$, e um número de clusters k, o algoritmo, então, procede da seguinte maneira[16]:

1. São escolhidos, aleatoriamente, k objetos de D como sendo os centroides.
2. Cada elemento $D_{i,}$ é associado ao centroide mais próximo, de acordo com a medida de distância adotada (neste caso, a distância Euclidiana).
3. Os centroides de cada um dos clusters são calculados.
4. Repetir os passos 2 e 3 até que não haja mudanças.

3.2.6 Medidas de erro

Para verificar a acurácia do sistema de recomendação, dado que a base utilizada foi dividida em base de treino e de teste, serão comparadas a avaliação prevista e a avaliação observada. Considerando $r_{i,j}$ a avaliação observada e $p_{i,j}$ a avaliação prevista pelo modelo do usuário i ao item j, as medidas utilizadas serão[7]:

3.2.6.1 Erro médio absoluto

O erro médio absoluto (EMA) se dá pela soma do módulo das diferenças.

$$EMA = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} |r_{i,j} - p_{i,j}| \tag{3.8}$$

3.2.6.2 Erro quadrático médio

O erro quadrático médio (EQM) é a soma das diferenças ao quadrado. Por este motivo, a unidade de medida muda, e a sua interpretação deve ser cautelosa.

$$EQM = \frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (r_{i,j} - p_{i,j})^2 \tag{3.9}$$

3.2.6.3 Raiz do erro quadrático médio

Ao calcular a raiz do EQM obtém-se um número na mesma unidade de medida dos dados.

$$REQM = \sqrt{\frac{1}{nm} \sum_{i=1}^{n} \sum_{j=1}^{m} (r_{i,j} - p_{i,j})^2} \tag{3.10}$$

4 Análise dos Resultados

5 Conclusão

Referências

[1] ISINKAYE, F.; FOLAJIMI, Y.; OJOKOH, B. Recommendation systems: Principles, methods and evaluation. *Egyptian Informatics Journal*, Elsevier, v. 16, n. 3, p. 261–273, 2015.

[2] MILD, A.; NATTER, M. Collaborative filtering or regression models for internet recommendation systems? *Journal of Targeting, Measurement and Analysis for marketing*, Springer, v. 10, n. 4, p. 304–313, 2002.

[3] TAKAHASHI, M. M.; JR, R. H. Estudo comparativo de algoritmos de recomendação. *USP. São Paulo*, 2015.

[4] SHAPIRA, B. et al. *Recommender systems handbook*. [S.l.]: Springer New York, 2011.

[5] MELVILLE, P.; SINDHWANI, V. Recommender systems. In: *Encyclopedia of machine learning*. [S.l.]: Springer, 2011. p. 829–838.

[6] REATEGUI, E. B.; CAZELLA, S. C. Sistemas de recomendaçao. In: *XXV Congresso da Sociedade Brasileira de Computação*. [S.l.: s.n.], 2005. p. 306–348.

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

[8] DAKHEL, G. M.; MAHDAVI, M. A new collaborative filtering algorithm using k-means clustering and neighbors' voting. In: IEEE. *Hybrid Intelligent Systems (HIS), 2011 11th International Conference on*. [S.l.], 2011. p. 179–184.

[9] O'CONNOR, M.; HERLOCKER, J. Clustering items for collaborative filtering. In: UC BERKELEY. *Proceedings of the ACM SIGIR workshop on recommender systems*. [S.l.], 1999. v. 128.

[10] HARPER, F. M.; KONSTAN, J. A. The movielens datasets: History and context. *Acm transactions on interactive intelligent systems (tiis)*, ACM, v. 5, n. 4, p. 19, 2016.

[11] HAHSLER, M. *recommenderlab: A framework for developing and testing recommendation algorithms*. [S.l.], 2015.

[12] SARWAR, B. et al. Item-based collaborative filtering recommendation algorithms. In: ACM. *Proceedings of the 10th international conference on World Wide Web*. [S.l.], 2001. p. 285–295.

[13] VALE, M. N. do. *Agrupamentos de dados: Avaliação de Métodos e Desenvolvimento de Aplicativo para Análise de Grupos*. Tese (Doutorado) — PUC-Rio, 2005.

[14] PARK, H.-S.; JUN, C.-H. A simple and fast algorithm for k-medoids clustering. *Expert systems with applications*, Elsevier, v. 36, n. 2, p. 3336–3341, 2009.

[15] BHAT, A. K-medoids clustering using partitioning around medoids for performing face recognition. *International Journal of Soft Computing, Mathematics and Control*, Citeseer, v. 3, n. 3, p. 1–12, 2014.

[16] MINING, W. I. D. Data mining: Concepts and techniques. *Morgan Kaufinann*, 2006.

ANEXO A – Título do primeiro anexo

ANEXO B – Título do segundo anexo