Knowledge-Based Systems 97 (2016) 188–202

Contents lists available at ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

A non negative matrix factorization for collaborative filtering recommender systems based on a Bayesian probabilistic model

Antonio Hernando[a,*], Jesús Bobadilla[a], Fernando Ortega[a]

[a] *Universidad Politécnica de Madrid, Carretera de Valencia km 7, 28031 Madrid, Spain*

ARTICLE INFO

Article history:
Received 12 April 2015
Revised 16 November 2015
Accepted 25 December 2015
Available online 6 January 2016

Keywords:
Recommender systems
Collaborative filtering
Matrix factorization
Graphical probabilistic models

ABSTRACT

In this paper we present a novel technique for predicting the tastes of users in recommender systems based on collaborative filtering. Our technique is based on factorizing the rating matrix into two non negative matrices whose components lie within the range [0, 1] with an understandable probabilistic meaning. Thanks to this decomposition we can accurately predict the ratings of users, find out some groups of users with the same tastes, as well as justify and understand the recommendations our technique provides.

1. Introduction

Recommender systems are systems able to provide personalized recommendations to users [22]. Recommender Systems have been used in different domains, such as music [23], television [24], books [25], e-learning [26] or e-commerce [27]. However, most research papers have focused on movie recommendations.

Recommender systems are usually classified into two main categories according to the input of these systems:

- Based on contents. Recommender systems of this kind require that the items are described by means of some features or a verbal description [2,4]. Besides, such recommender systems need that users inform about their preferences of the kind of items they like. This can be implicitly obtained by observing the kind of items users consume.
- Based on collaborative filtering. Such recommender systems use a rating matrix $\mathcal{M}$ in which each user u provides information about how much he likes some items (see Tables 1 and 9 for examples of rating matrices). The recommender system does not use information about features of users or of the items. In this way, recommender systems of this kind detect the taste of users through the ratings users have already made. In this paper we will focus on Recommender Systems based on collaborative filtering.

Recommender systems based on collaborative filtering can be classified on behalf of the kind of algorithm they use to predict the tastes of users.

- Based on Memory. Memory-based recommender systems basically use the K-Nearest Neighbor algorithm (K-NN algorithm) for predicting the tastes of users. These recommender systems involve very interesting features:
  - The algorithm is very intuitive, since it is based on the following idea: in order to recommend items to user u, the algorithm finds out the users with tastes similar to those of u, called the neighbors of u, and recommends the user u those items that some of these neighbors of u have already rated highly. Because of the intuitiveness of the algorithm, such recommender system are able to explain the recommendations they provide [17]. Indeed, the following property holds in the K-NN algorithm:

Proposition 1. *If the recommender system predicts that a user u will like the item i, then there are some users with the same taste as u who have very positively rated the item i.*

  - The quality of the predictions and recommendations can be quite good, provided that a suitable similarity function is used to measure how similar two users are on behalf of their tastes. Besides, we can measure the reliability of the predictions and recommendations [16].

Nevertheless, a major drawback of memory-based recommender systems is that the they do not use a scalable algorithm for predicting and recommending items. Therefore, these recommender systems are not suitable when dealing with a great

* Corresponding author. Tel.: +34 658537247.
E-mail addresses: antonio.hernando@upm.es, Ahernando@etsisi.upm.es (A. Hernando), jesus.bobadilla@upm.es (J. Bobadilla).

http://dx.doi.org/10.1016/j.knosys.2015.12.018
0950-7051/© 2015 Elsevier B.V. All rights reserved.

Table 1
First example of rating matrix.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}	I_{11}	I_{12}	I_{13}	I_{14}	I_{15}
U_1	5	5	5	5	5	•	•	•	•	•	•	•	•	•	•
U_2	5	5	5	5	5	•	•	•	•	•	•	•	•	•	•
U_3	5	5	•	5	5	•	•	•	•	•	•	•	•	•	•
U_4	•	•	•	•	•	5	5	5	5	5	•	•	•	•	•
U_5	•	•	•	•	•	5	5	5	5	5	•	•	•	•	•
U_6	•	•	•	•	•	5	5	5	5	5	•	•	•	•	•
U_7	•	•	•	•	•	•	•	•	•	•	5	5	5	5	5
U_8	•	•	•	•	•	•	•	•	•	•	5	5	5	5	5
U_9	•	•	•	•	•	•	•	•	•	•	5	5	5	5	5
U_{10}	5	5	5	5	5	•	•	•	•	•	•	•	•	•	•
U_{11}	•	•	•	•	•	5	5	5	5	5	•	•	•	•	•

number of items and users. In order to avoid this important drawback, we will here focus on model-based recommender systems.

- Based on models. Model-based recommender systems are based on the idea of using a model to predict the ratings that users make. Since such recommender systems involve scalable algorithms for recommending, they are very advisable when using large recommender systems with a great number of users or items. These recommender systems require a learning phase for finding out the user and item matrices before the recommender system starts. However, a model-based recommender system can predict the ratings of users very quickly, once the learning phase is finished.

  Among all models proposed, those providing better accuracy in the predictions are the one based on factoring the rating matrix into two matrices [6]: one related to items and another one related to users. This factorization technique is based on the idea of finding out unknown K latent factors that allow to predict the ratings of users by solving an optimization problem (the learning phase). Once the optimization problem is solved (the learning phase is through), each user u and each item i are respectively associated to vectors of K components $\vec{a}_u$ and $\vec{b}_i$ that are used to predict, $p_{u,\,i}$, the rating that user u would make on item i:

  $$p_{u,i} = \vec{a}_u \cdot \vec{b}_i$$

  Besides, this factorization technique provides significant improvement of the quality of predictions and recommendations over the ones based on memory.

  The paper [5] gives an interpretation of the optimization problem posed during the learning phase in probabilistic terms: they consider that vectors $\vec{a}_u$, $\vec{b}_i$ and the ratings $r_{u,\,i}$ follow a Gaussian distribution. However, the components of vectors $\vec{a}_u$, $\vec{b}_i$ are still very hard to understand: since their components can take arbitrary (even negative) values, they do not have any straightforward probabilistic interpretation. More sophisticated versions of this model have been proposed considering bias of user and items, and making use of information about the users' preferences [6]. However, all these models also suffer from an important drawback: they cannot justify the predictions the models make since the components of vectors $\vec{a}_u$, $\vec{b}_i$ are hard to understand. Besides, this technique does not fulfill Proposition 1 (we will see some examples of this in Section 3).

  Recently, [19] deals with a new probabilistic model based on the Poisson distribution, of interest for recommender systems with other kinds of input: in these, instead of considering that the input is a rating matrix where each user explicitly evaluates some items, a matrix is considered indicating how many times each user has consumed each item (without indicating whether users liked the item or not). In this model, vectors $\vec{a}_u$ and $\vec{b}_i$ are more easily understandable, since they only take positive values. However, since this model is not suitably designed for a rating matrix as input (here the input is a matrix of consumed items), the resulting recommender systems do not provide good predictions about the ratings of users (according to MAE).

In our paper we will present a novel technique for factoring the rating matrix, preserving the advantages of the classical factorization technique:

- Just like in classical matrix factorization, we also consider the existence of K latent factors explaining the ratings that users make. In this way, we associate a K dimensional vector $\vec{a}_u = (a_{u,1}, \ldots, a_{u,K})$ for each user u and a K dimensional vector $\vec{b}_i = (b_{1,i}, \ldots, b_{K,i})$ for each item i. However, as we will see next, the components of these vectors in our model present significant advantages over the ones in the classical matrix factorization:
- Since we present a model for making predictions, our technique is completely scalable.
- Our model provides good quality of predictions and recommendations. Indeed, as we will see, according to our results, the quality of recommendations can be considered still better as those made with the technique [5].

In addition to keeping the advantages of the classical matrix factorization, our technique provides additional advantages:

- While $a_{u,\,k}$ and $b_{k,\,i}$ may take arbitrary real values in the classical matrix factorization, in our model $a_{u,\,k}$ and $b_{k,\,i}$ are bound to lie within the range [0, 1] .
- Unlike classical matrix factorization, here the values $a_{u,\,k}$ and $b_{k,\,i}$ have a understandable probabilistic interpretation:
  - Latent factors represent groups of users who share the same tastes in our system. In this way, the parameter K indicates the number of such groups in our database.
  - The value $a_{u,\,k}$ represents the probability that user u belongs to the group k of users. In this way, the following holds:

    $$\sum_{i=1}^{K} a_{u,k} = 1$$

  - The value $b_{i,\,k}$ represents the probability that users in the group k like item i.
- In order to get accurate predictions in the recommender system, the parameter K is lower in our model than in the classical matrix factorization. Consequently, vectors $\vec{a}_u$ and $\vec{b}_i$ have a lower dimension in our model, involving more efficiency in memory as well as more efficiency when performing operations with these vectors (when performing the learning phase or when predicting if a user will like the item $p_{u,i} = \vec{a}_u \cdot \vec{b}_i$).