Information Sciences 376 (2017) 216–232

Contents lists available at ScienceDirect

Information Sciences

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

A probabilistic model for recommending to new cold-start non-registered users

Antonio Hernando[a,*], Jesús Bobadilla[a], Fernando Ortega[b], Abraham Gutiérrez[a]

[a] *Escuela Técnica Superior de Ingeniería de Sistemas Informáticos, Universidad Politécnica de Madrid, Madrid, Spain*
[b] *Big True Data Research, Madrid, Spain*

ARTICLE INFO

Article history:
Received 21 September 2015
Revised 23 September 2016
Accepted 1 October 2016
Available online 6 October 2016

Keywords:
Recommender systems
Collaborative filtering
Graphical probabilistic models

ABSTRACT

Recommender Systems are designed to provide recommendations to registered users. Non-registered users can be regarded as a particular case of the pure new user cold-start problem. Since non-registered users have neither created a profile account nor rated any item, recommender systems cannot know the tastes of non-registered users, and they typically provide these non-registered users with the average rating of each item. Nevertheless, non-registered users are an important proportion of users of many recommender systems. Therefore, more sophisticated ways of recommending to these non-registered users are wished. Here, we will propose to offer these non-registered users a natural inference model based on uncertainty rules that allows them to infer themselves their own recommendations. This is mathematically formalized by means of a probabilistic model that simulates the forward reasoning based on rules.

1. Introduction

Recommender Systems are intelligent systems designed to provide personal recommendations to registered users [7]. Recommender Systems have been used in different domains, such as music [27], television [2], books [31], e-learning [3] or e-commerce [15]. However, most research papers have focused on movie recommendations.

These systems recommend the items that are likely to appeal to each user. Therefore, the systems need to know the users' tastes in order to provide them with personal recommendations. In this paper, we will focus on recommender systems based on collaborative filtering: those that only use the ratings that registered users have made to discover their tastes. Recommender systems based on collaborative filtering can be classified according to the type of algorithm used to predict the tastes of users:

Memory-Based. Memory-based recommender systems basically use the K-Nearest Neighbor algorithm (K-NN algorithm) to predict the taste of users by means of different similarity measures [4–6,8,10–13,18].

Model-Based. A Model-based recommender system uses a model to predict the tastes of the registered users. The most successful models are those based on the idea of using K latent factors to detect their tastes [21,25,33,36,37].

In any case, these approaches provide personalized recommendations for users who have registered and have rated some items. These techniques cannot provide personalized recommendations for non-registered users as they have not provided

* Corresponding author.
E-mail addresses: ahernando@etsisi.upm.es (A. Hernando), jesus.bobadilla@upm.es (J. Bobadilla), fernando@bigtruedata.com (F. Ortega), abraham@etsisi.upm.es (A. Gutiérrez).

http://dx.doi.org/10.1016/j.ins.2016.10.009
0020-0255/© 2016 Elsevier Inc. All rights reserved.

any ratings. The problem of recommending to non-registered users can be regarded as a particular case of the new user cold-start problem. Cold-start problems have drawn the attention of many researchers in collaborative filtering. Typically, they involve two variants: the new item-cold start and the new user cold-start problem.

The new item cold-start problem. This problem occurs when a new item has been recorded in the system. Since no users registered in the system have rated this new item yet, it cannot be recommended to any user. This problem can be partially resolved by staff members of the system providing prior ratings to the new item.

The new user cold-start problem. This problem happens when a new user has been registered in the system. Like non-registered users, the system finds it very hard to provide personalized recommendations to this recently registered-user as they have yet to make any rating. In [39], the authors survey the different approaches to help the system to provide personalized recommendations to a recently registered user. Indeed, these approaches can be grouped in to the following (as we will see, all these approaches are aimed at registered users and they are not suitable for non-registered users):

- Many lines of research on the new user cold-start problem are indeed aimed at the *partial* new user cold-start problem: that is to say, they focus on registered users who have rated a few items. These lines of research require the user to have rated some items in order to refine the parameters of the technique used in the system (such as defining a specific similarity function in K-NN [1,9,29,34] for this kind of users) when dealing with these specific registered users. However, this line of research is not suitable for non-registered users since they have not rated any items yet.
- There is also research on the pure cold-start problem (users who have not rated any items). Here we can distinguish different strategies:
  - Making use of additional data sources. This approach is based on the idea of using other kinds of information other than ratings in case the user has rated no items, or only a small number. These sources of information are typically demographic data [28,41], social tags [22,43] or users' opinions [16,35], which are obtained from the users' profiles when registering. This approach may be very interesting for registered users who have been asked to use their profiles and associated accounts like Facebook or Twitter. However, this approach is not suitable for non-registered users, since this kind of information is obtained when users register in the system.
  - Designing an questionnaire for users who are registering [42]. This approach is based on designing a short questionnaire about some items that users must fill out when registering. In these questionnaires, users who are registering are required to answer questions about their tastes on some items. Since users usually find questionnaires boring, many researchers [17,40,44] have focused on finding out a minimal set of questions (about items) whose answers provide a great deal of information about the tastes of the registering user. In some ways, this approach is equivalent to ask registering users to rate some specific items so that the system indeed deals with the *partial* cold-start problem: once the user who is registering has been made to rate a small subset of items, the system is able to provide them with personalized recommendations. Although this approach does not require the use of other data sources, it is also aimed at registered users since these questionnaires are only presented to users who are registering.

In this way, although non-registered users can be regarded as a specific new user cold-star problem, the approaches suggested are aimed at registered users. Indeed, when real recommender systems face non-registered users (or partial cold-start users who have not rated enough), they typically resort to providing either the average rating of each item or a non-personalized list of the most popular items.

Since non-registered users in many domains often visit a recommender system, we believe that research on non-registered users is also of interest. In this way, this paper focuses on non-registered users presenting an alternative to the rating average or the list of the most popular items. Indeed, this paper proposes to offer non-registered users a natural inference model which allows them to infer the possible items they will probably like themselves. Non-registered users may infer their own recommendation from their tastes (which the system does not know but the non-registered user naturally does). As we have stated above, successful inference models used in recommender systems (such as restricted Boltzmann machines [36] or probabilistic matrix factorization [37]) cannot be used because they make use of latent factors (very hard to interpret) that make it difficult for a human to foresee the recommendations the system makes. Consequently, we need to find a natural and understandable inference model for non-registered users. We propose to show non-registered users a tree of items embodying a set of uncertainty rules that the non-registered users can use to informally infer their own recommendations. As we will see, the informal inference used for non-registered users is equivalent to finding a path in the tree. This inference reasoning is mathematically formalized as a probabilistic model so that the system may automatically calculate the probability that the non-registered user likes each item. Summing up, in this paper we propose a model aimed at non-registered users or partial cold-start users. The model proposed can be used in two ways:

- As a model allowing non-registered users to informally infer their own recommendations. Since the model proposed is visually represented by means of a tree embodying a set of uncertainty rules and facts, a non-registered user may informally use it to infer their own recommendations.
- As a model for formally inferring the recommendations. Non-registered users may interact with the tree by reporting which items they like or do not like. According to this interaction, the system automatically recalculates the probability

that the user likes each item. In order to achieve this, we propose a probabilistic model with which the system can automatically calculate the probability that the user likes each item. These probability values will be depicted by the size of the nodes in the tree.

The paper is structured in the following way. In Section 2 we discuss work related to the work presented here. In Section 3 we will present the first purpose of our model: the possibility that non-registered users may informally infer their own recommendations. In Section 4 we will present the second purpose of our model: the possibility that the system may formally and automatically calculate the probability that a non-registered user likes an item when this user interacts with the tree. In Section 5 we show that the informal reasoning used for non-registered users (Section 3) can be interpreted by means of the formal equations used for calculating the probabilities that the user likes the items (Section 4). In Section 6 we analyze the efficiency of our model when predicting the tastes of users. In Section 7 we will evaluate our model. Finally, in Section 8 we set our conclusions. In addition, we have included two appendices related to the mathematical details of our model.

2. Related work

In this paper we propose a probabilistic model that allows us to model the uncertainty reasoning based on rules and facts. This model will be shown to non-registered users so that they may infer their own recommendations themselves.

As stated above, the problem of recommending to non-registered users can be regarded as a particular case of the new user cold-start problem. However, so far the proposed approaches for solving this problem in collaborative filtering have been designed for registered users. These approaches consider that the system has information of the registered users (like demographic information or the answers to a set of questionnaires) that is obtained when they register. On the contrary, our approach is aimed at non-registered users (the system does not have any kind of information on the user).

Many successful techniques used in recommender systems are based on probabilistic models [36,37]. Although these techniques provide highly accurate predictions of the users' tastes, they make use of the idea of unobservable latent factors which may be very difficult to understand and cannot be used for a human to infer the recommendations the system will suggest. We propose here a probabilistic model without latent factors that provides a very natural inference model. Consequently, this model can be used by non-registered users to infer their own recommendations. This model is based on the visualization of a tree of items underlying a natural set of uncertainty rules and facts.

Trees have been proposed to be used to explain the recommendations the system suggests [19]. However, this use of trees is aimed at justifying the recommendations the system has made to a registered user: for each registered user, the system generates a specific tree with the recommendable and rated items displaying all the recommendations given to the registered user. Moreover, these trees do not involve any formal probabilistic model and they are only used as an intuitive way of convincing users about the recommendations suggested. Our model here is aimed at non-registered users: we will generate a tree with the whole set of items (or at least the most popular items) intrinsically providing a natural model so that non-registered users may infer their own recommendations. In addition, as we will see in Section 4, this model can be mathematically formalized using a probabilistic model.

Trees have also been used to visualize information on the recommender system. Visualizing information is an interesting field of Machine Learning. Multidimensional Scaling (MDS) [14] is a set of different techniques with this purpose: Principal Component Analysis (PCA) [23] (representing data through orthogonal transformations); Kernel PCA [38]; Self-Organizing Maps [24] based on neuronal networks; Spectral clustering [30]. In the recommender systems line of research, different tree-related techniques have been proposed [20,32]. All these techniques aim to represent data in two dimensions in such a way that the similarity between two data items is related to the distance of the points in the two-dimensional representation. Nevertheless, these techniques do not provide a formal model for inferring the recommendations, but only for visualizing items.

3. A model for *informally* inferring the recommendations

In this section, we will show the model proposed to non-registered users so that they can *informally* infer their own recommendations. In Section 3.1, we will show that the model proposed here is based on the natural model based on uncertainty rules and facts that the system displays to the non-registered user. These rules will be shown by means of a tree (see Section 3.2) so that non-registered users may easily infer their own recommendations.

3.1. Uncertainty rules

Since we constrain the inference model to being natural and completely understandable for humans, we propose a model based on the applications of a set of uncertainty rules and facts with the form:

- **General Facts.** Given an item, we will consider uncertainty facts with the form: "Users usually like item *i*.".
- **General Rules.** Given two items, *i* and *j*, we will consider uncertainty rules with the form: "If a user likes item *i*, this user will probably like item *j*."

- **Facts related to the non-registered user.** Non-registered users have a set of facts related to their tastes with the form:“I like item *i*.”.

A non-registered user can infer the recommendations the system would make by means of the previous uncertainty facts and rules described above. The main concern in this section lies in finding out these uncertainty general rules and facts. In this paper we propose the following:

- **General Facts.** For each item *i*, the recommender system considers a value b_i indicating the certainty of the general fact: “Users usually like item *i*”. The higher $|b_i|$, the less uncertainty in the fact. Indeed, when $b_i = 0$, we cannot say anything about the previous facts in relation to the item *i*. As we will see in Section 4.3, the value b_i may be learned in a initial learning phase of the model.
- **General Rules.** For each pair of items *i*, *j*, we consider a value w_{ij} indicating the certainty of the general rule: “If a user likes item *i*, this user will probably like item *j*” The higher w_{ij}, the less uncertainty in the rule. Like b_i, the value of the parameters w_{ij} may also be learned in a learning phase (see Section 4.3).

3.2. A model based on trees

In the previous section, we considered a model based on uncertainty general rules and facts that non-registered users may use to infer their own recommendations. Nevertheless, a simple model based on the textual description of these rules would have important disadvantages to be considered: The recommender system may offer a long list of uncertainty rules and facts, and the inference may not be simple for a human to handle; the model is not based on a visual representation that would be much more attractive for actual users instead.

In order to avoid these previous handicaps, we will consider a graph that implicitly describes the previous uncertainty rules and facts, and which takes into account the chain of reasoning described in the previous section:

Nodes. The nodes in the graph are the items of the recommender system. Each node *i* is labeled with the value b_i (described in Section 3.1) indicating the uncertainty of the general fact: “Users usually like item *i*.”

Edges. An edge in the graph between items *i* and *j*, means that the recommender system considers the general uncertainty rule: “if a user likes item *i*, this user will probably like item *j*.”. Each edge in the graph is labeled with the value w_{ij} (described in Section 3.1) indicating the uncertainty of the rule. In order to reduce the number of edges, we have only considered those edges whose label w_{ij} is greater than a given threshold.

A path in the graph from a rated item, i_1, to a recommended item, i_2, is related to the chain of reasoning used to infer whether the item i_2 is recommendable or not. One drawback of the approach just described is the need to draw graphs. It may become apparent that such graphs are not really helpful for reasoning in real recommender systems due to the following reasons: (i) these graphs may turn out to be obscure or misleading because they are not usually planar at all, but generally contain many edges; (ii) finding a path (representing a highly certain reasoning) from a rated item to a recommended item, may not be easy and quick, since the graph may be large and may contain many edges. To avoid these drawbacks, we propose considering, instead of the whole graph, its maximal spanning tree.

In order to provide useful information for the non-registered user when reasoning, we propose these trees to be drawn considering the following criteria:

- The size of each node *i* in the tree represents the probability (as we will see in Section 4.1) that the non-registered user likes item *i*. This is related to the uncertainty of the general fact: ‘Users usually like item *i*’.
- The width of each edge connecting items *i* and *j* represents the value w_{ij} related to the uncertainty of the general rule ‘If a user likes item *i*, this user will probably also like item *j*’. The wider the edge, the lower the uncertainty of this rule.

When a non-registered user visits the recommender system, the system shows this tree in such a way that the non-registered user may infer his own recommendations or determine if he likes or dislikes a specific item. Although the non-registered user may infer the recommendations by just looking at the tree, the system also provides the non-registered user with the possibility of interacting with the tree in order to help him with the reasoning. Indeed, the non-registered user may notify that he likes or dislikes an item so that the system automatically recalculates the size of nodes. In this way, the non-registered user can immediately visualize which items he will probably like (according to the inference model provided).

3.2.1. Example

Now, we will consider a small example to clarify the concepts explained. Let us consider a recommender system (see Table 1) with 17 users, $\{U_1, \ldots, U_{17}\}$ and 10 items, $\{I_1, \ldots, I_{10}\}$. We will consider that users can rate items in a scale from 1 to 5 (where 1 indicates that the user does not like the item at all, and 5 indicates that he likes it very much).

Although the values b_i and w_{ij} can be obtained in a learning phase, in order to clarify the ideas, we propose calculating them as:

- $b_i = m_i - 0.5$, where m_i is the mean of the ratings that registered users have made on item *i* if there is a significant number of users having rated this item *i*. Otherwise, $m_i = 0.5$. In Table 2 we show the values b_i associated to the uncertainty general facts.

Table 1
Ratings.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
U_1	4		5	5	5		4	5		4
U_2	5			5	5			5		4
U_3	2	2	1	1	1	2	1		5	
U_4		5		4	5			5		5
U_5					5					1
U_6					5					1
U_7					5					
U_8								1		
U_9								1		1
U_{10}		1	1	2	2	1		1		2
U_{11}						1		5	5	5
U_{12}					1				1	5
U_{13}				5			1	3	3	3
U_{14}						5	1			
U_{15}					5					
U_{16}	1	1	5		5	4	5	4	5	5
U_{17}	1	5	5		2	2	2	2	2	1

Table 2
Value b_i of item i.

I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
−0.10	−0.05	0.10	0.17	0.21	−0.13	−0.17	0.05	0.13	0.02

Table 3
Value w_{ij} between items i and j.

	I_1	I_2	I_3	I_4	I_5	I_6	I_7	I_8	I_9	I_{10}
I_1	•	0.00	0.00	**0.94**	0.51	0.00	0.13	0.80	0.50	0.33
I_2	0.00	•	0.46	0.84	0.13	0.00	0.00	0.32	0.00	0.00
I_3	0.00	0.46	•	**0.97**	0.73	0.69	0.73	0.73	0.00	0.37
I_4	**0.94**	0.84	**0.97**	•	**0.97**	0.00	0.50	0.77	0.00	0.57
I_5	0.51	0.13	0.73	**0.97**	•	0.84	**0.97**	**0.96**	0.49	0.09
I_6	0.00	0.00	0.69	0.00	0.84	•	0.21	0.26	0.13	0.34
I_7	0.13	0.00	0.73	0.50	**0.97**	0.21	•	0.71	0.38	0.75
I_8	0.80	0.32	0.73	0.77	**0.96**	0.26	0.71	•	**0.95**	**0.90**
I_9	0.50	0.00	0.00	0.00	0.49	0.13	0.38	**0.95**	•	0.41
I_{10}	0.33	0.00	0.37	0.57	0.09	0.34	0.75	**0.90**	0.41	•

- $w_{ij} = \max\{0, \rho_{ij}\}$ where ρ_{ij} is the similarity measure based on Pearson correlation between items i and j. In Table 3, we show the values w_{ij} associated to uncertainty general rules.

Considering Table 3 and Table 2, we would have the graph described in Fig. 1 in which we have only considered edges with $w_{ij} \geq 0.7$.

As can be seen in this figure, the nodes represent the items $I_1, \ldots, I_{10}$. A path in the graph from an item, i, to an item, j, means a possible reasoning for recommending the item j. For example, the path I_5, I_4, I_1 implies the following chain of reasoning:

Fact: “I like item I_5.”
Rule: “if a user likes item I_5, this user will probably like item I_4.”
Rule: “if a user likes item I_1, this user will probably like item I_1.”
Conclusion: “I will probably like item I_1”

In Fig. 2, we show the maximum spanning tree of the graph depicted in Fig. 1. As can be seen, the edges with the highest uncertainty are removed from the graph: they are from item I_1 to I_8 (with $w_{1,8} = 0.80$; from I_3 to I_5 (with $w_{3,5} = 0.73$); from I_3 to I_7 (with $w_{3,7} = 0.73$); from I_3 to I_8 (with $w_{3,8} = 0.73$); and from I_7 to I_{10} (with $w_{7,10} = 0.75$). Nevertheless, the reasoning with the lowest uncertainty can still be used through the tree. Besides, as we can see, the user can find it more easily.

Through this tree, the non-registered user may easily infer their own recommendations. For example, if the non-registered user liked item i_4, he could infer immediately that he would probably like items I_3 and I_5.

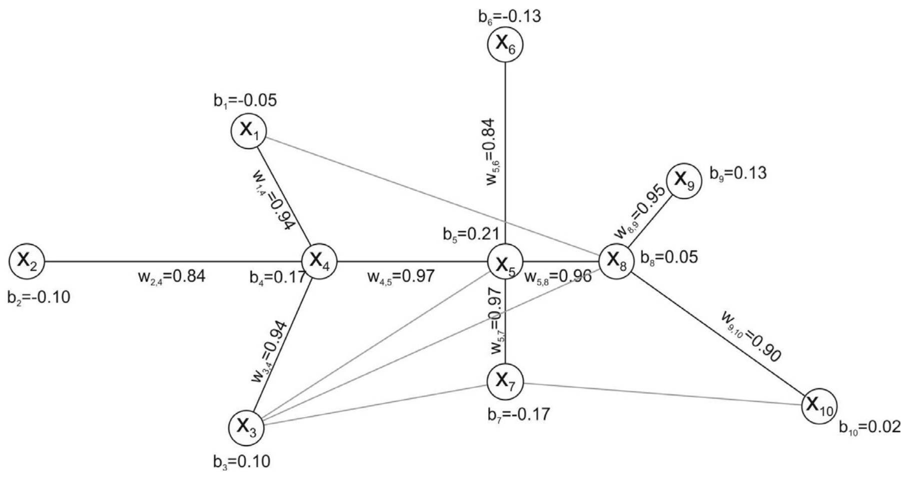

Fig. 1. Graph associated to the recommender system.

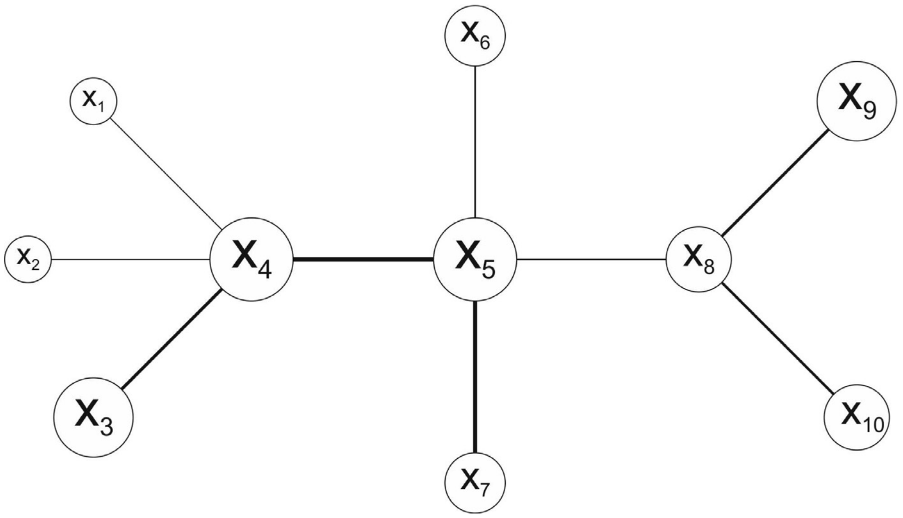

Fig. 2. Maximum spanning tree of the recommender system.

4. A model for *formally* and *automatically* inferring the recommendations

In Section 3, we described the model to be shown to the non-registered users so that they could informally infer their own recommendations. In this section, we will formalize this reasoning through a probabilistic model that allows the system to automatically calculate:

- The probability p_i that the non-registered user likes an item i.
- The probability p_{ij} that the non-registered user simultaneously either likes or dislikes the adjacent items in the tree i and j (this probabilistic value will be used in the learning phase).

As we have previously described, the tree is drawn according to these values:

- The size of a node represents the previous probabilistic value p_i. The higher p_i, the bigger the node drawn.
- The width of each edge between items i and j represents the uncertainty value w_{ij}. The lower w_{ij}, the wider the edge between items i and j drawn.

The uncertainty reasoning used for a non-registered user is modeled by means of a probabilistic model. This is a graphical model based on the tree obtained in Section 3.2.

For each item i, we will use a Bernoulli random variable x_i representing whether the user likes item i or not. In this paper, we will consider $x_i \in \{-1, +1\}$ indicating that the user either dislikes item i ($x_i = -1$) or the user likes item i ($x_i = +1$). We will use the notation $i \sim j$ to state that items i and j are connected in the tree obtained in Section 3.2.

Our probabilistic model is based on the following expression:

$$P(x_1,\dots,x_N)=\frac{1}{Z}e^{E(x_1,\dots,x_N)} \tag{1}$$

where

- $E(x_1,\dots,x_N)=\sum_{i\sim j}w_{ij}x_ix_j+\sum_i b_ix_i$.
- Z is a constant whose value is defined as:

$$Z=\sum_{x_1\in\{-1,1\}}\dots\sum_{x_N\in\{-1,1\}}e^{E(x_1,\dots,x_N)}$$

  As we will see in Section 4.1, the value of this constant Z does not need to be calculated to determine the probability p_i that the non-registered user likes item i.

As we stated above, our model allows the non-registered user to interact with the tree. In this way, non-registered users may inform about their tastes on some items. Consequently, we are interested in calculating the conditional probability that the non-registered user likes items knowing his tastes on some items $x_1^u\dots x_n^u$ (where $x_i^u\in\{-1,1\}$ indicates if the non-registered user u has notified whether he likes item i or not). We will use the notation $x_i^u=\bullet$ to indicate that the non-registered user u has not notified anything about item i. From (1), we can easily derive the expression for this conditional probability:

$$P(x_1,\dots,x_N|x_1^u\dots x_n^u)=We^{E(x_1,\dots,x_N)}\prod_i h_i(x_i) \tag{2}$$

where:

- W is another constant. Like Z, the value of W does not need to be calculated in order to determine the probability that the non-registered user likes item i knowing his tastes on some items $\{x_1^u\dots x_n^u\}$.
- The function $h_i(x_i)$ (where $x_i\in\{-1,+1\}$) takes into account the notifications the non-registered user u has given about his tastes:

$$h_i(x_i)=\begin{cases}0.5 & \text{if } x_i^u=\bullet\\ 1 & \text{if } x_i=x_i^u\\ 0 & \text{if } x_i\neq x_i^u\end{cases} \tag{3}$$

  As may be seen, this expression may be used when non-registered users have not initially interacted with the tree (for every item i we have $x_i^u=\bullet$). In this case, we have the value $h_i(x_i)=0.5$ for every x_i. If a non-registered user interacts with the tree, the value $h_i(x_i)$ is updated. Indeed, as we will see in the next section, the expression (2) will be used for calculating both conditional and non-conditional probabilities.

In Section 4.1 we will show how to calculate the values p_{ij} and p_i according to which the size of the nodes are drawn. These probabilistic values are efficiently calculated by means of the sum-product algorithm. Besides, a non-registered user may interact with the tree notifying whether he likes some of the items (or not) in order to facilitate his reasoning. In this case, the system would update the probabilistic values p_{ij} and p_i (and consequently the size of the nodes) as if this user were reasoning. In Section 4.2, we will describe an algorithm for efficiently updating the probabilistic values p_i and p_{ij} when non-registered users interact with the tree. In Section 4.3, we will show an algorithm (a stochastic gradient algorithm) for learning the parameters w_{ij} and b_i of the probabilistic model.

4.1. Inference in the model

In this section we will use the sum-product algorithm [26] to infer the recommendable items for a non-registered user. The sum-product algorithm is an optimized algorithm for calculating marginal probability distribution in a probabilistic tree-based graphical model. Indeed, this same algorithm has been known under different names in different domains: for example it is known as the forward-backward algorithm in Hidden Markov Chains, or as Belief Propagation in other domains.

Determining the exact marginal probabilistic distribution in a graphical probabilistic model is an NP problem. However, when the factor graph of the probabilistic model is a tree (our case), the marginal distributions can be efficiently calculated in O(N) steps thanks to the sum-product algorithm for a system with N items.[1] If the factor graph is not a tree, the exact calculation of the marginal distribution cannot be efficiently calculated and different approaches have been proposed: the sum-product algorithm (known in this case as loopy belief propagation) may still be used although it is not guaranteed to stop; or techniques based on sampling from the probability distribution (Markov Chain Monte Carlo) or approximating the problem into another tractable problem (Variational Inference).

In our case, the structure of our probabilistic model is a tree, and therefore, we can efficiently calculate the following by means of the sum-product algorithm:

[1] Indeed, the sum-product algorithm stops in only 2A steps, where A is the number of edges in the tree.

- $p_i = P(x_i = 1)$: The probability that the non-registered user likes the item i.
- $p_{ij} = P(x_i = x_j)$: The probability that the non-registered user simultaneously either likes or dislikes the adjacent items i and j. As we will see in Section 4.3, this probability value will be used to learn the value of the parameters b_i and w_{ij}.

In order to calculate these probabilities values, we first need to define some auxiliary functions:

- For each pair of adjacent items $i \sim j$ in the tree, we define the following function (where $x_i, x_j \in \{-1, +1\}$):

$$g_{ij}(x_i, x_j) = \exp\{w_{ij}x_i x_j + b_i x_i\} \tag{4}$$

  This function $g_{ij}(x_i, x_j)$ can be interpreted as the contribution of the user's taste on item j for determining the 'belief' that the user likes the item i.
- For each pair of adjacent items $i \sim j$ in the tree, we recursively define the function $\mu_{i \to j}(x_j)$ (where $x_j \in \{-1, +1\}$):

$$\mu_{i\to j}(x_j) = \sum_{k\in\{-1,+1\}} g_{ij}(k, x_j) h_i(k) \prod_{s\neq j | s\sim i} \mu_{s\to i}(k) \tag{5}$$

  In the particular case that the item i is a leaf, then the previous formula becomes:

$$\mu_{i\to j}(x_j) = \sum_{k\in\{-1,+1\}} g_{ij}(k, x_j) h_i(k) \tag{6}$$

  As may be seen, the previous formula establishes the base case for calculating $\mu_{i \to j}(k)$. From these calculations, we can progressively apply formula (5) to calculate $\mu_{i \to j}(k)$ for each pair of adjacent items in the tree.
  The function $\mu_{i \to j}(k)$ can be interpreted as the contribution of the 'belief' that the user likes the item i to determine the 'belief' that this user likes the item j.
- For each item i, we define $\lambda_i(k)$ (where $k \in \{-1, +1\}$):

$$\lambda_i(k) = \exp(k \cdot b_i) h_i(k) \cdot \prod_{j | i\sim j} \mu_{j\to i}(k) \tag{7}$$

  The function $\lambda_i(k)$ can be interpreted as the 'belief' that the user likes the item i.
- For each pair of connected items i and j, we define $\lambda_{ij}(k_1, k_2)$ (where $k_1, k_2 \in \{-1, +1\}$):

$$\lambda_{ij}(k_1, k_2) = \frac{\lambda_i(k_1)\lambda_j(k_2)\exp(w_{ij}k_1k_2)}{\mu_{j\to i}(k_1)\mu_{i\to j}(k_2)} \tag{8}$$

  The function $\lambda_{i,j}(k_1, k_2)$ can be interpreted as the 'belief' that the user likes the item i and the item j.

Based on the previous equations, we can calculate p_i and p_{ij}:

- The probability that a non-registered user likes the item i:

$$p_i = \frac{\lambda_i(1)}{\lambda_i(-1) + \lambda_i(1)} \tag{9}$$

- The probability that a non-registered user simultaneously either likes or dislikes the adjacent items i and j.

$$p_{ij} = \frac{\lambda_{ij}(1,1) + \lambda_{ij}(-1,-1)}{\lambda_{ij}(1,1) + \lambda_{ij}(1,-1) + \lambda_{ij}(-1,1) + \lambda_{ij}(-1,-1)} \tag{10}$$

As may be seen, the calculations of the functions $\mu_{i \to j}(x)$ are the essence of the calculation of p_i and p_{ij}. These functions are calculated by means of a dynamic programing algorithm: Applying equation (6) where i is a leaf node in the tree; then we progressively apply equation (5) until all the values $\mu_{i \to j}(k)$ are calculated. This algorithm requires the application of the formula (5) $2N - 1$ times where N is the number of nodes in the tree. Once we have calculated these values $\mu_{i \to j}(k)$, the values p_i and p_{ij} are immediately calculated respectively using formulae (9) and (10).

4.2. Updating the values p_i *and* p_{ij}

Once the values p_i are calculated, the system shows the tree (drawn according to these values) to the non-registered user. Through this tree, the non-registered user may infer which items he will probably like. In order to make this inference easier, non-registered users may interact with the tree notifying their taste on some items. According to this interaction, the system automatically updates the values p_{ij} and p_i (and therefore, the size of the nodes on the tree) as if the user was reasoning with the tree shown.

As we expected, interactions with the tree do not require completely recalculating all the values $\mu_{i \to j}(k)$, because only a few values $\mu_{i \to j}(k)$ are changed (and consequently, only a few values of p_{ij} and p_i change with the interaction). Thanks to this, we do not need to recalculate all the values p_{ij} and p_i (associated to every item in the tree) through the algorithm described in Section 4.1.

Table 4
Parameter b_i associated with each item.

b_1	b_2	b_3	b_4	b_5	b_6	b_7	b_8	b_9	b_{10}
−0.22	−0.16	0.15	0.28	0.31	−0.09	−0.26	−0.10	0.23	0.11

Table 5
Parameter $w_{i,j}$ associated with each pair of adjacent items in the tree.

$w_{1,4}$	$w_{2,4}$	$w_{3,4}$	$w_{4,5}$	$w_{5,6}$	$w_{5,7}$	$w_{5,8}$	$w_{8,9}$	$w_{8,10}$
0.14	0.11	0.56	0.62	0.32	0.40	0.33	0.55	0.44

4.3. Learning the parameters

As we have discussed above, our model depends on some parameters w_{ij} and b_i reflecting the uncertainty of the general facts and rules. The value of these parameters can be learned in an initial phase through the ratings that registered users have made.

We have considered a stochastic gradient descent method through which the values of these parameters w_{ij} and b_i are calculated.

The algorithm iterates for each registered user u in the system running the following steps:

- We calculate the values p_i and p_{ij} (through the present values of the parameters w_{ij} and b_i) for a non-registered user who has not yet interacted with the tree: that is to say, we have that $h_i(x_i) = 0.5$. These probabilistic values are calculated according to Section 4.1.
- We read the ratings of the registered user u in interactions he has had with the tree. According to Eq. (3), the function h_i is redefined depending on whether the user u likes the item i or not. We determine whether the user u likes the item i by checking if the rating $r_{u,\, i}$ is above a threshold value, θ. In this way, we define:

$$h_i^\theta(x_i) = \begin{cases} 0.5 & \text{if } r_{u,i} = \bullet \\ x_i & \text{if } r_{u,i} > \theta \\ 1 - x_i & \text{if } r_{u,i} \leq \theta \end{cases} \tag{11}$$

  According to Section 4.2, we calculate the values p_i^θ and p_{ij}^θ for different specific values θ. Then we calculate:
  - The probability, q_i, that the user u likes the item i (according to the ratings that the user has made) as the mean of the values p_i^θ for the different values θ:

$$q_i = \frac{\sum_\theta p_i^\theta}{R} \tag{12}$$

  - The probability, q_{ij}, that the user u simultaneously either likes or dislikes the items i and j (according to the ratings that the user has made) as the mean of the values p_{ij}^θ for the different values θ:

$$q_{ij} = \frac{\sum_\theta p_{ij}^\theta}{R} \tag{13}$$

  where R is the number of possible values that we have considered in θ.
- We update parameters w'_{ij} and b'_i by means of the following formulae:
  - Update b'_i, the bias of the item i:

$$b'_i = b_i + \gamma \cdot (q_i - p_i - \lambda b_i) \tag{14}$$

  - Update w'_{ij}, the weight of the edge between items i and j:

$$w'_{ij} = w_{ij} + \gamma \cdot (q_{ij} - p_{ij} - \lambda w_{ij}) \tag{15}$$

  where λ is a regularized parameter and γ is the step parameter of the stochastic gradient descent algorithm.

4.4. Example

We can use the algorithm described in Section 4.3 to calculate the value of the parameters b_i and w_{ij}. In Tables 4 and 5 we respectively show the values of the parameters b_i and w_{ij} once the learning algorithm has finished.

According to Section 4.1, we can obtain the probability p_i for a non-registered user (see Table 6).

According to these probabilistic values we can draw the tree that will be shown to any non-registered user (see Fig. 2). As may be seen, the size of the nodes is drawn considering the values p_i.

Table 6
Probability p_i that a non-registered user likes an item.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
0.42	0.44	0.67	0.73	0.72	0.53	0.46	0.59	0.63	0.59

Table 7
Probability p_i that the non-registered user likes an item after notifying that he likes the item I_4.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
0.46	0.47	0.81	1.00	0.86	0.57	0.51	0.64	0.65	0.60

Table 8
Probability p_i that the non-registered user likes an item after notifying that he likes the item I_4 and dislikes the item I_9.

p_1	p_2	p_3	p_4	p_5	p_6	p_7	p_8	p_9	p_{10}
0.46	0.47	0.81	1.00	0.81	0.55	0.49	0.32	0.00	0.47

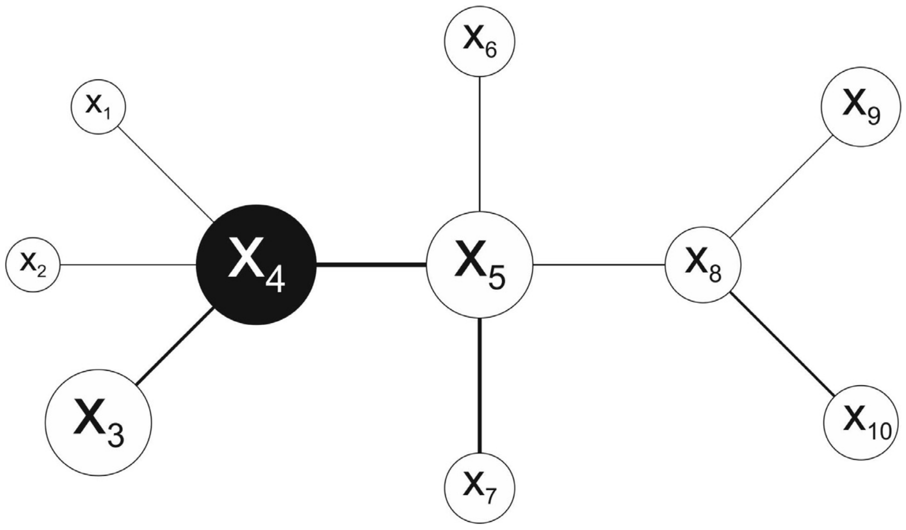

Fig. 3. Tree shown to the non-registered user after notifying that he likes the item I_4.

If the non-registered user likes the item I_4, he may immediately infer through this tree that he will also like the item I_5 (item I_5 is adjacent to I_4 with a wide line). In order to help the non-registered user with his reasoning, the system allows this user to interact with the tree. In this way, the non-registered user will draw the same conclusion if he notifies the system that he likes the item I_4. In this case, the probability values p_i would be recalculated through the algorithm described in Section 4.2. In Table 7 we show these updated probabilistic values. As can be seen, $p_4 = 1$ since this non-registered user has notified that he likes the item I_4.

The size of the nodes in the tree would be redrawn as shown in Fig. 3. Now, these nodes are bigger because the probability p_i is higher. Indeed, as we expected, the change in p_i is greater for the items closer to item I_4.

As we can see, these probability values p_i are intuitively related to the application of the implicit rules in the tree (corresponding to the reasoning of the non-registered user). For example, the probability that this non-registered user likes I_5 is very high (we can state that this user will like item I_5) because:

- The user likes the adjacent item I_4.
- Users usually like the item I_5.

If this non-registered user also notified that he dislikes the item x_9, the probability values p_i would be recalculated again (see Table 8). As may be seen, we have now that $p_9 = 0$ since this non-registered user has notified that he dislikes the item I_9.

The size of the nodes in the tree would be redrawn again as shown in Fig. 4.

As may be seen, these probability values p_i are intuitively related to the application of the implicit rules in the tree.

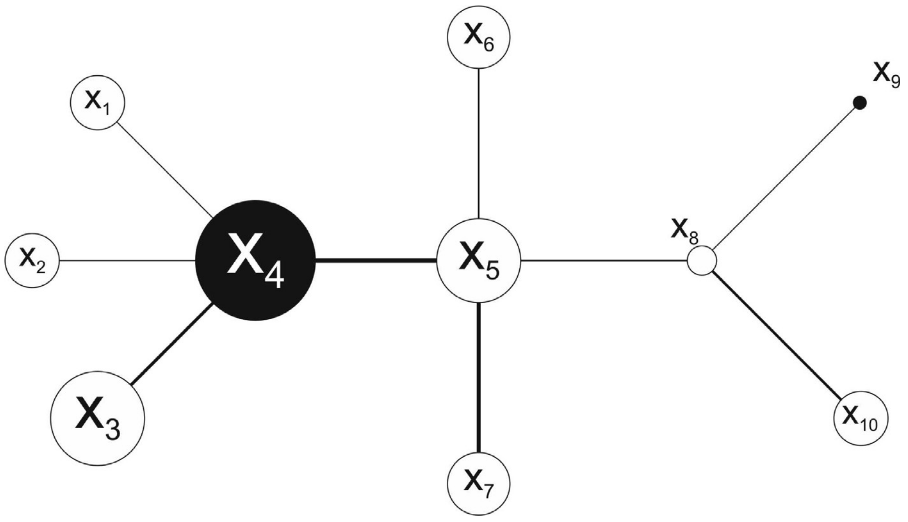

Fig. 4. Tree shown to the non-registered user after notifying that he likes the item I_4 and he dislikes the item I_9.

5. Interpretation of the probabilistic model

There is an interpretation of the equations described in Section 4.1 on behalf of the application of the uncertainty rules and facts described in Section 3.

According to the output tree obtained in Section 3.2, we may directly infer (by means of the uncertainty rules and facts described in Section 3) that the user likes the item i in three ways:

- By applying the personal fact: "The user u likes the item i".
- By applying the general fact: "Users usually like the item i.".
- By applying the rule: "If a user likes the item j, this user will probably like the item i.", where j is an adjacent item to i in the tree and we have previously inferred that the user likes the item j (by the same three ways we now indicate).

These three ways are directly reflected in the factors in the formula (7):

- $h_i(k)$ measures the belief that the user likes the item i by the application of the personal fact: "The user u likes the item i".
- $\exp\{k \cdot b_i\}$ measures the belief that the user likes the item i by the application of the general fact: "Users usually like the item i.".
- $\mu_{j \to i}(k)$ measures the belief that the user likes the item i by the application of the rule: "if a user likes the item j, this user will probably like the item i.".

All these measures are combined through formula (9) to get p_i, the probability that the non-registered user likes the item i.

6. Computational efficiency of the inference algorithm

The model proposed is completely scalable and runs very fast in practice. As we have seen above, our model involves two steps:

- Calculating the values p_i related to the tree that the system shows to every non-registered user (see Section 4.1). Once these values are calculated, they are stored and they do not need to be recalculated. Consequently, this step only needs to be calculated once. As we have stated above, this step involves a computational complexity $O(N)$ where N is the number of items in the system. For MovieLens 1M, this step takes less than 0.1 seconds on a very common computer (Intel i3 1Gb RAM).
- Updating the values p_i when the non-registered user interacts with the tree (see Section 4.2). This step is even faster than the previous one, since only a few values of p_i are updated: when a user notifies something with the item i, only the value p_j of the items j close to i are updated. Like the previous step, for MovieLens 1M, this step takes lower time than 0.1 seconds on a very common computer (Intel i3 1Gb RAM).

7. Evaluation

In this section we will quantitatively and qualitatively evaluate our approach. We have analyzed our approach from the following points of view:

- A visual model so that non-registered users may infer their own recommendations. This is the main purpose of our model. As we will see, our model provides important advantages over the previous approaches proposed so far.
- A probabilistic model for cold-start users: registered users who have rated few items. As we will see, our model may also be useful for registered users who have rated few items.
- A probabilistic model for providing recommendations to ordinary registered users. Although our model is designed for non-registered users, we find it interesting to evaluate our model with other accurate techniques in order to have a reference.

7.1. Qualitative evaluation

7.1.1. Comparing our approach with those aimed at non-registered users

The model we have proposed is specially aimed at non-registered users. Non-registered users may be considered as new users in the system who have rated no items. In this way, making recommendations to non-registered users may be regarded as a specific case of the pure new user cold-start problem, which has been thoroughly studied in collaborative filtering [39]. Different approaches in collaborative filtering have been proposed to provide personal recommendations to users who have not yet rated any items (like non-registered users). These approaches can be grouped into the following categories[2]:

- Those approaches that consider other sources of information different from the ratings [35,41,43]. These approaches use the registered user's profile (demographic information or accounts like Facebook or Twitter) to find out the user's tastes. Our approach is aimed at non-registered users for which the system does not have this kind of information. Therefore, this approach is not suitable for the non-registered users we are considering.
- Those approaches whose purpose is to design a questionnaire that users must answer when they register. In this questionnaire, the user (who is registering) is forced to answer about his tastes regarding some items. Since users often find these questionnaires boring when they register, this approach has the purpose to find out a small set of items whose answers potentially provide a great deal of information about the users' tastes. Once the user has finished the questionnaire, we can deal with the problem as a partial new user cold-start problem (since the user has rated some items in the questionnaire). As in the previous case, we think that this approach is not suitable for non-registered users, since these questionnaires are obviously designed for users who are registering. Nevertheless we have considered them in order to quantitatively compare our approach (see Section 7.2).

Since previous approaches are aimed at registered users, systems usually provide the non-registered users with the average rating or ranking of the most popular items in the system. Our approach is aimed at providing some other kind of information to these non-registered users. As we have stated above, we propose to provide a completely understandable model which allows the non-registered users to infer their own recommendations. From a qualitative point of view, our model provides the following advantages:

- Like average rating, our model also shows the popularity of an item through the size of the nodes in the tree (formally, the probability p_i that a user likes the item i). Indeed, the popularity of an item is represented by the uncertainty of the general fact: "Users usually like the item i."
- In addition our model also shows the relation between two items by means of rules with the form: "If a user likes the item i, then this user will probably like the item j.".
- Our model allows non-registered users to interact with the tree. When a non-registered user notifies that he likes or dislikes an item, the system automatically changes the size of the nodes (representing the probability that the non-registered user likes an item) as if this same user was reasoning.

7.1.2. Comparing our approach with those aimed at cold-start registered users

Researchers on collaborative filtering have thoroughly studied the *partial* new user cold-start problem: registered users who have not rated enough items so that the system may accurately and reliably predict their tastes. They try to propose specific techniques (defining specific similarity functions in K-NN [1,9] for these cold-start registered users) in order to improve the system's accuracy.

Although our model is specially designed for non-registered users, it may also be used for these cold-start registered users. Indeed, our approach provides even better accuracy (see Section 7.2) than these techniques proposed. In addition our technique also provides qualitative advantages:

- Our model allows registered cold-start users to infer other recommendations (different from the ones that the system provides) through their tastes.
- Our model allows registered cold-start users to contrast the recommendations that the system provides. Since cold-start registered users have not rated enough items, the system is not able to provide them with reliable predictions. Therefore, it is very important that cold-start registered users may contrast the recommendations provided.

[2] Many lines of research of the new user cold-start problem have indeed aimed at the *partial* new user cold-start problem: that is to say, they focus on registered users who have rated only a few items. However, these lines of research are not suitable for non-registered users since these users have not rated any items in the system (recommender systems only contain information of the registered users).

Table 9
RMSE of different techniques designed for non-registered users.

Approach	MovieLens 1M	NetFlix
questionnaire in [44] (7 questions)	0.941	0.912
questionnaire in [17] (7 questions)	0.952	0.922
questionnaire in [40] (7 questions)	0.917	0.918
Our approach	0.867	0.864

7.1.3. Comparing our approach with those aimed at ordinary users

As we have stated above, the model proposed here is specially designed for non-registered users. Nevertheless, we find it interesting to compare our model with the accurate techniques used in collaborative filtering when dealing with ordinary registered users. We have designed a new probabilistic model, which presents some advantages:

- As we will see, our approach provides accuracy measures competitive to the ones obtained with the classical *K*-NN algorithms used in collaborative filtering (see Section 7.2).
- Unlike the *K*-NN algorithm, our model provides a scalable algorithm for inferring the taste of users (see Section 6).
- Unlike the successful models used in collaborative filtering (like [21,25,36,37]), our model allows the recommendations provided to be easily understood.
- Neither the successful models used in collaborative filtering (like [21,25,36,37]) nor the *K*-NN approach allow non-registered users to infer their own recommendations. On the contrary, ours allows the non-registered users to infer their own recommendations.

7.2. Quantitative evaluation

Here we will evaluate our model through quantitative measures.

We have used the databases MovieLens 1M and NetFlix to evaluate our technique because they are established references in the research literature and other approaches in cold-start have used them.

7.2.1. Comparing our approach with those aimed at non-registered users

Here, we will quantitatively evaluate our model for non-registered users. We compare our approach with the one based on designing a questionnaire when users are registering. In this way, we consider that we are forcing non-registered users to give their opinions about a small set of items when visiting the system. Although we think that this approach is not suitable (indeed this approach is not used in practice for non-registered users), we have considered it in order to quantitatively compare our model. Since we are strictly focusing on recommender systems based on collaborative filtering that do not use demographic or social information about non-registered users, we are not considering here the hybrid approaches based on different data sources (we are assuming that the recommender system only has information about ratings).

We have carried out the same experiment used to evaluate the questionnaires [17,40,44]. This is based on the following steps:

- We have divided the registered users of the database into two sets:
  - The test user set, T, composed of 25% of users in the test set. The ratings of these users are employed to measure and compare the accuracy of the different approaches considered.
  - The training user set, E, composed of 75% of users.
- We consider a user u from the test user set T. We consider that this user u has not been registered in the system (and therefore the system does not know his tastes).
- We group the items that the user u has rated into two groups:
  - T_1: This set is composed of 25% of the items that the user u has rated. These items are regarded as the items the user u wishes to know whether he would like or not.
  - T_2: This set is composed of 75% of the items that the user u has rated. These items are regarded as the items that the user u has consumed (and therefore he knows whether he likes them or not).
- We try to predict if the user u would like each item $i \in T_1$ through different methods:
  - We consider that the user u has been forced to complete the recent intelligent questionnaires designed in [17,40,44]. The user u answers the questionnaire according to the ratings T_2. Then the system predicts how much the user u would like the item i according to these answers in the questionnaire.
  - We consider that the system provides part of the tree around each item i. Fig. 5 depicts part of the tree calculated in E when the user u wishes to know whether he would like the film *"Aliens (1986)"* or not. We simulate the reasoning the user u would follow according to the ratings in T_2 (see Section 4.1).
- We compare the prediction of the previous item in T_1 with the real rating the user u has made through the RMSE quality measure (see Table 9). As may be seen, our approach clearly provides better accuracy values than questionnaires.

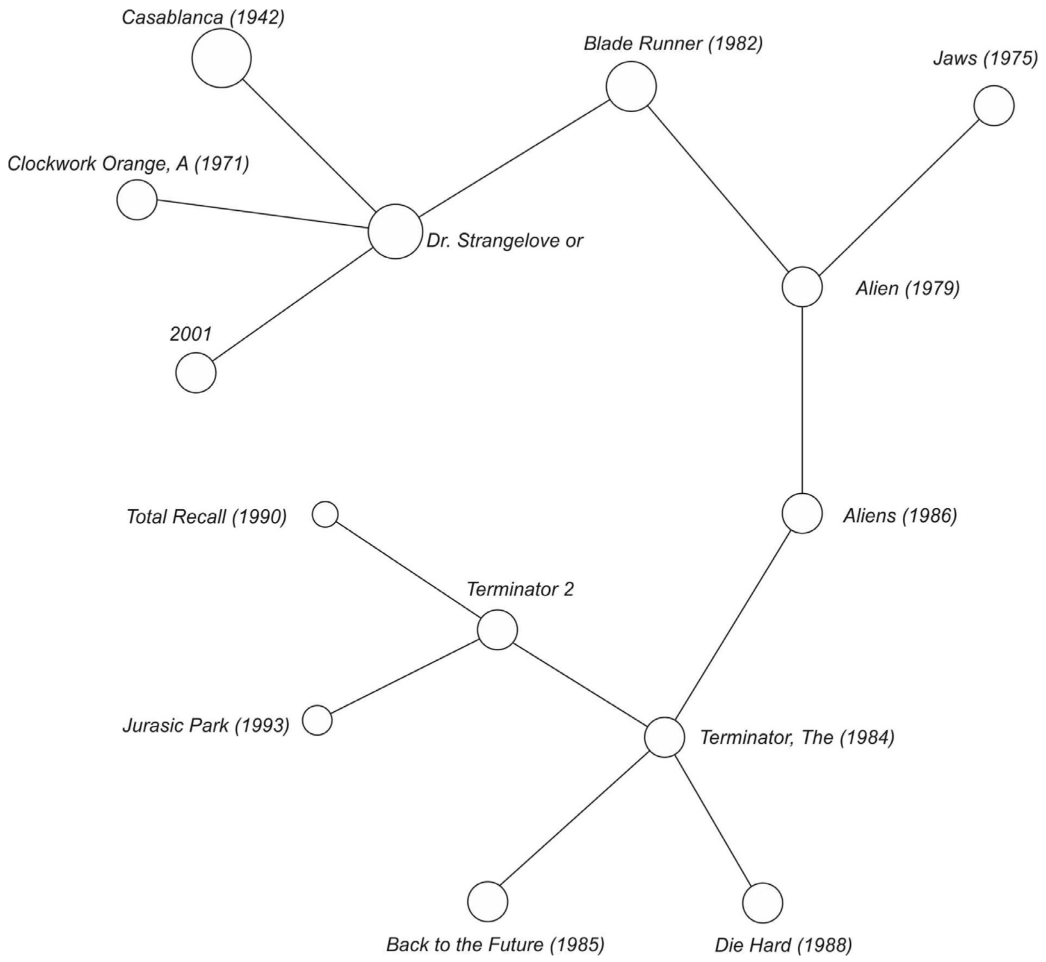

Fig. 5. Piece of the output tree obtained in MovieLens 1M shown to a non-registered users who wishes to know if he would like the film 'Aliens (1986)'.

7.2.2. Comparing our approach with those aimed at cold-start registered users

In this section, we will analyze our proposal as a model aimed at partial cold-start registered users: registered users who have only rated a small set of users. That is to say, we are considering that the system deals with registered users who have not rated enough so that the system may provide accurate predictions. Although, the model proposed is specially designed for non-registered users, we think that this may also be useful for cold-start registered users. In this section, we will analyze the predictions that our model may provide through the small set of ratings that a cold-start registered user has made. In this section, we consider that the registered users do not interact or infer their own recommendations through the tree, but that the system directly predicts them[3].

We have compared our proposal with different techniques designed for cold-start users. In this way, we have carried out a similar experiment to the one in the previous section (but considering that the test registered users have rated few items):

- We have divided the registered users of the database into two sets:
  - The test user set, T, randomly composed of 20% of users in the test set. The ratings of these users are employed to measure and compare the accuracy of the different approaches considered.
  - The training user set, E, randomly composed of 80% of users.
- We consider a user u from the test user set T. We consider that this user u has not been registered in the system (and therefore the system does not know his tastes).
- In order to mainly consider partial cold-start users, we group the items that the user u has rated into two groups:
  - T_1: This set is randomly composed of 90% of the items that the user u has rated. These items are regarded as the items the user u wishes to know whether he would like or not.

[3] Nevertheless, we think that the tree should also be shown to these cold-start registered users so that they can infer other recommendations (that the system cannot provide because they have not rated enough), and they may also understand and contrast the recommendations the system provides.

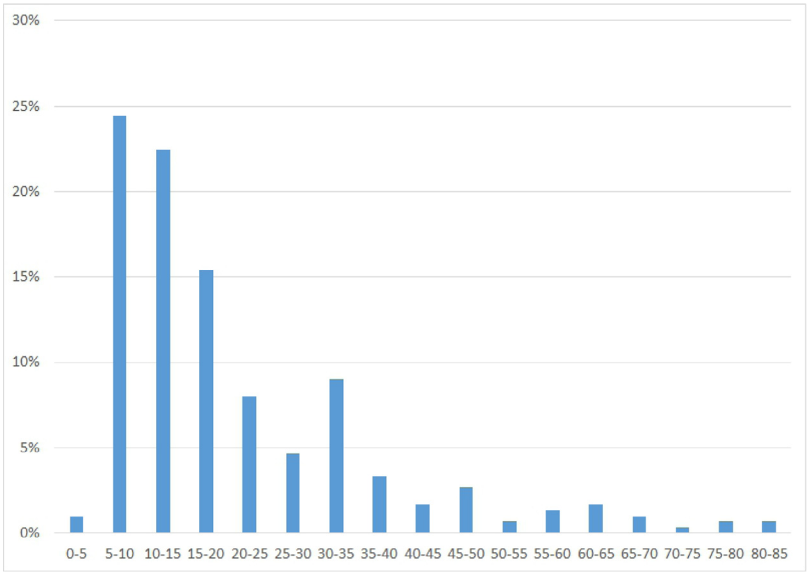

Fig. 6. Percentage of test users who have rated different amounts of test items.

Table 10
MAE of different techniques for cold-start users.

Approach	MovieLens 1M	Netflix
PIP [1]	0.812	0.818
MJD [9]	0.810	0.816
JMSD [12]	0.817	0.820
Sing [10]	0.819	0.822
Our approach	0.800	0.814

  – T_2: This set is randomly composed of 10% of the items that the user u has rated. These items are regarded as the items that the user u has consumed (and therefore he knows whether he likes them or not). As shown in Fig. 6, test users often rate very few items (between 5 and 10 items).
- We try to predict if the user u would like each item $i \in T_1$ through different methods:
  – Methods proposed in collaborative filtering for partial cold-start users.
  – Method based on the model proposed in Section 4.1.
- We compare the prediction of the previous item in T_1 with the real rating the user u has made through the MAE quality measure (this measure is used when dealing with cold-start users). As shown in Table 10, our approach provides slightly better accuracy than the approach based on defining similarity measures.

7.2.3. Comparing our approach with those used for ordinary registered users

Although our model is aimed at non-registered users, we find it interesting to analyze our model when dealing with ordinary registered users.

The experiment designed here is similar to the previous ones, but here we have considered that T_1 is composed of 80% of the ratings.

Here we have also considered the average rating used in recommender systems for non-registered users. Although this approach does not provide personalized recommendations, it is used in practice as a way to provide understandable recommendations for non-registered users. Advanced models with high accuracy (like [21,25]) cannot provide personalized recommendations when the user has rated no items either. Furthermore, although these models have very high accuracy, they are reduced to predict with the average rating when the user has not rated any item. In this way, average ratings can be considered as a reference measure for users (like non-registered users) who have not rated any items.

As shown in Table 11, the accuracy of our model is very competitive compared to the one based on the K-NN algorithm. Furthermore, unlike the K-NN algorithm, our approach provides a scalable method for determining the recommendations (see Section 6).

As expected, our proposed model (aimed at non-registered users) is not as accurate as the successful models based on matrix factorization when dealing with ordinary registered users. On the other hand, our approach provides the advantage over these accurate models [21,25,36,37] of providing understandable and completely justifiable recommendations (that is

Table 11
MAE of different techniques for ordinary users.

Approach	MovieLens 1M	Netflix
Average of the items	0.782	0.793
Pearson Correlation	0.762	0.773
JMSD [12]	0.746	0.752
Sing [10]	0.748	0.756
CJMSD [13]	0.751	0.759
PMF [37]	0.684	0.692
NMF [21]	0.664	0.666
Our approach	0.752	0.764

to say, users can understand the recommendations suggested), which is very desirable when dealing with non-registered users or partial new cold-start users. In this way, our model can be used to build a recommender system composed of two models: our proposed model for new cold-start non-registered users or partial cold-start users; and a traditional model based on matrix factorization for ordinary registered users.

8. Conclusions

In this paper we have presented a novel technique for non-registered users in a recommender system. With this technique we show non-registered users a very simple inference model that allows them to infer their own recommendations. This model has been mathematically formalized using a probabilistic model showing the uncertainty in a typically forward reasoning inference.

Acknowledgment

This research was supported by the Spanish Ministerio de Economía y Competitividad TIN2012-32682 project.

Supplementary material

Supplementary material associated with this article can be found, in the online version, at 10.1016/j.ins.2016.10.009

References

[1] H.J. Ahn, H.J., A new similarity measure for collaborative filtering to alleviate the new user cold-starting problem, Inf. Sci. 178 (1) (2008) 37–51.
[2] A.B. Barragáns-Martínez, E. Costa-Montenegro, J.C. Burguillo, M. Rey-López, F.A. Mikic-Fonte, A. Peleteiro, A hybrid content-based and item-based collaborative filtering approach to recommend TV programs enhanced with singular value decomposition, Inf. Sci. 180 (22) (2010) 4290–4311.
[3] J. Bobadilla, F. Serradilla, A. Hernando, Collaborative filtering adapted to recommender systems of e-learning, Knowl.-Based Syst. 22 (2009) 261–265.
[4] J. Bobadilla, F. Ortega, A. Hernando, J. Bobadilla, F. Ortega, A. Hernando, A collaborative filtering similarity measure based on singularities, Inf. Process. Manage. 48 (2) (2012a) 204–217.
[5] J. Bobadilla, A. Hernando, F. Ortega, A. Gutirrez, Collaborative filtering based on significances, Inf. Sci. 185 (1) (2012b) 1–17.
[6] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowl.-Based Syst. 23 (2010) 520–528.
[7] J. Bobadilla, F. Ortega, A. Hernando, A. Gutiérrez, Recommender systems survey, Knowl.-Based Syst. 46 (2013a) 109–132.
[8] J. Bobadilla, F. Ortega, A. Hernando, A. Arroyo, A balanced memory-based collaborative filtering similarity measure, Int. J. Intell. Syst. 27 (10) (2013b) 939–946.
[9] J. Bobadilla, F. Ortega, A. Hernando, J. Bernal, A collaborative filtering approach to mitigate the new user cold start problem, Knowl.-Based Syst. 26 (2011) 225–238.
[10] J. Bobadilla, F. Ortega, A. Hernando, A collaborative filtering similarity measure based on singularities, Inf. Process. Manage. 48 (2012a) 204–217.
[11] J. Bobadilla, A. Hernando, F. Orteqa, A. Gutirrez, Collaborative filtering based on significances, Inf. Sci. 185 (2012b) 1–17.
[12] J. Bobadilla, F. Serradilla, J. Bernal, A new collaborative filtering metric that improves the behavior of recommender systems, Knowl.-Based Syst. 23 (6) (2010) 520–528.
[13] J. Bobadilla, F. Ortega, A. Hernando, A. Arroyo, A balanced memory-based collaborative filtering similarity measure, Int. J. Intell. Syst. 27 (10) (2013) 939–946.
[14] I. Borg, P. Groenen, Modern Multidimensional Scaling: Theory and Applications, second ed., Springer-Verlag, 2005.
[15] J.J. Castro-Sanchez, R. Miguel, D. Vallejo, L.M. López-López, A highly adaptive recommender system based on fuzzy logic for b2c e-commerce portals, Expert Syst. Appl. 38 (3) (2011) 2441–2454.
[16] C. Chen, Y. Wan, M. Chung, Y. Sun, An effective recommendation method for cold start new users using trust and distrust networks, Inf. Sci. 224 (2013) 19–36.
[17] N. Golbandi, Y. Koren, R. Lempel, Adaptive Bootstrapping of Recommender Systems Using Decision Trees, WSDM, 2011.
[18] A. Hernando, J. Bobadilla, F. Ortega, J. Tejedor, Incorporating reliability measurements into the predictions of a recommender system, Inf. Sci. 218 (2013a) 1–16.
[19] A. Hernando, J. Bobadilla, F. Ortega, A. Gutirrez, Trees for explaining recommendations made through collaborative filtering, Inf. Sci. 218 (2013b) 1–17.
[20] A. Hernando, J. Bobadilla, F. Ortega, R. Moya, Hierarchical graph maps for visualization of collaborative recommender systems, J. Inf. Sci. 40 (1) (2014) 97–106.
[21] A. Hernando, J. Bobadilla, F. Ortega, A non negative matrix factorization for collaborative filtering recommender systems based on a bayesian probabilistic model, Knowl.-Based Syst. 97 (2016) 188–202.
[22] K. Ji, H. Shen, Addressing cold-start: Scalable recommendation with tags and keywords, Knowl.-Based Syst. 83 (2015) 42–50.
[23] I. Jolliffe, Principal Component Analysis, second ed., Springer, 2002.

[24] T. Kohonen, Self-organized formation of topologically correct feature maps, Biol. Cybern. (1982) 59–69.
[25] Y. Koren, R. Bell, C. Volinsky, Matrix factorization techniques for recommender systems, Computer 42 (8) (2009) 30–37.
[26] F.R. Kschischnang, B.J. Frey, H.A. Loeliger, Factor graphs and the sum-product algorithm, IEEE Trans. Inf. Theory 47 (2) (2001) 498–519.
[27] S.K. Lee, Y.H. Cho, S.H. Kim, Collaborative filtering with ordinal scale-based implicit ratings for mobile music recommendations, Inform. Sci. 180 (11) (2010) 2142–2155.
[28] B. Lika, K. Kolomvatsos, S. Hadjiefhymiades, Facing the cold start problem in recommender systems, Expert Syst. Appl. 41 (2014) 2065–2073.
[29] H. Liu, Z. Hu, A. Mian, H. Tian, X. Zhu, A new user similarity model to improve the accuracy of collaborative filtering, Knowl.-Based Syst. 56 (2014) 156–166.
[30] U.V. Luxburg, A tutorial on spectral clustering, Stat. Comput. 17 (4) (2007) 395–416.
[31] E.R. Núez-Valdéz, J.M. Cueva-Lovelle, O. Sanjuán-Martínez, V. García-Díaz, P. Ordoez, C.E. Montenegro-Marín, Implicit feedback techniques on recommender systems applied to electronic books, Comput. Hum. Behav. 28 (4) (2012) 1186–1193.
[32] F. Ortega, J. Bobadilla, A. Hernando, f. Rodríguez, Using hierarchical graph maps to explain collaborative filtering recommendations, Int. J. Intell. Syst. 29 (2014) 462–477.
[33] F. Ortega, A. Hernando, J. Bobadilla, J.H. Kang, Recommending items to group of users using matrix factorization based collaborative filtering, Inf. Sci. 345 (2016) 313–324.
[34] B.K. Patra, R. Launonen, V. Ollikainen, S. Nandi, A new similarity measure using bhattacharyya coefficient for collaborative filtering in sparse data, Knowl.-Based Syst. 82 (2015) 163—177.
[35] D. Poirier, F. Fessant, I. Tellier, Reducing the cold-start problem in content recommendation through opinion classification, in: Proceedings of the IEEE/WIC/ACM International Conference on Web Intelligence and Intelligent Agent Technology (WI-IAT 2010), vol. 1, 2010, pp. 204–207.
[36] R. Salakhutdinov, A. Mnih, G. Hinton, Restricted boltzmann machines for collaborative filtering, in: Proceedings of the 24th International Conference on Machine learning - ICML '07, 2007, p. 791.
[37] R. Salakhutdinov, A. Mnih, Probabilistic matrix factorization, in: Advances in Neural Information Processing Systems 20(NIPS07), 2008, pp. 1257–1264.
[38] B. Scholkopf, A. Smola, K. Müller, Kernel principal component analysis artificial neural networks, Neural Comput. (1997) 583–588.
[39] L.H. Son, Dealing with the new user cold-start problem in recommender systems: A comparative review, Inf. Syst. (2015).
[40] M. Sun, L. Fuxin, J. Lee, Learning multiple-question decision trees for cold-start recommendation, in: WSDM–13, Rome, Italy, 2013, pp. 4–8.
[41] M. Vozalis, K. G.Margaritis, Collaborative filtering enhanced by demographic correlation, in: Proceedings of the AIAI Symposium on Professional Practice in AI of the 18th World Computer Congress, 2004.
[42] X. Zhang, J. Cheng, S. Qiu, G. Zhu, H. Lu, DualDS: a dual discriminative rating elicitation framework for cold start recommendation, Knowl.-Based Syst. 73 (2015) 161–172.
[43] Z.K. Zhang, C. Liu, Y. C.Zhang, T. Zhou, Solving the cold-start problem in recommender systems with social tags, Europhys. Lett. 92 (2) (2010).
[44] K. Zhou, S.H. Yang, H. Zha, Functional matrix factorizations for cold-start recommendation, in: In: 34th International ACM SIGIR Conference on Research and Development in Information Retrieval, ACM, SIGIR '11, 2011, pp. 315–324.