J Intell Inf Syst (2015) 44:397–414
DOI 10.1007/s10844-014-0338-z

A novel hybrid approach improving effectiveness of recommender systems

G. M. L. Sarnè

Received: 22 October 2013 / Revised: 28 September 2014 / Accepted: 30 September 2014 /
Published online: 10 October 2014

Abstract Recommender systems support users by generating potentially interesting suggestions about relevant products and information. The increasing attention towards such tools is witnessed by both the great number of powerful and sophisticated recommender algorithms developed in recent years and their adoption in many popular Web platforms. However, performances of recommender systems can be affected by many critical issues as for instance, over-specialization, attribute selection and scalability. To mitigate some of such negative effects, a hybrid recommender system, called Relevance Based Recommender, is proposed in this paper. It exploits individual measures of perceived relevance computed by each user for each instance of interest and, to obtain a better precision, also by considering the analogous measures computed by the other users for the same instances. Some experiments show the advantages introduced by this recommender when generating potentially attractive suggestions.

Keywords Hybrid recommender system · Linear systems · Relevance measures

1 Introduction

The increasing amount of information available on the Web usually involves a lot of no relevant contents so that users have to spend a significant part of their navigation to search more interesting contents. A possible solution to this problem is represented by the use of recommender systems (Konstan and Riedl 2012). Over the years, they have evolved to provide users with more and more potentially useful suggestions about their interests and preferences.

To provide users with personalized suggestions, recommender systems need to collect a large amount of data on past users' interests and preferences to appropriately represent

G. M. L. Sarnè (✉)
DICEAM Department, University "Mediterranea" of Reggio Calabria, Loc. Feo di Vito,
89122 Reggio Calabria, Italy
e-mail: sarne@unirc.it

consolidated and emerging behaviors. In the detail, users' data can derive from the rates directly provided by them or automatically elicited by the system in monitoring their behaviors (Adomavicius and Tuzhilin 2001; Mobasher et al. 2002) and are usually represented by means of individual users' profiles.

For the adopted approach, recommender systems can be classified in the following categories (Burke 2007): (*i*) *Content-based* (CB) that considers past users' interests (Lops et al. 2011); (*ii*) *Collaborative Filtering* (CF), which realizes a knowledge-sharing environment to find people having similar interests for generating suggestions about items resulting unknown to the users (Breese et al. 1998; Su and Khoshgoftaar 2009); (*iii*) *Demographic*, to identify those recommender users belonging to the same demographic niche (Stiller et al. 2010); (*iv*) *Knowledge-based*, for inferring user's needs and references (Trewin 2000). A further category, commonly recognized as the most performing, is that of the hybrid systems (Burke 2002) which combine more approaches to promote mutual synergies and improve effectiveness and efficiency of the recommendation process. Interested reader can find a more comprehensive overview on the matter in Adomavicius and Tuzhilin (2005), Burke (2002), Konstan and Riedl (2012), Manouselis and Costopoulou (2007), Montaner et al. (2004) and Wei et al. (2007).

In particular, the main characteristic of a CB recommender is the evaluation of the user's interest in a potentially recommendable item only based on similarity measures computed between its features and those of the items stored in his/her profile (Adomavicius and Tuzhilin 2001), independently of the item popularity among other users. However, CB recommenders are unable to consider items unknown to the user, deal with a great number of item features and result to be inefficient in presence of newcomers. On the contrary, CF recommenders suggest an item only when it is popular among users (Shardanand and Maes 1995). CF recommenders are classifiable in (*i*) memory-based, generally effective and easy to implement (Hofmann 2004; Linden et al. 2003), that use information derived from some type of neighborhood-based process (Herlocker et al. 2002) to identify a suitable set of top-ranked items (Herlocker et al. 1999; Karypis 2001) or (*ii*) model based, that exploit data mining or machine learning techniques (i.e. Bayesian clustering (Miyahara and Pazzani 2000), latent semantic (Hofmann 2004), neural network (Postorino and Sarné 2011), etc.) or other approaches different from those adopted in the memory-based category. The need to compare users and items (Weng and Liu 2004) implies that CF computational costs are higher than for CB recommenders also for the usual presence of noising and sparse data. To perform more sophisticated prediction processes, such computational costs need to be released, for instance, by clustering users (Mobasher et al. 2002) and/or by pre-computing similarity measures off-line as well as recommendations (Rosaci and Sarnè 2010). Note that the latter strategy might introduce some mismatching because pre-computed suggestions may not consider the latest user's choices.

In recent years hybrid recommender systems are increased in popularity. In fact, they allow to overcome the weaknesses of a recommendation technique with the strengths of another, although their performances also depend on the way the different components are combined together (Burke 2007). Among hybrid recommender systems, a wide part of them is based on the combination of only CB and CF algorithms due to their high complementarity degree and the excellent performances in terms of effectiveness and efficiency (Burke 2002).

In this scenario a novel hybrid recommender, called Relevance Based Recommender (RBR) is presented. It adapts the same mathematical model proposed and applied in Buccafurri et al. (2004) and Rosaci et al. (2012) to different contexts. In particular, such a mathematical model considers several contributions that assume a different mean in RBR and are differently computed with respect to Buccafurri et al. (2004) and Rosaci et al. (2012). More in detail, RBR exploits the individual measures of the relevance degree of the instances (i.e. a products or information) computed for all the users belonging to a same community. In determining such individual relevance measures, for each instance RBR takes account of the existing interdependencies among the analogous measures of all the other users within the same community and this allows RBR to improve the precision of the relevance measures. This computation involves CB and CF contributions in a unified model able to weight dynamically their reciprocal relevance for generating high-quality suggestions. Furthermore, RBR allows users to preserve their desired privacy degree about the information on their interests and preferences. To test the performance of RBR, some experiments have been performed on simulated and real users. The obtained results show a significant improvement in the effectiveness of the suggestions generated by RBR which seem to meet users' orientations better than the other tested systems.

The paper is organized as follows. Related work are discussed in Section 2, while Section 3 describes the knowledge representation adopted in RBR. The proposed recommender system is illustrated in Section 4 and some experiments to evaluate the advantages introduced by RBR are presented in Section 5. Finally, in Section 6 some conclusions are drawn.

2 Related work

Hybrid recommenders are increasingly adopted in various contexts and they differ mainly for the nature of the algorithms exploited in generating suggestions and the way they are combined. A great number of hybrid recommenders which combine CB and CF components have been proposed in the literature by exploiting a large variety of "conventional" and "novel" CB and CF techniques (Adomavicius and Tuzhilin 2005; Burke 2002; Manouselis and Costopoulou 2007; Woźniak et al. 2014) or by integrating them into a unified model. More in detail, hybrid recommenders based on CB and CF components can be classified in:

– Linear: CF and CB predictions are first separately generated and then joined by: (*i*) weighting ratings or votes (Claypool et al. 1999; de Campos et al. 2010; Melville et al. 2002); (*ii*) selecting those CB and CF predictions appearing as the most interesting on the basis of suitable metrics or priority (Popescul et al. 2001); (*iii*) using thresholds about the maximum number suggestions to present (Rosaci et al. 2009).
– Fusion: CB features are embedded into a CF algorithm or vice versa. For instance, in Balabanovic and Shoham (1997) and Pazzani (1999) CB techniques analyse users' profiles and apply CF techniques to refine CB suggestions, while in Salter and Antonopoulos (2006) users profiles, represented as vectors, are indexed by means of a latent semantic and clustered to apply CB techniques.
– Unified: in this category, a global model incorporates both CB and CF approaches, e.g. rule based systems (Liu and Liou 2011), probabilistic approaches (Schein et al. 2002), learning machines (Gunawardana and Meek 2009), mobile learning agents (Rosaci and Sarnè 2013).

In the context described above, a well known hybrid recommender which linearly combines CB and CF suggestions is the Content-Boosted Collaborative Filtering (Melville et al. 2002) (CBCF). It recommends movies to users based on their past ratings, where each rating is provided on six textual classes. A user-rating matrix, which is very sparse, is built from the users' ratings. The CB predictor adopts a bag-of-words naive Bayesian text classifier (Mitchell 1997) trained by the user's ratings and returns a pseudo user-rating vector containing his/her actual ratings. All the pseudo user-rating vectors are joined to form a dense pseudo-ratings matrix, that is used to make CF predictions by exploiting a pure CF neighborhood-based algorithm (Herlocker et al. 1999), while the similarities among users are computed by means of a Pearson correlation.

To improve the prediction accuracy, context-aware strategies can be applied to enrich profiles with significant data. For example, MASHA (Rosaci and Sarnè 2006) and MUADDIB (Rosaci et al. 2009) are two XML agent-based CB and CF hybrid recommenders that in their predictions adapt their behaviors to consider both users' profiles and exploited devices. MASHA presents three main characteristics: (*i*) it constructs a global user profile by exploiting data coming from the different devices exploited by the user and providing them with different relevance degrees; (*ii*) it exploits two different agent types to build such a profile. The former, called *client agent* is associated with each device exploited by a user and it is able to manage only information deriving from the use of just that device; the latter, called *server agent*, is associated with the user and collects information coming from all the user's devices; (*iii*) it generates both CB and CF recommendations on the site-side by using a third agent type, called *adapter agent*, associated with each visited Web site. Then each client agent, which runs on the device, performs only a relatively light task for constructing its local user's profile, while the global one is built by the more powerful server agent and the recommendations are generated in a centralized manner by the site adapter agent.

MUADDIB is a recommender that, with respect to MASHA, adopts a more efficient distributed architecture and a more effective recommendation algorithm. In particular, the MUADDIB platform exploits another type of agent, *called recommender agent*. Each recommender agent affiliates users similar for interests and provides to pre-compute CB and CF suggestions for them based on their global profiles and on the catalogs of the Web sites affiliated with the platform. When a user is visiting a Web site affiliated with MUADDIB, his/her device agent provides the site agent with the information for contacting the recommender agents where its visitor is affiliated to in order to receive their best personalized suggestions for him/her. Note that MUADDIB and MASHA can work as traditional hybrid recommender systems without considering the exploited devices in generating recommendations. Moreover, similarly to RBR, they significantly preserve users' privacy by performing a consistent part of their computations on the client-side, while MUADDIB differs from them for the adoption of a full distributed architecture.

Computational costs of hybrid recommenders, which integrate CB and CF components by adopting a linear or a fusion modality, depend mainly on the computational costs of their CB and CF components. For the CB component such costs depend on the numbers of items (m) and their features (f), which give a complexity of $\mathcal{O}(f \cdot m^2)$. CF algorithms are more expensive than CB methods because they have to compare all the items and their features for all the n users, the cost, in the worst case, is $\mathcal{O}(f \cdot m \cdot n)$. The usual high sparsity degree of users' profiles makes this cost closer to $\mathcal{O}(f \cdot (m + n))$. In the cases of MASHA and MUADDIB, their costs are respectively $\mathcal{O}(n^2 \cdot c)$ and $\mathcal{O}(c + p)$, where c represents the number of associated categories and p that of the partitions. On the contrary, if there is a a great number of users and items, the *item-to-item* CF algorithm (Linden et al. 2003) compares accessed items to similar items to search those accessed together. In this way, its computational cost depends only on the number of visited items. To save computational resources, some CF algorithms exploit clustering processes (Berkhin 2006; Jain et al. 1999) to reduce the searching space and/or compute similarities off-line. Specific analysis are required to know the computational cost of unified hybrid recommenders. In fact, their computational complexities cannot be inferred from those of the CB and CF components given the impossibility to identify the respective costs.

The proposed unified hybrid recommender RBR differs from all the cited systems mainly for the generation of the suggestions (see Section 4). Indeed, it exploits users' relevance measures of each instance (i.e. a product or information) computed by taking account of the existing interdependencies among all the analogous measures. In the author's knowledge, this is a unique characteristic among the recommenders and RBR can evaluate more precisely the relevance of each instance into a users' community. Consequently, RBR allows the most interesting instances to be identified better then other recommender systems. This is shown in Section 5 where RBR is compared with CBCF (Melville et al. 2002), MASHA (Rosaci and Sarnè 2006) and MUADDIB (Rosaci et al. 2009). Finally, the better performances of RBR require significant computational costs that can be optimized by clustering users and pre-computing suggestions as described in Section 4.2.

3 The knowledge representation

The knowledge representation adopted in RBR is mainly based on the use of: (*i*) a common *Dictionary* ($\mathcal{D}$) storing those *categories* and *instances* recommendable by RBR; (*ii*) an individual *User Profile* (UP) storing those *categories* and *instances* of $\mathcal{D}$ of interest for the user on the basis of his/her monitored Web activities (for example, by exploiting one or more software agents running on his/her clients (Rosaci et al. 2013; Rosaci et al. 2009)). Other data structures used by RBR to store data temporarily and generated recommendations will be described in Section 4.1.

The *Dictionary* $\mathcal{D}$ is organized in *categories* and *instances* and it is periodically updated to consider new categories and instances of interest for the users. Each *Dictionary* category consists of a set of instances associated with it and each instance belongs to only one category. More in detail, each category c/instance i is identified by a pair of attributes $(Cid_c, Cd_c)/(Iid_i, Id_i)$, where Cid_c/Iid_i is a unique *code* identifying the category/instance and Cd_c/Id_i is its textual *description*. The *Dictionary* is implemented by a unique XML-Schema, shown in the Appendix.

Interests and preferences of each user with respect to the categories and the instances stored in $\mathcal{D}$ are collected in an individual user profile associated with him/her and implemented by a XML-Schema. Let u be a user and let UP^u be his/her profile (Figure 1). The information stored in UP^u are:

– *Delta* (δ^u): a coefficient autonomously set by u in $]0, 1[\in \mathbb{R}$ and used in computing IW_i^u measures (see below);

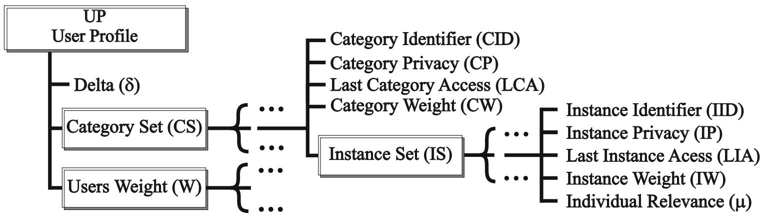


Fig. 1 The XML elements of the *User Profile*

- *Category Set* (CS_c^u): a set where each element is associated with a category c of interest for u and in turn consists of:
  - *Category Identifier* (Cid^c): the code associated with c in $\mathscr{D}$;
  - *Category Privacy* (CP_c^u): a flag set by u to $0/1$ to make public/private his/her interest in c. Note that if a category is set as private then all its associated instances become private;
  - *Last Category Access* (LCA_c^u): the date of the last access of u to an instance belonging to the category c;
  - *Category Weight* (CW_c^u): a measure of the interest of u about c (see below) which ranges in $[0, 1] \in \mathbb{R}$;
  - *Instance Set* (IS_c^u): a set where each element represents an instance i belonging to the category c visited by u and in turn consists of:
    - *Instance Identifier* (Iid^i): the code associated with the instance i in $\mathscr{D}$;
    - *Instance Privacy* (IP_i^u): a flag that u sets to $0/1$ to make public/private his/her interest in i;
    - *Last Instance Access* (LIA_i^u): the date of the last access of u to i;
    - *Instance Weight* (IW_i^u): a measure of the interest of u about i (see below) which ranges in $[0, 1] \in \mathbb{R}$;
    - *Individual Relevance* (μ_i^u): a coefficient, ranging in $[0, 1] \in \mathbb{R}$, which takes account of the individual relevance of i for u with respect to the other instances of interest for u (see Section 4).
- *User Weight* (W): a set of coefficients exploited by u in computing his/her instance weight measures (see below). Each element consists of the user identifier of j (i.e. Uid_j) and a value W_j^u, belonging to $[0, 1] \in \mathbb{R}$, to weight the information provided by the user j to the user u about an instance.

IW and CW measures are computed on the whole user's Web history. To measure the user's interest in the content of a Web page, several approaches have been proposed in the literature. In particular, in Chan (2000) the authors suggest to consider the time spent by a user on a page, its length and a possible score provided by the user to show his/her interest level about that Web page. In Parsons et al. (2004) too, the visiting time of a Web page is the main parameter considered in evaluating the user's interest in the instances present therein, while in Rosaci and Sarnè (2006) the typology of the device exploited in its access is also taken into account. All these approaches evaluate the user's interest in an instance belonging to a Web page by using the time spent in visiting that Web page.

In RBR, the interest of the user u about an instance i (i.e. IW_i^u) set by him/her as public is measured by two factors: (i) his/her whole past Web history in accessing i ; (ii) the time t (measured in seconds) spent by u in his/her actual visit to the page containing i . Each user can decide how to weight these two components of IW_i^u by setting autonomously the coefficient δ^u, a real value belonging to $]0; 1[$, to tune the "memory" behavior of IW_i^u. In particular, to calculate the new IW_i^u its actual value is weighted by $(1 - \delta^u)$, while the new contribution to IW_i^u, due to the current visit of the user u to i, is weighted by δ^u. Furthermore, when IW_i^u is updated, its current value is decreased based on its age. This latter is computed as difference (in days) between the dates of the present and last visit to i performed by u. To this purpose, the function $\mathscr{F}(current, past)$ is exploited. $Current$ is the actual date and $past$ is assumed to be equal to the value of $LIA_i^u \in UP^u$, associated with the instance i. More in detail, the contribution provided by the current value of IW_i^u is set to 0 when the last access is older than a year. Formally, IW_i^u is calculated as:

$$IW_i^u = (1 - \delta^u) \cdot \mathscr{F}(current, past) \cdot IW_i^u + \delta^u \cdot t \tag{1}$$

where

$$\mathscr{F}(current, past) = \begin{cases} 1 - \frac{current-past}{365}, & \text{if}(current - past) \leq 365 \\ 0, & \text{otherwise} \end{cases} \tag{2}$$

In this way a reasonable priority to new items is given. Then LIA_i^u is updated to the current date and the measure of the interest of u in a public category c (i.e CW_c^u) is computed as the average of all the IW measures of the instances belonging to c and set as public by u. Finally, LCA_c^u is set to the most recent date stored in the LCI_i^u parameters associated with the instances belonging to c.

4 The relevance based recommender

This section describes the proposed unified hybrid Relevance Based Recommender (RBR) system. This tool exploits and adapts the mathematical model described and applied in Buccafurri et al. (2004) and Rosaci et al. (2012) in order to generate effective recommendations.

As described in the Introduction, RBR is based on the computation of a measure of relevance degree for each recommendable instance (i.e. a product or information) for each affiliated user. The RBR peculiarity is that each relevance measure takes into account explicitly the existing interdependencies among the analogous measures of all the other users by means of simple linear systems of equations. Therefore, each relevance measure permeates all the other relevance measures by assuming also a "social" dimension that makes possible to obtain more precise measures.

More in detail, let u be a user belonging to a community of n users, called $\mathscr{A}$. To build his/her opinion about an instance i (called *instance relevance*) the user considers (*i*) the measure of how much i is significant for him/her (called *individual relevance*), based on his/her personal point of view, and (*ii*) the measure of how much i is important for the other users belonging to $\mathscr{A}$ (called *social relevance*), by exploiting the instance relevance measures provided by the other users of $\mathscr{A}$. Moreover, RBR allows the contributions due to the individual and social relevance measures to be dynamically and individually weighted for each user.

More formally, let ρ_i^u be the measure of the instance relevance that the user $u \in \mathscr{A}$ has for a given instance i. It is obtained by combining the individual opinion (μ_i^u) of u and a social contribution (γ_i^u) consisting of the relevance measures of the instance i provided to u by each user $j \neq u \in \mathscr{A}$. These two components are combined in a unique measure (i.e. ρ_i^u) by means of the real coefficient α_c^u by taking into account the expertise level of u in $\mathscr{A}$ about the category c, which the instance i belongs to. In particular, a high value of α_c implies a greater relevance of μ with respect to γ in calculating the measure of the relevance ρ for an instance belonging to the category c. Therefore, the instance relevance ρ_i^u can be generally expressed as a function $\mathscr{G}$ depending on μ_i^u, γ_i^u and α_a^c (i.e., $\rho_i^u = \mathscr{G}(\mu_i^u, \gamma_i^u, \alpha_c^u)$), where ρ, μ, γ and α are real values belonging to [0, 1]. In the following, μ, γ and α will be described in detail.

The individual relevance. The individual relevance μ_i^u of an instance i is computed by u based on the *Interest Weight* measures IW stored in his/her profile UP^u. The IW measures are not directly available because they need to be (i) updated to the current date by using the function $\mathscr{F}$ (with $past = LIA_i^u$) and (ii) normalized with respect to the sum of the IW measures of all the m instances stored in UP^u that belong to the same category c of i. Therefore, the value μ_i^u for the generic instance i is computed as:

$$\mu_i^u = \frac{IW_i^u}{\sum_{r=1}^{m} IW_r^u} \tag{3}$$

The social relevance This contribution to ρ_i^u is provided by γ_i^u as the average of the ρ measures of relevance of i required by u to each of the other $(n-1)$ users belonging to his/her community $\mathscr{A}$. In computing γ_i^u, each exploited relevance measure is weighted by the reliability that u assigned to the user provided it. Therefore, γ_i^u represents the measure of the relevance of i within $\mathscr{A}$ as it is perceived by u. More formally, γ_i^u is computed by the user u as:

$$\gamma_i^u = \frac{1}{n-1} \cdot \sum_{j=1, \forall j \neq u}^{n-1} W_j^u \cdot \rho_i^j \tag{4}$$

where W_j^u (a real value ranging in [0, 1]) is the measure of the reliability of j computed by u to weight the relevance measure provided by j. It is based on the concordance degree between the ρ measures of u and j for the instance i. In particular, each time that u exploits an instance relevance measure provided by j, he/she also computes his/her W_j^u value by taking account of the average of all the latest d (an integer value that is a system parameter) differences among the instance relevance measures computed by u and those provided by the user j to him/her. Note that each newcomer user has initially the maximums reliability, i.e. $W = 1$. Formally, W_j^u is computed as:

$$W_j^u = 1 - \frac{1}{d} \cdot \sum_{i=1}^{d} \left| \rho_i^u - \rho_i^j \right| \tag{5}$$

For example, when u needs to evaluate the social relevance measure of i, belonging to c, with respect to a community of four users, he/she receives three "opinions" from the users j_1, j_2 and j_3 (Fig. 2). Such opinions are represented by their instance relevance measures about the instance i (i.e. $\rho_i^{j_1}$, $\rho_i^{j_2}$ and $\rho_i^{j_3}$). By assuming $W_j^u = 1.0\ \forall j$, the social relevance ρ_i^u computed by u is $\gamma_i^u = (1 \cdot 0.8 + 1 \cdot 0.9 + 1 \cdot 0.4)/(4-1) = 0.7$.

The α_c^u coefficient This coefficient measures the expertise level of the user u on the category c which the instance i belongs to, within his/her community. It is used to automatically weight the social and the individual relevance contributions in computing the relevance measure of i. Therefore, a different α has to be computed for each of the p categories stored

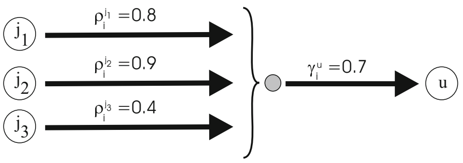


Fig. 2 The social relevance computed by the user u based on the perceived relevance of the users j_1, j_2 and j_3

in the user profile of u with respect to the maximum *Category Weight* measure (CW^{max}) within his/her community. However, each CW measure has to be updated to its latest LCA date by the function $\mathscr{F}$. More formally, for the category c the coefficient α_c^u is computed by u as:

$$\alpha_c^u = \frac{CW_c^u}{CW_c^{max}} \tag{6}$$

The instance relevance Finally, for a user u the instance relevance measure ρ_i^u of an instance i takes account of both the individual (μ_i^u) and the social (γ_i^u) relevance of that instance suitably weighted by u on the basis of his/her expertise level (α_c^u) on the category c which i belongs to. The function $\mathscr{G}$ of the (4) has the form:

$$\rho_i^u = \alpha_c^u \cdot \mu_i^u + (1 - \alpha_c^u) \cdot \gamma_i^u \tag{7}$$

that, by taking account of the (3), (4) and (6), becomes:

$$\rho_i^u = \frac{CW_c^u}{CW_c^{max}} \cdot \frac{IW_i^u}{\sum_{r=1}^{m} IW_r^u} + \left(1 - \frac{CW_c^u}{CW_c^{max}}\right) \cdot \frac{1}{n-1} \cdot \sum_{j=1}^{n-1} W_j^u \cdot \rho_i^j \tag{8}$$

where m and n are respectively the numbers of instances belonging to the category c and the number of users belonging to $\mathscr{A}$.

Therefore, for all the n users belonging to $\mathscr{A}$ the *instance relevance* measures of i are obtained by solving a system of n linear equations in n variables (in the form of the (8)). This system is equivalent to that described in Buccafurri et al. (2004) and Rosaci et al. (2012) and admits only one solution. To compute the *instance relevance* measures of m instances for n users, m linear systems of n equations in n variables have to be solved.

4.1 The RBR algorithm

The RBR algorithm runs periodically in order to pre-compute updated and supposedly interesting suggestions for its users. For each user RBR suggests to him/her the first M most relevant and unvisited instances belonging to the first most attractive P categories for that user. Note that the two system parameters M and P are two positive integer values that have to be considered as two upper bounds. In presence of newcomer users and/or too many categories and instances set as private then RBR might not be able to select/identify all the desired categories and instances. Furthermore, note that usually $M << m$ and $P << p$, where m and p are the overall numbers of instances and categories stored into the dictionary $\mathscr{D}$.

The RBR algorithm is implemented by the function `RBR ( )` that in turn calls the functions `Relevance ( )` and `Recommendations ( )`. The first one returns the most relevant instances for each category of interest with respect to the instances and the categories in $\mathscr{D}$ and the user community $\mathscr{A}$, while the other function provides to extract iteratively the most relevant M suggestions for each user. The algorithm implementing RBR is described in the form of pseudo-code in Fig. 3. The function `RBR ( )` receives in input the list of the n users belonging to $\mathscr{A}$ and the integers P and M and returns in output the set $RSet_{\mathscr{A}}$ that is the set of the recommendation lists computed for each user of $\mathscr{A}$. Initially, the function `RBR ( )` receives from each user his/her (*i*) "public" CW and IW measures (updated to the current date by the function $\mathscr{F}$ with *past* respectively set to LCA and LIA) and (*ii*)

```
recomListSet RBR(userList $\mathscr{A}$, integer $P$, integer $M$) {
    for($j = 0$; $j < n$; $j$++) {
        DS $DS_j$=receive(user $j$);
        $DSSet_{\mathscr{A}} = DSSet_{\mathscr{A}} \bigcup DS_j$; }
    relevanceSet $RSet_{\mathscr{A}}$=Relevance(userList $\mathscr{A}$, DS $DSSet_{\mathscr{A}}$, integer $P$, integer $M$)
    for($j = 0$; $j < n$; $j$++) {
        recomList $RecList_j$ = Recommendations(user $j$, DS $DS_j$ relevanceSet $RSet_{\mathscr{A}}$, integer $P$, integer $M$)
        $RecListSet_{\mathscr{A}} = RecListSet_{\mathscr{A}} \bigcup RecList_j$; }
    return $RecListSet_{\mathscr{A}}$; }


relevanceSet Relevance(userList $\mathscr{A}$, DS $DSSet_{\mathscr{A}}$, integer $P$, integer $M$) {
    for($j = 0$; $j < n$; $j$++) {
        instanceSet $ISet_j$=SelectTop(user $j$, DS $DS_j$, integer $P$, integer $M$);
        $ISet_{\mathscr{A}} = ISet_{\mathscr{A}} \bigcup ISet_j$;}
    for($i = 0$; $i < k$; $i$++) {
        relevanceSet $RSet_i$=InstanceRelevance(instance $i$, DS $DSSet_{\mathscr{A}}$);
        $RSet_{\mathscr{A}} = RSet_{\mathscr{A}} \bigcup RSet_i$;}
    return $RSet_{\mathscr{A}}$; }


recomList Recommendations(user $j$, DS $DS_j$, relevanceSet $RSet_{\mathscr{A}}$, integer $M$) {
    instanceList $VList_j$=Visited(user $j$, DS $DS_j$);
    relevanceSet $UVSet_j$=Prune(relevanceSet $RSet_{\mathscr{A}}$, instanceList $VList_j$);
    recomList $RecList_j$=Sort(relevanceSet $UVSet_j$);
    recomList $RecList_j$=Extract(relevanceSet $UVSet_j$, integer $P$, integer $M$);
    return $RecList_j$; }
```

Fig. 3 The Relevance Based Recommender Algorithm

the weights W. Such values are stored in a data structure DS, associated with that user, belonging to the set $DSSet$ collecting all these users' data structures. To realize the two main tasks of the algorithm, first the function `Relevance ( )` is called and after, for each of the n users of $\mathscr{A}$, the function `Recommendations ( )` runs to generate his/her recommendation list. Then each recommendation list is joined to the set $RListSet_{\mathscr{A}}$ returned by RBR.

The first function `Relevance ( )` receives in input the list of users of $\mathscr{A}$ and the set $DSSet$ with the integers P and M and provides in output the set $RSet_{\mathscr{A}}$ storing, for each category of $\mathscr{D}$, the instances mostly significant in $\mathscr{A}$ computed on the basis of the individual public measures of interest (i.e. CW and IW) provided by each of the n users of $\mathscr{A}$. For each one of the n users of $\mathscr{A}$ the function `Relevance ( )` calls the function `SelectTop ( )`. For each user $j \in \mathscr{A}$ this function receives in input the same user j, his/her associated data structure DS_j and the integers P and M previously described, while its output consists of the set $ISet_j$. The output set $ISet_j$ stores the first M top-ranked public instances, for each of the first P top-ranked public categories of interest for j, associated with their IW value computed by j. Then the union of this set with the global set $ISet_{\mathscr{A}}$ is performed. The set $ISet_{\mathscr{A}}$ collects all the top-ranked instances in $\mathscr{A}$, for each category of $\mathscr{D}$, associated with a global IW value obtained by summing all the individual IW contributions. Let k be the number of different instances stored in $ISet_{\mathscr{A}}$, with $k \leq m$, and for each of them the function `InstanceRelevance ( )` is called.

The function `InstanceRelevance ( )` performs the most expensive computation in RBR because it has to solve the linear system required by the mathematical model previously described for each of the k instances stored in $ISet_{\mathscr{A}}$. Let $RSet_i$ and $RSet_{\mathscr{A}}$ be respectively the set storing the instance relevance measures computed for the instance $i \in ISet_{\mathscr{A}}$ (i.e. the ρ_i values of each user belonging to $\mathscr{A}$) and the set union of all the $RSet$ of each instance belonging to $ISet_{\mathscr{A}}$. For each instance $i \in ISet_{\mathscr{A}}$ the function `InstanceRelevance ( )` receives in input the instance i and the global set $DSSet$ described above and returns the set $RSet_i$ that contributes to form the global set $RSet_{\mathscr{A}}$. Finally, $RSet_{\mathscr{A}}$ is returned by the function `Relevance ( )`.

The second main function is `Recommendations ( )` which selects the best M suggestions for a target user. This function receives in input a target user (i.e. j), his/her associated data structures (i.e. DS_j), the set of instance relevance measures $IRSet_{\mathscr{A}}$ previously computed by the function `Relevance ( )` and the already introduced integers P and M. The output returned by this function is a list of the M unvisited instances potentially most attractive for the user j. They are selected as described in the following. The function `Visited ( )` receives the user j and his/her associated DS_j from which selects all the visited instances belonging to the P top-ranked categories of interest for j and that are returned in output in the list $VList_j$ (for a newcomer user this set will be obviously empty). Then (*i*) the function `Prune ( )` is called to delete from the set $RSet_{\mathscr{A}}$ all the instances visited by j and stored into the list $VList_j$, the remaining instances are returned in $UVSet_j$, (*ii*) the function `Sort ( )` orders in $RecList_j$ the instances of $UVSet_j$ in a decreasing way based on their relevance measures and (*iii*) the function `Extract ( )` returns in $RecList_j$ the only first M unvisited instances that are the recommendations computed by RBR for j. Finally, $RecList_j$ is returned by `Recommendations ( )`.

4.2 The RBR computational complexity

Let n, m and p be the overall number of considered users and those of instances and categories belonging to the dictionary $\mathscr{D}$, respectively. To compute the perceived instance relevance within a community of n users RBR has to solve a system of n linear equations in n variables that admits only one solution (Buccafurri et al. 2004; Rosaci et al. 2012). The resulting linear system is dense and if solved by using classical direct methods its computational complexity in time is of $\mathscr{O}(n^3)$. To determine the perceived instance relevance measures of all the m instances belonging to $\mathscr{D}$, RBR requires to solve m systems of n linear equations in n variables with a complexity in time of $\mathscr{O}(m \cdot n^3)$.

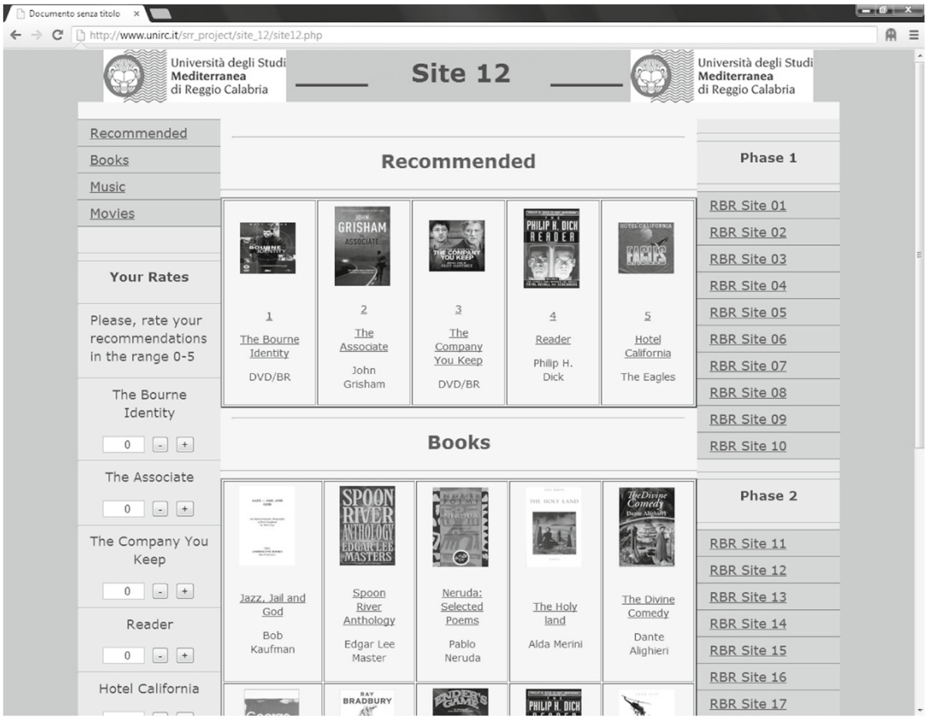
This analysis shows that RBR has a significant high cost but there some elements to assess it correctly. The first considered element to limit the computational cost takes account of users' similarities by a clustering process, if there is a wide number of users. In RBR the adopted user profile allows an easy and effective clustering process. Therefore, let $\overline{n}$ be the average number of users for cluster, with $\overline{n} << n$. The overall number of clusters will be approximatively $n/\overline{n}$. For each instance, RBR has to solve $(n/\overline{n}) < n$ linear systems of $\overline{n}$ equations in $\overline{n}$ variables with a computational complexity of $\mathscr{O}(\frac{n}{\overline{n}} \cdot \overline{n}^3) = \mathscr{O}(n \cdot \overline{n}^2) < \mathscr{O}(n^3)$. Furthermore, let M be the number of top-ranked instances for each considered category of interest and P the number of top-ranked categories of interest for which, in the worst case, RBR has to compute the recommendations. Commonly, it results $P << p$ and $M << m$ and therefore, in the worst case, the maximum number of instances for which their relevance measures are computed will be $q = M \cdot P$, with $q << m$. In this way, the complexity of RBR will be $\mathscr{O}(q \cdot n \cdot \overline{n}^2) << \mathscr{O}(m \cdot n^3)$.

The other element is the cost C to perform a clustering. In the previously assumed hypothesis, the overall cost of this recommender will be $\mathscr{O}(C + q \cdot n \cdot \overline{n}^2)$, where C depends on the particular clustering algorithm adopted. Finally, such recommendations are computed off-line for each user and then they can be provided to him/her in a constant time without delay for the target user. Therefore, under the usual conditions, RBR runs with a computational cost that is significantly lower than in the worst case.

5 Experiments

In this section, two different experiments to test the effectiveness of RBR in generating potentially interesting recommendations are presented. To this aim, the performances of RBR have been compared with those of other three recommender systems, namely MUAD-DIB, CBCF and MASHA described in Section 2. These competitors have been implemented in accordance with their descriptions respectively provided in Rosaci et al. (2009), Melville et al. (2002) and Rosaci and Sarnè (2006). Moreover, MUADDIB and MASHA have been tested without considering their adaptivity with respect to the exploited users' devices.

The first experiment refers to more sets, different for size, of simulated users while the second one refers to the data of 80 real users. Both the experiments have required: (*i*) a common *Dictionary* consisting of 20 categories (i.e. $p = 20$), each one provided with 90 instances (i.e. $m = 90$), implemented by a unique XML Schema shown in the Appendix; (*ii*) 20 XML Web sites (Fig. 4), each one dealing with only three categories of interest, belonging to the common dictionary, and with 40 instances for category; (*iii*) 3 top-ranked categories (i.e. $P = 3$) and 5 top-ranked instances (i.e. $M = 5$); (*iv*) the system parameter d of (5) has been reasonably set to 5. The first 10 Web sites have been used for generating/building the user profiles (see Section 3) of the simulated/real users, while the other 10 Web sites have been exploited for measuring the performances of the tested recommenders. For sake of simplicity, all the information stored in the user profiles have been considered as public.



Recommended

1 The Bourne Identity DVD/BR
2 The Associate John Grisham
3 The Company You Keep DVD/BR
4 Reader Philip H. Dick
5 Hotel California The Eagles

Please, rate your recommendations in the range 0-5

The Bourne Identity 0
The Associate 0
The Company You Keep 0
Reader 0
Hotel California

Books

Jazz, Jail and God Bob Kaufman
Spoon River Anthology Edgar Lee Master
Neruda: Selected Poems Pablo Neruda
The Holy land Alda Merini
The Divine Comedy Dante Alighieri

Fig. 4 The RBR Web site 12

For each simulated/real user u and for the g-th tested recommender system (with $g = 1, \cdots 4$) a set of recommendations has been generated and stored in a different list L_g^u consisting of the suggestions $l_{g,s}^u$ (with $s = 1, \cdots, M$). Then, the recommendations contained in L_g^u have been divided into two lists of *good recommendations* (denoted by G_g^u) and *bad recommendations* (denoted by B_g^u), based on their supposed relevance for u. More in particular, the recommendations having relevance greater or equal than 0.5 have been assigned to G_g^u, while those with a relevance smaller than 0.5 have been inserted into B_g^u. To measure the quality of each recommendation generated for u, a rate $r_{g,s}^u$ (i.e. an integer ranging in [0, 5]) has been computed in the first experiment based on the information stored in the simulated profile or directly assigned by the real user in the other experiment. If, for u and for each suggestion $l_{g,s}^u \in G_g^u$, the rate $r_{g,s}^u$ is greater or equal than 3, it is considered as a *true positive* and inserted in the list TP_g^u containing all the true positives of L_g^u. If $r_{g,s}^u$ is smaller than 3, it is considered as a *false positive* and inserted in the list FP_g^u of that user. Finally, for u and for each suggestion $l_{g,s}^u \in B_g^u$, if the rate is greater or equal than 3, the suggestion is considered as a *false negative* and inserted in the list FN_g^u. After the first phase necessary to generate/build the user profiles, in both the experiments only the first 100 recommendation sessions have been generated by each considered recommender (i.e. about 500 suggestions for recommender).

The performances of the recommenders have been evaluated by means of the standard measures *Precision* (P), *Recall* (R) and *F-measure* (F) (van Rijsbergen 1974). *Precision* can be interpreted as the probability that a suggestion is considered relevant by the user, while *Recall* can be considered as the probability that it is relevant and the *F-Measure* represents their harmonic mean. Formally, for the user u and for the g-th recommender system P_g^u, R_g^u and F_g^u are computed as:

$$P_g^u = \frac{|TP_g^u|}{|TP_g^u \cup FP_g^u|} \tag{9}$$

$$R_g^u = \frac{|TP_g^u|}{|TP_g^u \cup FN_g^u|} \tag{10}$$

$$F_g^u = \frac{2 * R_g^u * P_g^u}{R_g^u + P_g^u} \tag{11}$$

Furthermore, the Average Precision $\overline{P}\,_g^u$ (resp. Average Recall $\overline{R}\,_g^u$ and Average F-Measure $\overline{F}\,_g^u$) of each system is defined as the average of the P_g^u (resp. R_g^u and F_g^u) values of all the considered users.

Simulated users For the first experiment three sets of 250 (S_1), 500 (S_2) and 1000 (S_3) simulated users (and their profiles) have been built. Moreover, other 10.000 simulated users (and their profiles) have been generated and distributed on 20 clusters by means of a partitional clustering algorithm by using the Jaccard similarity measure on the basis of their generated profiles. To this purpose, each user has been represented by a pattern consisting of CW values (normalized among them) of the 20 exploited categories of interest considered in the experiment. The parameter δ (see Section 3), to tune the memory in computing the interest weights has been set to a reasonably value of 0.3.

To realize this experiment, the individual interests of the simulated users have been generated. Two different and complementary activities have been performed to this purpose. The first one refers to order into the dictionary $\mathscr{D}$ all the instances of each category based on their similarity. In this way, similar instances have sequential identifier codes. The other one

Table 1 The $\overline{P}^{u}_{g}/\overline{R^{u}_{g}}/\overline{F^{u}_{g}}$ measures obtained in the case of simulated users for each recommender system, for the three sets S_1, S_2, S_3, for the best cluster ($\mathscr{C}^*$) and in average on all the clusters ($\overline{\mathscr{C}}$)

	RBR	MUADDIB	CBCF	MASHA
S_1	0.85/0.80/0.82	0.78/0.71/0.74	0.72/0.67/0.69	0.74/0.70/0.72
S_2	0.82/0.76/0.79	0.74/0.67/0.70	0.62/0.60/0.61	0.68/0.65/0.67
S_3	0.79/0.75/0.77	0.67/0.65/0.66	0.58/0.53/0.55	0.64/0.60/0.62
$\mathscr{C}^*$	0.83/0.76/0.79	0.75/0.68/0.71	0.63/0.59/0.61	0.69/0.64/0.66
$\overline{\mathscr{C}}$	0.81/0.76/0.78	0.73/0.66/0.69	0.61/0.60/0.60	0.67/0.60/0.63

refers to provide each user with a set of "behaviors". In particular, each behavior consists of a set of categories and a little share of their instances described by means of a domain identified with an initial and an ending instance identifier code. Such behaviors result to be partially overlapped and, therefore, the same category and/or set of instances can be present more times for the same user. This allows each user to be provided with different levels of interest in the categories and in the instances of $\mathscr{D}$ based on the number of time that a category or an instance is repeated. Then, for a simulated user his/her choices of the site instances and his/her interest in the suggested instances are simulated based on assigned behaviors.

In terms of results (Table 1) RBR is always the best performer for all the considered sets/clusters of users. The advantage of RBR in terms of Average Precision (i.e., Average Recall and Average F-measure) ranges in $9 \div 18$ % (i.e., $12 \div 15$ % and $11 \div 17$ %) with respect to the second best performer that is always MUADDIB. Moreover, the performances of RBR are almost similar when the size of the set increases, differently from the other tested recommender systems.

Real users Firstly the profiles of the 80 real users have been built by exploiting the first 10 Web sites for a three-week period. In this phase, real users set the parameter δ individually (0.42 in average with respect to all the real users). The quality of the recommendations generated by RBR and by the other recommender systems for supporting the users have been tested on the other 10 Web sites and individually ended when all the tested recommenders generated 100 recommendation lists, as above specified. The overall time required by this phase has been of about nine weeks with a minimum of about six weeks (this time varied for each of the real user based on his/her time spent on the platform).

The results (Table 2) show that in terms of Average Precision (resp. Average Recall, Average F-measure) the advantage of RBR with respect to the second best performer (i.e. MUADDIB) is 9 % (i.e., 13 % and 11 %), which substantially confirms the results of the first experiment.

Discussion The experimental results show that the performances of RBR are better than the other tested recommenders, in terms of average precision, average recall and average F-measure. It is possible to argue that the advantage of RBR with respect to the other

Table 2 The $\overline{P}^{u}_{g}/\overline{R^{u}_{g}}/\overline{F^{u}_{g}}$ measures obtained from each recommender system for the set of real users

	RBR	MUADDIB	CBCF	MASHA
Real users	0.73/0.69/0.71	0.67/0.61/0.64	0.62/0.57/0.59	0.63/0.60/0.61

recommender systems can be attributed to its peculiarity, i.e. the dynamic interdependencies among all the computed instance relevance measures, as described in Section 4. In this way, RBR has the opportunity to identify accurately the most interesting instances to be recommended. Moreover, another characteristic behavior shown by RBR is that the performances are almost uniform when the population size becomes greater, while those of the other recommender systems change significantly.

6 Conclusions

In this paper a recommender system, called Relevance Based Recommender (RBR), able to generate personalized high quality suggestions potentially attractive for the users has been presented. For each user, RBR computes precise single relevance measures of products or information of interest for him/her. These measures takes account of the interdependencies existing among all the users' relevance measures of such instances in order to improve their precision. Then such measures are exploited by RBR to generate personalized recommendations. Some experimental evaluations performed on simulated and real users have shown that RBR provides better performances in terms of effectiveness then some other tested recommender systems.

In the next future, researches will be addressed to improve the RBR efficiency in terms of computational complexity and investigate on a possible distributed implementation of the RBR algorithm.

Appendix

The Section Appendix presents the two XML-Schema of the *Dictionary* and of the *User Profile* described in Section 3.

The Dictionary XML-Schema

```
<xs:schema attributeFormDefault=”unqualified” elementFormDefault=”qualified”
                                    xmlns:xs=”http://www.w3.org/2001/XMLSchema”>
  <xs:element name=”Dictionary”>
    <xs:complexType>
      <xs:sequence>
        <xs:element name=”Category” maxOccurs=”unbounded” minOccurs=”1”>
          <xs:complexType>
            <xs:sequence>
              <xs:element name=”Instance” maxOccurs=”unbounded” minOccurs=”1”>
                <xs:complexType>
                  <xs:simpleContent>
                    <xs:extension base=”xs:string”>
                      <xs:attribute type=”xs:ID”     name=”Iid” use=”required”/>
                      <xs:attribute type=”xs:string” name=”Id”  use=”required”/>
                    </xs:extension>
                  </xs:simpleContent>
                </xs:complexType>
              </xs:element>
            </xs:sequence>
            <xs:attribute type=”xs:ID”     name=”Cid” use=”required”/>
            <xs:attribute type=”xs:string” name=”Cd”  use=”required”/>
          <xs:complexType>
        </xs:element>
      </xs:sequence>
    </xs:complexType>
  </xs:element>
</xs:schema>
```

The User Profile XML-Schema

```
<xs:schema attributeFormDefault="unqualified" elementFormDefault="qualified"
                                    xmlns:xs="http://www.w3.org/2001/XMLSchema">
 <xs:element name="UserProfile">
  <xs:complexType>
   <xs:sequence>
    <xs:element type="xs:float" name="Delta"/>
    <xs:element name="CS" maxOccurs="1" minOccurs="1">
     <xs:complexType>
      <xs:sequence>
       <xs:element type="xs:string" name="Cid"  />
       <xs:element type="xs:byte"   name="CP"   />
       <xs:element type="xs:string" name="LCA"  />
       <xs:element type="xs:float"  name="CW"   />
       <xs:element name="IS" maxOccurs="unbounded" minOccurs="1">
        <xs:complexType>
         <xs:sequence>
          <xs:element type="xs:string" name="Iid"  />
          <xs:element type="xs:byte"   name="IP"   />
          <xs:element type="xs:string" name="LIA"  />
          <xs:element type="xs:float"  name="Mu"   />
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
    <xs:element name="Weight">
     <xs:complexType>
      <xs:sequence>
       <xs:element name="USER" maxOccurs="unbounded" minOccurs="0">
        <xs:complexType>
         <xs:sequence>
          <xs:element type="xs:ID"    name="Uid" use="required" />
          <xs:element type="xs:float" name="W"   use="required" />
         </xs:sequence>
        </xs:complexType>
       </xs:element>
      </xs:sequence>
     </xs:complexType>
    </xs:element>
   </xs:sequence>
   <xs:attribute type="xs:ID" name="Uid" use="required"/>
  </xs:complexType>
 </xs:element>
</xs:schema>
```

References

Adomavicius, G., & Tuzhilin, A. (2001). Using data mining methods to build customer profiles. *Computer*, *34*, 74–82.

Adomavicius, G., & Tuzhilin, A. (2005). Toward the next generation of recommender systems: a survey of the state-of-the-art and possible extensions. *IEEE Transactions on Knowledge and Data Engineering*, *17*(6), 734–749.

Balabanovic, M., & Shoham, Y. (1997). Content-based, collaborative recommendation. *Communication of the ACM*, *40*(3), 66–72.

Berkhin, P. (2006). A survey of clustering data mining techniques. In J. Kogan, C. Nicholas, M. Teboulle (Eds.), *Grouping multidimensional data pages* (pp. 25–71). Springer-Verlag.

Breese, J., Heckerman, D., Kadie, C. (1998). Empirical analysis of predictive algorithms for collaborative filtering. In *Proceedings of the 14th international conference on uncertainty in artificial intelligence* (pp. 43–52). Morgan Kaufmann.

Buccafurri, F., Palopoli, L., Rosaci, D., Sarné, G.M.L. (2004). Modeling Cooperation in Multi-Agent Communities. *Cognitive Systems Research*, *5*(3), 171–190.

Burke, R.D. (2002). Hybrid recommender systems: Survey and experiments. *User Modeling and User-Adapted Interaction*, *12*(4), 331–370.

Burke, R.D. (2007). Hybrid web recommender systems. In *The adaptive web, volume 4321 of LNCS* (pp. 377–408). Springer.

Chan, P.K. (2000). Constructing web user profiles: A non invasive learning approach. In *Web usage analysis and user profiling, volume 1836 of LNCS* (pp. 39–55). Springer.

Claypool, M., Gokhale, A., Miranda, T., Murnikov, P., Netes, D., Sartin, M. (1999). Combining content-based and collaborative filters in an online newspaper. In *Proceedings of ACM SIGIR work on recommender systems* (vol. 60). ACM.

de Campos, L.M., Fernńdez-Luna, J.M., Huete, J.F., Rueda-Morales, M.A. (2010). Combining content-based and collaborative recommendations: A hybrid approach based on Bayesian networks. *International Journal of Approximate Reasoning*, *51*(7), 785–799.

Gunawardana, A., & Meek, C. (2009). A unified approach to building hybrid recommender systems. In *Proceedings of the 3rd ACM conference on recommender systems* (pp. 117–124). ACM.

Herlocker, J., Konstan, J.A., Riedl, J. (2002). An empirical analysis of design choices in neighborhood-based collaborative filtering algorithms. *Information Retrieval*, *5*(2), 287–310.

Herlocker, J.L., Konstan, J.A., Borchers, A., Riedl, J. (1999). An algorithmic framework for performing collaborative filtering. In *Proceedings of the 22nd annual international ACM SIGIR conference on research and development in information retrieval, SIGIR '99* (pp. 230–237). New York: ACM.

Hofmann, T. (2004). Latent semantic models for collaborative filtering. *ACM Transaction on Information Systems*, *22*(1), 89–115.

Jain, A. K., Murty, M.N., Flynn, P.J. (1999). Data clustering: A review. *ACM Computer Survey*, *31*, 264–323.

Karypis, G. (2001). Evaluation of item-based top-N recommendation algorithms. In *Proceedings of the 10th international conference on information and knowledge management, CIKM '01* (pp. 247–254). New York: ACM.

Konstan, J., & Riedl, J. (2012). Recommender systems: From algorithms to user experience. *User Modeling and User-Adapted Interaction*, *22*(1), 101–123.

Linden, G., Smith, B., York, J. (2003). Amazon.com recommendations. Item-to-item collaborative filtering. *IEEE Internet Computing*, *7*(1), 76–80.

Liu, D.R., & Liou, C.H. (2011). Mobile commerce product recommendations based on hybrid multiple channels. *Electronic Commerce Research and Applications*, *10*(1), 94–104.

Lops, P., Gemmis, M., Semeraro, G. (2011). Content-based recommender systems: State of the art and trends. In *Recommender systems handbook* (pp.73–105). Springer.

Manouselis, N., & Costopoulou, C. (2007). Analysis and classification of multi-criteria recommender systems. *World Wide Web*, *10*(4), 415–441.

Melville, P., Mooney, R.J., Nagarajan, R. (2002). Content-boosted collaborative filtering for improved recommendations. In *Proceedings of the 18th national conference on artificial intelligence* (pp. 187–192). Edmonton: AAAI/IAAI.

Mitchell, T.M. (1997). Machine learning. Burr Ridge, IL: McGraw Hill, 45.

Miyahara, K., & Pazzani, M.J. (2000). Collaborative filtering with the simple Bayesian classifier. In *Proceedings of the 6th pacific rim international conference on artificial intelligence, PRICAI'00* (pp. 679–689). Berlin: Springer-Verlag.

Mobasher, B., Dai, H., Luo, T., Nakagawa, M. (2002). Discovery and evaluation of aggregate usage profiles for web personalization. *Data Mining Knowledge Discovery*, *6*, 61–82.

Montaner, M., Lopez, B., de la Rosa, J.L. (2004). A taxonomy of recommender agents on the internet. *Journal on Web Semantics (JWS)*, *19*(4), 285–330.

Rosaci, D., Palopoli, L., Sarné, G.M.L. (2013). Introducing specialization in e-commerce recommender systems. *Concurrent Engineering: Research and Applications*, *21*(3), 187–196.

Parsons, J., Ralph, P., Gallagher, K. (2004). Using viewing time to infer user preference in recommender systems. In *AAAI workshop on semantic web personalization* (pp. 52–64). Menlo Park: AAAI Press.

Pazzani, M.J. (1999). A framework for collaborative, content-based and demographic filtering. *Artificial Intelligence Review*, *13*(5-6), 393–408.

Popescul, A., Pennock, D.M., Lawrence, S. (2001). Probabilistic models for unified collaborative and content-based recommendation in sparse-data environments. In *Proceedings of the 17th conference on uncertainty in artificial intelligence*. Morgan Kaufmann Pub.

Postorino, M.N., & Sarné, G.M.L. (2011). A neural network hybrid recommender system. In *Proceedings of the 2011 conference on neural Nets WIRN10* (pp. 180–187). Amsterdam: IOS Press.

Rosaci, D., & Sarnè, G.M.L. (2006). A multi-agent system handling user and device adaptivity of web sites. *User Modeling and User-Adapted Interaction*, *16*(5), 435–462.

Rosaci, D., & Sarnè, G.M.L. (2010). Efficient personalization of e-learning activities using a multi-device decentralized recommender system. *Computational Intelligence*, *26*(2), 121–141.

Rosaci, D., Sarnè, G.M.L., Garruzzo, S. (2009). MUADDIB: A distributed recommender system supporting device adaptivity. *ACM Transansacion on Information Systems*, *27*(4).

Rosaci, D., Sarnè, G.M.L., Garruzzo, S. (2012). Integrating trust measures in multiagent systems. *International Journal of Intelligent Systems*, *27*(1), 1–15.

Rosaci, D., & Sarnè, G.M.L. (2013). Cloning mechanisms to improve agent performances. *Journal of Network and Computer Applications*, *36*(1), 402–408.

Salter, J., & Antonopoulos, N. (2006). CinemaScreen recommender agent: Combining collaborative and content-based filtering. *IEEE Intelligent Systems*, *21*(1), 35–41.

Schein, A.I., Popescul, A., Ungar, L.H., Pennock, D.M. (2002). Methods and metrics for cold-start recommendations. In *Proceedings of the 25th Annual International ACM SIGIR conference on research and development in information retrieval, SIGIR '02* (pp. 253–260). ACM.

Shardanand, U., & Maes, P. (1995). Social information filtering: algorithms for automating "word of mouth". In *Proceedings of the SIGCHI conference on human factors in computing systems* (pp. 210–217). ACM.

Stiller, C., Ross, F., Ament, C. (2010). Demographic recommendations for WEITBLICK, an assistance system for elderly. In *International symposium on communications and information technologies (ISCIT)* (pp. 406–411). IEEE.

Su, X., & Khoshgoftaar, T.M. (2009). A survey of collaborative filtering techniques. *Advances in Artificial Intelligence*, *2009*, 4:2-4:2.

Trewin, S. (2000). Knowledge-based recommender systems. *Encyclopedia of Library and Information Science*, *69*(Supplement 32), 69.

van Rijsbergen, C.J. (1974). Foundation of evaluation. *Journal of Documentation*, *30*(4), 365–373.

Wei, K., Huang, J., Fu, S. (2007). A survey of e-commerce recommender systems. In *Proceedings of the 13th international conference on service systems and service management* (pp. 1–5). Washington: IEEE Computer Society.

Weng, S.S., & Liu, M.J. (2004). Feature-based recommendations for one-to-one marketing. *Expert Systems with Applications*, *26*, 493–508.

Woźniak, M., Graña, M., Corchado, E. (2014). A survey of multiple classifier systems as hybrid systems. *Information Fusion*, *16*, 3–17.