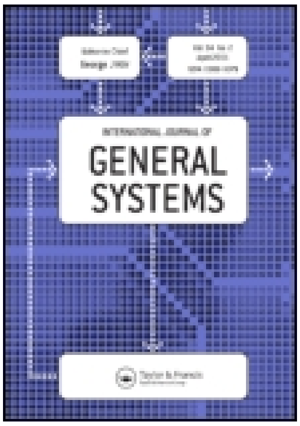

International Journal of General Systems
Publication details, including instructions for authors and subscription information:
http://www.tandfonline.com/loi/ggen20

Incremental collaborative filtering based on Mahalanobis distance and fuzzy membership for recommender systems

Maytiyanin Komkhao [a] , Jie Lu [b] , Zhong Li [a] & Wolfgang A. Halang [a]

[a] Chair of Computer Engineering, Fernuniversität in Hagen, Hagen, Germany

[b] Decision Systems and e-Service Intelligence Lab, Faculty of Engineering and Information Technology, School of Software, Centre for Quantum Computation and Intelligent Systems, University of Technology Sydney, Sydney, Australia

Version of record first published: 27 Jul 2012.

To cite this article: Maytiyanin Komkhao , Jie Lu , Zhong Li & Wolfgang A. Halang (2013): Incremental collaborative filtering based on Mahalanobis distance and fuzzy membership for recommender systems, International Journal of General Systems, 42:1, 41-66

To link to this article: http://dx.doi.org/10.1080/03081079.2012.710437

International Journal of General Systems
Vol. 42, No. 1, January 2013, 41–66

Incremental collaborative filtering based on Mahalanobis distance and fuzzy membership for recommender systems

Maytiyanin Komkhao[a]*, Jie Lu[b], Zhong Li[a] and Wolfgang A. Halang[a]

[a]*Chair of Computer Engineering, Fernuniversität in Hagen, Hagen, Germany;* [b]*Decision Systems and e-Service Intelligence Lab, Faculty of Engineering and Information Technology, School of Software, Centre for Quantum Computation and Intelligent Systems, University of Technology Sydney, Sydney, Australia*

(*Received 9 December 2011; final version received 22 June 2012*)

Recommender systems, as an effective personalization approach, can suggest best-suited items (products or services) to particular users based on their explicit and implicit preferences by applying information filtering technology. Collaborative filtering (CF) method is currently the most popular and widely adopted recommendation approach. It works by collecting user ratings for items in a given domain and by computing the similarity between the profiles of several users in order to recommend items. Current similarity measures and models updated by traditional model-based CF have, however, shortcomings with respect to accuracy of prediction and scalability of recommender systems. To overcome these problems, here an incremental CF algorithm based on the Mahalanobis distance is presented. The algorithm has two phases: *the learning phase*, in which models of similar users are constructed incrementally, and *the prediction phase*, in which interested users are clustered by measuring their similarity to existing clusters in a model. To handle confusion of decision making on overlapping clusters, fuzzy sets are employed, and the degree of membership to them is expressed by the Mahalanobis radial basis function. Experimental results demonstrate that the proposed algorithm leads to improved prediction accuracy and prevents the scalability problem in recommendation systems.

Keywords: recommender systems; collaborative filtering; learning algorithms; Mahalanobis distance; fuzzy sets

1. Introduction

At present, the enormously increasing amount of data available over the Internet makes it very hard for human perception to analyse the vast amount of interesting information and to determine products and services that we really need (Lu, Shambour and Zhang 2009; Lu, Shambour, Xu, Lin and Zhang 2010). Recommender systems are effective methods to suggest interesting items according to user needs, and provide personalized e-services (Lu *et al.* 2009, 2010; Shambour and Lu 2011). They have established themselves widely in e-commerce and e-business applications (Cornelis, Lu, Guo and Zhang 2007; Lee, Cho and Kim 2010) in recent years. Recommender systems work on the basis of two major techniques (Lekakos and Giaglis 2006), viz. content-based and collaborative filtering (CF) as well as hybrid methods combining them with each other or with other techniques (Burke 2002; Lekakos and Giaglis 2006; Candillier, Meyer and Boullé 2007).

*Corresponding author. Email: maytiyanin.komkhao@fernuni-hagen.de

ISSN 0308-1079 print/ISSN 1563-5104 online
© 2012 Taylor & Francis
http://dx.doi.org/10.1080/03081079.2012.710437
http://www.tandfonline.com

In content-based filtering, taking the recommendation of new movies to an interested user as an example, the recommendation is calculated based on matching new movies' profiles with all movies' profiles of the interested user's preferences in the past, and include actor, director, and title of a movie (Lekakos and Giaglis 2006; Lu *et al.* 2010). Content-based filtering is mostly applied to text-based documents that are difficult for computers to interpret (Adomavicius and Tuzhilin 2005).

As an efficient technique, CF is widely employed in recommender systems (Sarwar Karypis, Konstan and Riedl 2000; Burke 2002; Cho and Kim 2004; Herlocker, Konstan, Terveen and Riedl 2004). There are two major versions of CF, viz. memory-based and model-based CF (Adomavicius and Tuzhilin 2005; Su and Khoshgoftaar 2009; Pham, Cao, Klamma and Jarke 2011). For both, there is a user-based approach (based on groups of users) and an item-based approach (based on groups of items) (Shambour and Lu 2011). Memory-based CF is based on entire matrices of previous user-item ratings. To avoid extensive searching in these rating matrices, model-based CF builds models by grouping similar users into clusters (Adomavicius and Tuzhilin 2005; Xue *et al.* 2005). To determine a similarity, the Pearson correlation coefficient (Su and Khoshgoftaar 2009) is widely used. It measures the linear correlation between two vectors of ratings (Herlocker *et al.* 2004; Adomavicius and Tuzhilin 2005; Gong 2010; Pham *et al.* 2011). Even though memory-based CF techniques are quite easy to implement in many real-world applications (Pham *et al.* 2011), the main problems are sparsity, scalability, and handling of new users (Adomavicius and Tuzhilin 2005; Xue *et al.* 2005; Lekakos and Giaglis 2006; Pham *et al.* 2011). With sparsity problem, it is meant that memory-based CF is not computationally feasible to compute the similarity of users when there are only a few users to rate a small set of items (Adomavicius and Tuzhilin 2005; Pham *et al.* 2011). A new user has no ratings when being joined into a recommender system. Therefore, similarities cannot be computed and predictions for recommendations cannot be made (Lekakos and Giaglis 2006), giving rise to the new user problem. Finally, the scalability problem arises when in real-world applications for any newly occurring user or item the entire user-item rating matrix has to be searched in full (Xue *et al.* 2005; Su and Khoshgoftaar 2009; Pham *et al.* 2011).

In solving the problems of memory-based CF, we orientate at the advantages of model-based CF, viz. (i) higher accuracy of prediction (Hofmann 2003; Pham *et al.* 2011), and (ii) no need for searching the whole user-item rating matrix when grouping users into models (Sarwar, Karypis, Konstan and Riedl 2002a; Hofmann 2003). Model-based CF techniques build models by grouping similar users into clusters based on a rating matrix. Design and creation of models are based on statistical and machine learning techniques. An example for the former is probabilistic latent semantic analysis (Hofmann 2003), and an example for the latter is clustering (Sarwar *et al.* 2002a; Candillier *et al.* 2007; Pham *et al.* 2011). Models constructed by these techniques are used to predict interested users' ratings of particular items (Cho and Kim 2004; Adomavicius and Tuzhilin 2005; Lekakos and Giaglis 2006).

To ease learning on the basis of large databases, incremental learning techniques have been proposed that build models continuously, leading to incremental improvements of recommendations. In doing so, it is not necessary to calculate the similarity values for all items previously rated by the users, because a model represents a number of similar users (Sarwar *et al.* 2002a; Adomavicius and Tuzhilin 2005).

In principle, hybrid methods combining content-based and CF to eliminate the drawbacks of each technique and to improve performance lead to advanced recommender systems (Adomavicius and Tuzhilin 2005; Lu *et al.* 2010). The main shortcoming of hybrid methods, however, is increased implementation complexity and expense (Su and

Khoshgoftaar 2009). Many researchers have proposed different techniques to improve accuracy and efficiency of existing CF techniques. Combined CF techniques have been studied, in which machine learning techniques are applied, such as techniques based on artificial neural networks or fuzzy set theory (Cornelis *et al.* 2007). In addition, the set of input vectors (rating profiles) is selected carefully in learning systems. Data reduction techniques, such as principle component analysis and singular value decomposition (SVD; Papagelis, Rousidis, Plexousakis and Theoharopoulos 2005), can be applied to reduce data-set sizes. Feature subset selection, a strategy of data reduction, is concerned with reducing the complexity of calculation, which not only decreases data-set size, but also removes redundant attributes or dimensions that are weakly relevant only (Han and Kamber 2006).

This paper focuses on model-based CF techniques aiming to increase their efficiency and reduce their complexity. In particular, we use the Mahalanobis distance (Mahalanobis 1936) as similarity measure and propose an incremental CF (InCF) algorithm based on it. The InCF algorithm uses a user-item ratings matrix to build models that rely on the similarity between given user ratings (user-to-user similarity computation) in order to predict an interested user's ratings of particular items.

We also focus on two main challenges, namely accuracy of prediction and scalability. To improve prediction accuracy, we apply the Mahalanobis distance (Mahalanobis 1936) instead of the Euclidean distance (Adomavicius and Tuzhilin 2005; Su and Khoshgoftaar 2009) in model-based CF. The Mahalanobis distance has proven useful (Lekakos and Giaglis 2006) to determine the decision boundaries of clusters (Tanner and Cruickshank 1998; Xu and Vuskovic 2004). Moreover, the InCF algorithm employs the Mahalanobis radial basis function to express fuzzy membership in overlapping clusters. The assumption is, however, that in many real-world data-sets, a group of clusters tends to be elongated and sized differently.

To handle the scalability challenge, InCF is designed to predict the ratings of particular items for interested users incrementally, in order to be able to cope with the tremendous growth of both items and interested users present in the web (Papagelis *et al.* 2005). Redundant attributes will be removed in the user-item rating matrix, and correlation-based feature subset selection (CFS; Hall 2000; Witten and Frank 2005) will be used to reduce the data-sets in size and to remove irrelevant attributes.

The paper is organized as follows. Section 2 introduces preliminaries of this study and related work. In Section 3, the proposed algorithm is detailed. The results of empirical validation are given in Section 4. Conclusions and directions for future study are presented in Section 5.

2. Preliminary and related work

This section reviews the current research on three areas related to this study: similarity measures, clustering algorithms, and incremental learning in CF algorithms.

2.1 *Similarity measures*

To identify the similarity of users, similarity measures are employed. The relationship between dissimilarity and similarity is given by $S_{ij} = 1 - \delta_{ij}$, where S_{ij} is the similarity between users i and j, and δ_{ij} is the dissimilarity between users i and j (Han and Kamber 2006). Similar users are grouped into the same cluster, and dissimilar users are grouped into different clusters (Han and Kamber 2006).

Distances are used to represent similarity and dissimilarity between users. If the ratings of two users are most similar, then the distance between them is minimized; if the ratings are more dissimilar, then the distance between the users is maximized. Let d_{ij} be the distance between features i and j, then d_{ij} is always positive or zero and symmetric, i.e. $d_{ij} = d_{ji}$ (Han and Kamber 2006).

There are many ways to measure similarity and dissimilarity between users based on feature types or on quantitative, ordinal, binary, and nominal variables. In this paper, we focus on quantitative variables such as the Euclidean, Manhattan, and Mahalanobis distance.

For quantitative variables, most existing algorithms use the Euclidean and Manhattan distances as proximity functions, because they are easy to implement (Candillier *et al.* 2007; Su and Khoshgoftaar 2009). Applying a similarity measure in model-based CF, many researchers have focused on different kinds of similarity measures. In Candillier *et al.* (2007), for instance, normalized Manhattan and Euclidean distances were used to define groups in model-based CF.

For many real data-sets, clusters of data tend to be elongated and sized differently. Here, the Mahalanobis distance (Mahalanobis 1936) turns out to be more suitable than the Euclidean distance, which is the most popular similarity measure. The Mahalanobis distance considers the orientation of data in the sample space, and groups them by mean (called prototype in this paper) and covariance matrices (Dempster 1972). The shapes of clusters are hyper-ellipsoidal rather than hyper-spherical as obtained when working with the Euclidean distance. The difference between them is described in mathematical terms as follows (Mahalanobis 1936; Dempster 1972).

The jth user is specified by an attribute vector

$$\mathbf{x}_j = \begin{bmatrix} x_{1,j} \\ x_{2,j} \\ \vdots \\ x_{n,j} \end{bmatrix}, \quad j = 1, \dots, N \tag{1}$$

with $n > 0$ being the number of attributes and N the total number of users. The mean of these vectors is given by

$$\bar{\mathbf{x}} = \begin{bmatrix} \bar{x}_1 \\ \vdots \\ \bar{x}_n \end{bmatrix} \quad \text{with}\, \bar{x}_i = \frac{1}{N}\sum_{j=1}^{N} x_{i,j}. \tag{2}$$

No, they should stay as they are. However, there was a lack of consistency throughout the paper, which I have corrected now. Then, between two vectors $\mathbf{x}_j$ and $\mathbf{x}_s$, the Manhattan distance ($\mathbf{dist}_C$) (Han and Kamber 2006; Su and Khoshgoftaar 2009) is

$$\mathbf{dist}_C(j, s) = |\mathbf{x}_j - \mathbf{x}_s|, \tag{3}$$

the Euclidean distance ($\mathbf{dist}_E$) (Han and Kamber 2006; Su and Khoshgoftaar 2009) is

$$\mathbf{dist}_E(j, s) = [(\mathbf{x}_j - \mathbf{x}_s)^{\mathrm{T}}(\mathbf{x}_j - \mathbf{x}_s)]^{1/2}, \tag{4}$$

and the Mahalanobis distance ($\mathbf{dist}_M$) (Mahalanobis 1936) is

$$\mathbf{dist}_M(j,s) = [(\mathbf{x}_j - \mathbf{x}_s)^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{x}_j - \mathbf{x}_s)]^{1/2}, \tag{5}$$

where the users' covariance matrix

$$\mathbf{K} = \begin{bmatrix} k_{1,1} & k_{1,2} & \dots & k_{1,n} \\ k_{2,1} & k_{2,2} & \dots & k_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ k_{m,1} & k_{m,2} & \dots & k_{m,n} \end{bmatrix} \tag{6}$$

is calculated as

$$\mathbf{K} = \frac{1}{N-1}\sum_{j=1}^{N}(\mathbf{x}_j - \bar{\mathbf{x}})^{\mathrm{T}}(\mathbf{x}_j - \bar{\mathbf{x}}). \tag{7}$$

2.2 *Clustering algorithms*

For setting up models in model-based CF, clustering is an intermediate step. The clusters generated by clustering algorithms are used for further analyses and prediction purposes (Su and Khoshgoftaar 2009). Clustering has been successfully used in several exploratory pattern analyses, data mining, machine learning, and in pattern classifications (Jain, Murthy and Flynn 1999). The effect of clustering is that users are grouped in such a way that variables within the same cluster have high similarity to one another, but are very dissimilar to variables in other clusters (Han and Kamber 2006). There are several clustering algorithms employed for model-based CF such as K-nearest neighbours (K-NN), hierarchical, density-based, and K-means clustering (Candillier *et al.* 2007; Su and Khoshgoftaar 2009).

In our experimental comparison, K-means and K-NN are used as part of model creations, as they are the simplest methods for pattern classification based on similarity measurement (e.g. using the Euclidean distance) to compute distances between users.

K-means consists of the following steps (Jain *et al.* 1999; Witten and Frank 2005; Han and Kamber 2006):

Step 1: initialize randomly h clusters from the data-set.
Step 2: variables from the data-set are assigned to each cluster by calculating the similarity values between centres and variables.
Step 3: iteratively process cluster reassignments until convergence is reached. Generally, the Euclidean distance is used to calculate the similarity values.
Step 4: if a convergence criterion is not met, go to Step 2.

K-NN consists of two steps (Sarwar *et al.* 2002a):

Step 1: set the number of neighbourhoods k; input a set of vectors, whose number is a multiple of k, and partition it into k neighbourhoods of equal size, e.g. by clustering.
Step 2a: for a new vector, calculate its distances to all vectors in the neighbourhoods.
Step 2b: determine the vector's k nearest neighbours.
Step 2c: the vector is assigned to the neighbourhood in which lie most of the k closest neighbours (Witten and Frank 2005).

Candillier *et al.* (2007) compared several clustering algorithms commonly employed in CF such as K-means, locally adaptive clustering, and statistical subspace clustering (Candillier, Tellier, Torre and Bousquet 2005). The simulation results show that K-means performs better than others, but affects a smaller number of clusters. The algorithms compared in Candillier *et al.* (2007) lack, however, the ability to continuously learn in real-world applications, where new users and new items are added incrementally. To build models, on-line learning should be used instead of batch (off-line) learning.

Incremental learning has been applied in machine learning (Sarwar *et al.* 2000; Polikar, DePasquale, Mohammed, Brown and Kuncheva 2010) and recommender systems (Sarwar, Karypis, Konstan and Riedl 2002b; Papagelis *et al.* 2005) (see more details in Section 2.3). Based on the Mahalanobis distance, it has been used to create models continuously without examining entire data-sets. Rating profiles of interested users not rated before are derived from models as built with the techniques above.

2.3 *Incremental learning in CF algorithms*

In traditional CF, first a clustering algorithm is applied on the user-item rating matrix to create a model in off-line mode as detailed in Section 2.2 and, then, with the model of the interested users' ratings the ones of new users are predicted. This approach faces, however, the problem of model re-computations whose expense grows polynomially with the number of users and items (Papagelis *et al.* 2005) in the real-world data-set. To cope with the scalability problem under the constraints of fast prediction making and high recommendation quality (accuracy) (Sarwar *et al.* 2002a; Papagelis *et al.* 2005; Khoshneshin and Street 2010), several learning techniques based on incremental updates have been utilized in CF algorithms when new users and items arrive in streams.

Papagelis *et al.* (2005) proposed ICF (memory-based) using user-to-user similarity computation. The focus is on re-computing similarity values of an interested user when submitting a new rating. The Pearson correlation similarity measure is split into three factors, and then new values of each factor are calculated and finally combined into the new similarity value.

Khoshneshin and Street (2010) proposed incremental (model-based) CF via evolutionary co-clustering, which is the process of updating rows and columns at the same time, using both user-to-a-user and item-to-an-item clusters. The results show that the accuracy of this approach is better than that of earlier applied methods.

Sarwar *et al.* (2002b) proposed an incremental SVD algorithm in model-based CF. A model is created by user-to-user similarity computation using folding-in, which means that a new user joins a model with an already reduced user-item matrix. Using the SVD-based prediction algorithm, existing users and items are not affected by new users or items. When a new user with a set of profile ratings is added to a database, it is not necessary, however, to re-compute the whole database. For Sarwar's algorithm, it was shown that it is able to quickly produce high-quality recommendations at the expense of considerable matrix computations.

Sarwar *et al.* (2002a) also proposed recommender systems for large-scale e-commerce. To create a model on the basis of user-to-user similarity computation, first a clustering algorithm is applied on the user-item rating matrix in order to obtain partitions. Then, the partition where a new user belongs to is examined by using traditional memory-based CF to make pedictions.

In the research literature, many examples of employing incremental learning in CF can be found to improve the performance of CF algorithms, as discussed above. There are two important issues, however, that should be considered in InCF, namely similarity measures in clustering algorithms and the decision boundaries in identifying clusters when creating models. For the former, incremental similarity updates have been considered in Papagelis *et al.* (2005), but there is no relation to new users at all as the focus is just on updating ratings for existing users. In the latter, in order to effectively create clusters of users, the decision boundaries of clusters should be clearly identified to enable the best fit with each user.

3. An InCF algorithm based on the Mahalanobis distance

This section explains in detail the InCF algorithm, the constituting learning, and prediction algorithms as well as related definitions and concepts.

3.1 *Definitions and concepts*

In the InCF algorithm, input vectors are considered in *the learning phase* and interested users in *the prediction phase*. The input vectors are joined one-by-one in *the learning phase* in order to create a model.

The set of input attribute vectors is denoted by $\mathbf{p} = \{\mathbf{p}_{\mathrm{L}}, \mathbf{p}_{\mathrm{U}}\}$, where $\mathbf{p}_{\mathrm{L}} = \{\mathbf{p}_{\mathrm{L}1}, \mathbf{p}_{\mathrm{L}2}, \ldots, \mathbf{p}_{\mathrm{L}k}\}$ is the set of k input vectors to *the learning phase* and $\mathbf{p}_{\mathrm{U}} = \{\mathbf{p}_{\mathrm{U}1}, \mathbf{p}_{\mathrm{U}2}, \ldots, \mathbf{p}_{\mathrm{U}k}\}$ is the set of h interested users in *the prediction phase*.

Definition 1 (Model). In *the learning phase*, the k input vectors are incrementally submitted to clustering, giving rise to prototypes of a model. The final prototype after k steps is the model sought. It consists of $m \leq k$ clusters of input vectors.

To determine the decision boundaries of clusters, normally standard membership functions such as Gaussian-type radial basis functions with hyper-spherical shapes are employed. Data-sets frequently occurring in real-world applications, however, are better mapped by clusters that tend to be elongated in certain dimensions. Therefore, to increase accuracy, the InCF algorithm employs the Mahalanobis radial basis function with hyper-ellipsoidal shape as distance measure used for clustering.

Definition 2 (Mahalanobis radial basis function (Tanner and Cruickshank 1998; Yen and Meesad 2001)). Let $\mathbf{p}$ be an input vector and $\mathbf{W}$ the centre of the ith cluster in the model, then $\mathbf{p}$'s degree of membership to the ith cluster $\mathbf{mem}(\mathbf{p},\mathbf{W})$ is calculated with the Mahalanobis distance

$$\mathbf{dist}_M = [(\mathbf{p} - \mathbf{W})^{\mathrm{T}} \mathbf{K}^{-1} (\mathbf{p} - \mathbf{W})]^{1/2} \tag{8}$$

using the Mahalanobis radial basis function:

$$\mathbf{mem}(\mathbf{p}, \mathbf{W}) = \mathbf{exp}\left(- \frac{(\mathbf{p} - \mathbf{W})^{\mathrm{T}} \mathbf{K}^{-1} (\mathbf{p} - \mathbf{W})}{2} \right). \tag{9}$$

In order to cope not only with non-spherical cluster shapes, but also with some further uncertainties about their forms, we employ fuzzy sets to determine decision boundaries. In this context, the following operation is needed.

Definition 3 (Fuzzy OR operation (∨) (Yen and Meesad 2001)). An extension of Boolean disjunction to fuzzy logic is defined as:

$$\text{winner} = \texttt{argmax}_i(\mathbf{mem}_i), \tag{10}$$

$$\text{winner} = \mathbf{mem}_1 \vee \mathbf{mem}_2 \vee \cdots \vee \mathbf{mem}_n, \tag{11}$$

where $\mathbf{mem}_1 \vee \mathbf{mem}_2 = \mathbf{mem}_1$, if $\mathbf{mem}_1 > \mathbf{mem}_2$, $\mathbf{mem}_1 \vee \mathbf{mem}_2 = \mathbf{mem}_2$, if $\mathbf{mem}_1 < \mathbf{mem}_2$.

Definition 4 (Winner cluster (called winner node in (Shambour and Lu 2011)). The winner cluster is defined as the cluster whose centre matches an input vector best, as expressed by the highest degree of membership assumed.

Definition 5 (Learning rules) (Yen and Meesad 2001; Vuskovic and Sijiang 2002). Learning rules are used to update the clusters of the model's prototypes in *the learning phase* (Section 3.3), in particular (12) to increment the prototype count as well as (13) to update the winner cluster and (14) the covariance matrix in case the input vector considered in the learning step does not give rise to the creation of a new cluster:

$$\mathbf{C}_{i,\text{new}} = \mathbf{C}_{i,\text{old}} + 1, \tag{12}$$

$$\mathbf{w}_{i,\text{new}} = (1-\beta)\mathbf{w}_{i,\text{old}} + \beta\mathbf{p}, \tag{13}$$

$$\mathbf{K}_{i,\text{new}} = (1-\beta)\mathbf{K}_{i,\text{old}} + \beta(1-\beta^2)(\mathbf{p}-\mathbf{w}_{i,\text{new}})(\mathbf{p}-\mathbf{w}_{i,\text{new}})^{\mathrm{T}}, \tag{14}$$

where the subscript 'new'/'old' denotes the model's prototype after/before updating, $\mathbf{C}_i$ counts the number of patterns combined into the ith cluster, $\mathbf{w}_i$ is the centre of the ith cluster, $\beta = 1/\mathbf{C}_{i,\text{new}}$, and $\mathbf{K}_i$ represents the covariance matrix at the ith cluster (Yen and Meesad 2001; Vuskovic and Sijiang 2002).

3.2 *Structure of the InCF algorithm*

The InCF algorithm creates models, in the form of sets of clusters, from the user-rating matrix representing groups of similar input vectors. It uses the Mahalanobis distance and Mahalanobis radial basis function to identify that a cluster is the best match with an input vector, which is called the winner cluster. The best match is defined by the degree of membership based on the similarity distance. The algorithm compares input vectors joining the system with known clusters. The incremental learning algorithm either associates an input vector with an existing cluster as target cluster or includes it into the model as a new cluster.

The algorithm is constructed according to the idea and definitions mentioned. Figure 1 presents its structure. The InCF algorithm has two main phases, viz. *the learning phase* and *the prediction phase*. *The initialization phase* is utilized to first preprocess the data-set (in case it is complex).

The initialization phase consists of three parts:

(1) *Input*: preprocessing of the user-item rating matrix for the feature extraction part.
(2) *Feature extraction*: CFS is utilized to select relevant input vectors from the user-item rating matrix in order to reduce the calculation complexity and to find relevant attributes.
(3) *Output*: the set of input vectors.

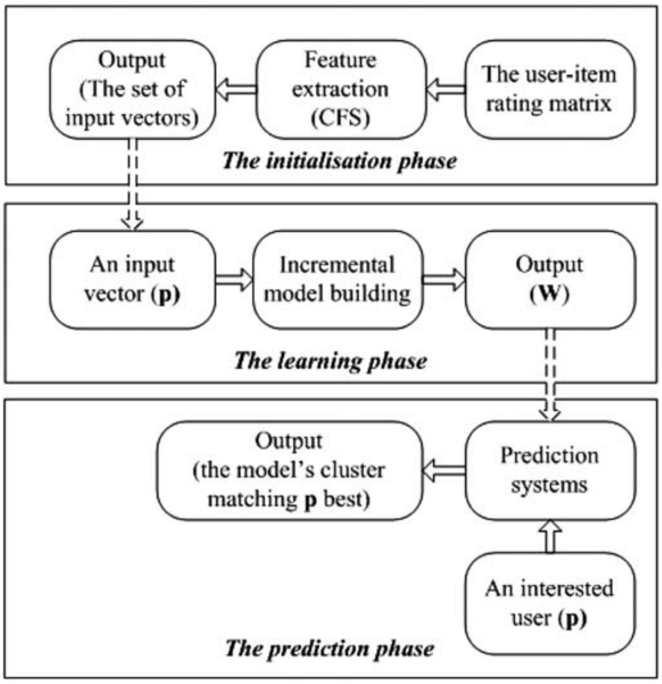
Output (The set of input vectors)
Feature extraction (CFS)
The user-item rating matrix
The initialisation phase
An input vector (**p**)
Incremental model building
Output (**W**)
The learning phase
Output (the model's cluster matching **p** best)
Prediction systems
An interested user (**p**)
The prediction phase

Figure 1. Structure of the InCF algorithm.

The learning phase consists of three parts in a loop:

(1) *Input*: an input vector (**p**) from the set of input vectors prepared in *the initialization phase* is submitted to model building.
(2) *Incremental model building*: the Mahalanobis distance radial basis function is applied to determine which cluster is the best match with an input vector.
(3) *Output*: a model prototype is obtained by updating the winner cluster determined in Step 2 or by creating an additional cluster.

The prototype generated in the last execution of *the learning phase* constitutes the model **W** sought. *The prediction phase* consists of three major parts:

(1) *Input*: an interested user (**p**) joins the system.
(2) *Prediction systems*: for classification, the Mahalanobis distance radial basis function determines which cluster of the model **W** is the best match with **p**.
(3) *Output*: the model's cluster matching **p** best.

3.3 *Learning algorithm in InCF*

The algorithm InCF consists of two major parts: the learning algorithm and the prediction algorithm. The learning algorithm aims to construct models by using input vectors provided by *the initialization phase*. The prediction algorithm is utilized to classify interested users. To create a model incrementally, input vectors are combined one-by-one with model prototypes. The model arises as a result of this prototyping process. Figure 2 depicts the steps of InCF's learning algorithm whose details are presented below.

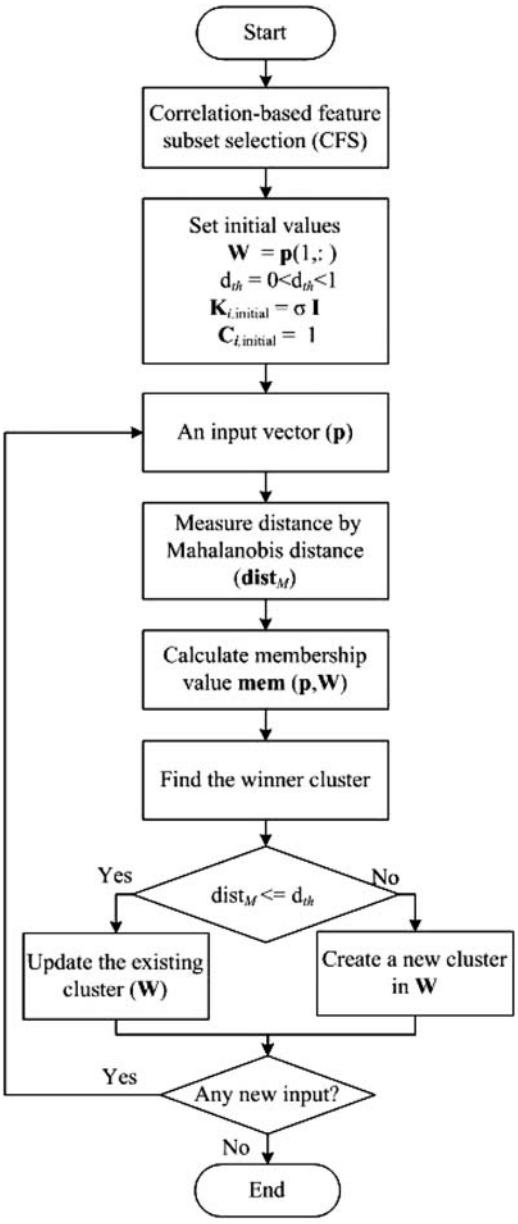
Start
Correlation-based feature subset selection (CFS)
Set initial values
$\mathbf{W} = \mathbf{p}(1,:)$
$d_{th} = 0<d_{th}<1$
$\mathbf{K}_{i,\text{initial}} = \sigma\,\mathbf{I}$
$\mathbf{C}_{i,\text{initial}} = 1$
An input vector (**p**)
Measure distance by Mahalanobis distance (**dist**$_M$)
Calculate membership value **mem (p,W)**
Find the winner cluster
$\text{dist}_M <= d_{th}$
Yes
No
Update the existing cluster (**W**)
Create a new cluster in **W**
Any new input?
Yes
No
End

Figure 2. Steps of the learning algorithm in InCF.

3.3.1 *Description of learning algorithm in InCF*

Step 0: Apply CFS on the user-item rating matrix.
Step 1: Let the first input vector **p** be the cluster forming the initial prototype of the model **W**. Set the initial variables as follows: a counter $\mathbf{C}_{i,\text{initial}} = 1$, the distance threshold $0 < d_{\text{th}} < 1$, and the covariance matrix ($\mathbf{K}_{\text{initial}}$):

$$\mathbf{K}_{\text{initial}} = \sigma \mathbf{I}_n, \tag{15}$$

where

$$\mathbf{I}_n = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \vdots & \vdots & \ddots & \vdots \\ 0 & 0 & \cdots & 1 \end{bmatrix} \quad \text{and} \quad 0 < \sigma < 1.$$

Step 2: read the next **p**.
Step 3: Calculate Mahalanobis distance ($\mathbf{dist}_M$) values between **p** and the clusters in **W** using the non-singular covariance matrix. Here, $\sigma\mathbf{I}$ is added to **K** in order to prevent the problem that the matrix becomes singular:

$$\mathbf{K}_{\text{new}} = \mathbf{K}_{\text{old}} + \sigma\mathbf{I}_n. \tag{16}$$

Step 4: calculate the membership value (find soft decision) for each cluster using the Mahalanobis radial basis function

$$\mathbf{mem}(\mathbf{p}, \mathbf{W}) = \mathbf{exp}\left(-\frac{(\mathbf{p}-\mathbf{W})^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{p}-\mathbf{W})}{2}\right). \tag{17}$$

Step 5: select the winner cluster as the one for which the maximum membership value identified by the fuzzy OR operator (i.e. max operator) is assumed:

$$\text{winner} = \texttt{argmax}_i(\mathbf{mem}_i) = \mathbf{mem}_1 \vee \mathbf{mem}_2 \vee \cdots \vee \mathbf{mem}_n. \tag{18}$$

Step 6: update the cluster variables count, model prototype, and covariance matrix by (12)–(14):
  if the distance between input vector and winner cluster does not exceed a given threshold, then update the cluster's variables;
  if the distance between input vector and winner cluster exceeds this threshold, a new cluster is created.
Step 7: if there is a new input vector **p**, go to Step 2.

3.3.2 *Example of creating a model by the learning algorithm of InCF*

As shown in Table 1, three users have rated items from dislike to like, 1–5, and 0 means that users have never given a rating. The three input vectors are processed in *the learning phase* (without CFS in *the initialization phase*).

By our learning algorithm in InCF, we have

Step 0: according to the small rating matrix in the example, it is not necessary to use CFS to reduce the size of the rating matrix.

Table 1. An artificial user-item rating matrix.

	Item 1	*Item 2*	*Item 3*	*Item 4*	*Item 5*
User 1	2	3	0	3	2
User 2	0	2	0	0	2
User 3	2	2	0	3	2

Step 1: set the first input vector $\mathbf{p}_1$ to be the only cluster in the model's first prototype $\mathbf{W}$

$$\mathbf{p}_1 = \mathbf{w}_1 = \begin{bmatrix} 2 \\ 3 \\ 0 \\ 3 \\ 2 \end{bmatrix}.$$

Here, σ, d_{th}, and $\mathbf{K}_{i,\text{initial}}$ are initialized as follows: $\sigma = 0.99$, $d_{\text{th}} = 0.5$, and

$$\mathbf{K}_{i,\text{initial}} = \begin{bmatrix} 0.99 & 0 & 0 & 0 & 0 \\ 0 & 0.99 & 0 & 0 & 0 \\ 0 & 0 & 0.99 & 0 & 0 \\ 0 & 0 & 0 & 0.99 & 0 \\ 0 & 0 & 0 & 0 & 0.99 \end{bmatrix},$$

then, $\mathbf{K}_i = \mathbf{K}_{i,\text{initial}} + 0.99\mathbf{I}_n$.
Step 2: read $\mathbf{p}_2$

$$\mathbf{p}_2 = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}.$$

Step 3: similarity between the cluster ($\mathbf{w}_1$) and the user $\mathbf{p}_2$ is calculated by

$$\mathbf{dist}_M(\mathbf{p}_2, \mathbf{w}_1) = \left[\left(\begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \\ 0 \\ 3 \\ 2 \end{bmatrix} \right)^{\mathrm{T}} \mathbf{K}^{-1} \left(\begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \\ 0 \\ 3 \\ 2 \end{bmatrix} \right) \right]^{1/2} = 2.6458.$$

Step 4: the membership value is computed as

$$\mathbf{mem}(\mathbf{p}_2, \mathbf{w}_1) = \mathbf{exp}\left(-\frac{2.6458}{2} \right) = 0.2664.$$

Step 5: The winner cluster is ($\mathbf{w}_1$).
Step 6: since for the distance $\mathbf{dist}_M(\mathbf{p}_2, \mathbf{w}_1) > d_{\text{th}}$, the cluster variables are updated with

(12)–(14), and the new cluster $\mathbf{w}_2 = \begin{bmatrix} 0 & 2 & 0 & 0 & 2 \end{bmatrix}^T$ is joined to the model:

$$\mathbf{W} = \begin{bmatrix} 2 & 0 \\ 3 & 2 \\ 0 & 0 \\ 3 & 0 \\ 2 & 2 \end{bmatrix}.$$

Step 7: read $\mathbf{p}_3$.
Step 8: as in Step 3, $\mathbf{p}_3$ is compared for similarity and membership with $\mathbf{W}$

$$\mathbf{dist}_M(\mathbf{p}_3, \mathbf{w}_1) = \left[\left(\begin{bmatrix} 2 \\ 2 \\ 0 \\ 3 \\ 2 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \\ 0 \\ 3 \\ 2 \end{bmatrix}\right)^T \mathbf{K}^{-1} \left(\begin{bmatrix} 2 \\ 2 \\ 0 \\ 3 \\ 2 \end{bmatrix} - \begin{bmatrix} 2 \\ 3 \\ 0 \\ 3 \\ 2 \end{bmatrix}\right)\right]^{1/2} = 0.5,$$

$$\mathbf{mem}(\mathbf{p}_3, \mathbf{w}_1) = \mathbf{exp}\left(-\frac{0.5}{2}\right) = 0.78$$

and

$$\mathbf{dist}_M(\mathbf{p}_3, \mathbf{w}_2) = \left[\left(\begin{bmatrix} 2 \\ 2 \\ 0 \\ 3 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}\right)^T \mathbf{K}^{-1} \left(\begin{bmatrix} 2 \\ 2 \\ 0 \\ 3 \\ 2 \end{bmatrix} - \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}\right)\right]^{1/2} = 6.5,$$

$$\mathbf{mem}(\mathbf{p}_3, \mathbf{w}_2) = \mathbf{exp}\left(-\frac{6.5}{2}\right) = 0.0388.$$

As in Steps 4 and 5, $\mathbf{w}_1$ is determined as the winner cluster is determined using the fuzzy OR operator, because $\mathbf{mem}(\mathbf{p}_3, \mathbf{w}_1) > \mathbf{mem}(\mathbf{p}_3, \mathbf{w}_2)$.

As in Step 6, since $\mathbf{dist}_M(\mathbf{p}_3, \mathbf{w}_1) = d_{\text{th}}$, the cluster variables count, winner cluster and covariance matrix are updated:

$$\mathbf{C}_{1,\text{new}} = \mathbf{C}_{1,\text{old}} + 1 = 2,$$

$$\mathbf{w}_{1,\text{new}} = (1 - \beta)\mathbf{w}_{1,\text{old}} + \beta\mathbf{p}_3 = \begin{bmatrix} 2 \\ 2.5 \\ 0 \\ 3 \\ 2 \end{bmatrix},$$

where $\beta = 1/\mathbf{C}_{1,\text{new}}$,

$$\mathbf{K}_{1,\text{new}} = (1-\beta)\mathbf{K}_{1,\text{old}} + \beta(1-\beta^2)(\mathbf{p}_3 - \mathbf{w}_{1,\text{new}})(\mathbf{p}_3 - \mathbf{w}_{1,\text{new}})^{\mathrm{T}},$$

$$\mathbf{K}_{1,\text{new}} = \begin{bmatrix} 0.99 & 0 & 0 & 0 & 0 \\ 0 & 1.0838 & 0 & 0 & 0 \\ 0 & 0 & 0.99 & 0 & 0 \\ 0 & 0 & 0 & 0.99 & 0 \\ 0 & 0 & 0 & 0 & 0.99 \end{bmatrix}.$$

Finally, after learning with three input vectors, the resulting model is

$$\mathbf{W} = \begin{bmatrix} 2 & 0 \\ 2.5 & 2 \\ 0 & 0 \\ 3 & 0 \\ 2 & 2 \end{bmatrix}.$$

3.4 *Prediction algorithm in InCF*

The prediction phase is aimed to predict the characteristics of an interested user (**p**) by associating the model's cluster with him the closest distance to him. The process of prediction in InCF is shown in Figure 3. It has the following steps.

3.4.1 *Description of prediction algorithm in InCF*

Step 0: read an interested user (**p**).
Step 1: calculate the distance between **p** and the model (**W**) generated in *the learning phase* by using the Mahalanobis distance ($\mathbf{dist}_M$).
Step 2: calculate the value of the membership of **p** in each cluster using the Mahalanobis radial basis function:

$$\mathbf{mem}(\mathbf{p}, \mathbf{W}) = \mathbf{exp}\left(-\frac{(\mathbf{p}-\mathbf{W})^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{p}-\mathbf{W})}{2}\right). \tag{19}$$

Step 3: find the winner cluster by identifying the highest degree of membership:

$$\text{winner} = \texttt{argmax}_i(\mathbf{mem}_i) = \mathbf{mem}_1 \vee \mathbf{mem}_2 \vee \cdots \vee \mathbf{mem}_n. \tag{20}$$

Step 4: associate **p** with the winner cluster and assign corresponding ratings and characteristics.
Step 5: if there is a new interested user **p**, go to Step 0.

3.4.2 *Prediction example with InCF*

One interested user is needed for classification based on a model created previously. The user-item rating matrix of such user and the items are presented in Table 2. Here, the prediction algorithm of InCF is applied to recommend which item this user should choose.

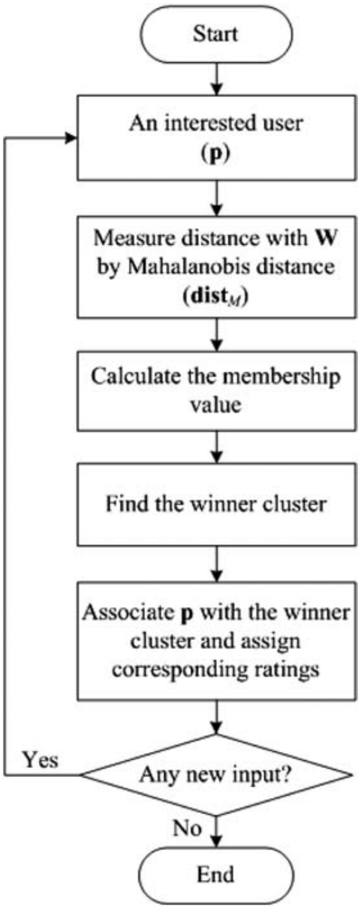
Start
An interested user (**p**)
Measure distance with **W** by Mahalanobis distance (**dist**$_M$)
Calculate the membership value
Find the winner cluster
Associate **p** with the winner cluster and assign corresponding ratings
Any new input?
Yes
No
End

Figure 3. Steps of prediction algorithm in InCF.

Step 0: read the interested user ($\mathbf{p}_1$)

$$\mathbf{p}_1 = \begin{bmatrix} 0 \\ 3 \\ 0 \\ 2 \\ 0 \end{bmatrix}.$$

Step 1: calculate the distance as in Step 3 of the learning example.
Step 2: compute the value of the membership of $\mathbf{p}_1$ in the two clusters of model $\mathbf{W}$:

$$\mathbf{mem}(\mathbf{p}_1, \mathbf{w}_1) = \exp\left(-\frac{(\mathbf{p}_1 - \mathbf{w}_1)^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{p}_1 - \mathbf{w}_1)}{2}\right) = 0.0095$$

Table 2. An artificial user-item rating matrix of the interested user and items.

	Item 1	*Item 2*	*Item 3*	*Item 4*	*Item 5*
User 1	0	3	0	2	0

and

$$\mathbf{mem}(\mathbf{p}_1, \mathbf{w}_2) = \exp\left(-\frac{(\mathbf{p}_1 - \mathbf{w}_2)^{\mathrm{T}}\mathbf{K}^{-1}(\mathbf{p}_1 - \mathbf{w}_2)}{2}\right) = 0.0111.$$

Step 3: the winner cluster is determined by using the fuzzy OR operator. It is $\mathbf{w}_2$ because $\mathbf{mem}(\mathbf{p}_1, \mathbf{w}_1) < \mathbf{mem}(\mathbf{p}_1, \mathbf{w}_2)$

$$\mathbf{w}_2 = \begin{bmatrix} 0 \\ 2 \\ 0 \\ 0 \\ 2 \end{bmatrix}.$$

Step 4: Associate $\mathbf{p}_1$ with the winner cluster, i.e. assign for prediction to $\mathbf{p}_1$ from the winner cluster the rating still missing, namely the 5th in Table 2

$$\mathbf{p}_1 = \begin{bmatrix} 0 \\ 3 \\ 0 \\ 2 \\ 2 \end{bmatrix}.$$

Step 5: if there is a new interested user $\mathbf{p}$, go to Step 0.

4. Experimental analysis

In this section, we first compare the proposed algorithm with traditional clustering and, then, we investigate the effect of CFS by measuring the average normalized mean absolute error (NMAE). Finally, the InCF algorithm is applied to recommend movies to interested users.

4.1 *Experimental set-up*

As a benchmark, the data-set 'MovieLens' is used containing real-movie ratings gathered by the GroupLens research laboratory at the University of Minnesota, and available online at http://www.grouplens.org/node/73. It contains 100,000 ratings for 1682 movies by 943 users. Since there is no standard to split the data-set into a testing and a training data-set, we apply fivefold cross-validation to divide the data randomly into five data-sets. In addition, this prevents any bias caused by the particular sampling chosen for training and testing data-sets.

4.2 *Evaluation matrices*

In order to obtain evaluation results, we calculate predictive accuracy matrices using NMAE, which is determined from the MAE. It calculates the average of the absolute differences between predicted ratings and actual ratings. Both MAE and NMAE calculations are shown as follows (Su and Khoshgoftaar 2009):

$$\mathrm{MAE} = \frac{\sum_{\{i,j\}} |p_{i,j} - r_{i,j}|}{n}, \tag{21}$$

where n is the total number of ratings over all users, $p_{i,j}$ is the predicted rating for user i on item j, and $r_{i,j}$ is the actual rating. The lower the MAE is, the better is the prediction (Su and Khoshgoftaar 2009). Different recommender systems may use different numerical rating scales. NMAE normalizes MAE to express errors as full-scale percentages (Hofmann 2003):

$$\mathrm{NMAE} = \frac{\mathrm{MAE}}{r_{\max} - r_{\min}}, \tag{22}$$

where $r_{\max}$ is an upper and $r_{\min}$ is a lower bound of rating (Hofmann 2003). Finally, we compare the performance, by average NMAE, between the proposed InCF algorithm and K-means based CF.

4.3 *Building a CF by K-means*

In this subsection, experiments are conducted using model-based CF developed on the basis of the K-means algorithm. These experiments aim to compare the performance of the K-means algorithm with and without CFS, respective of data-sets for recommendation benchmarking.

The results are shown in Figure 4, where the average NMAE of five data-sets is compared in a number of different clusters, also CFS is considered in order to reduce the

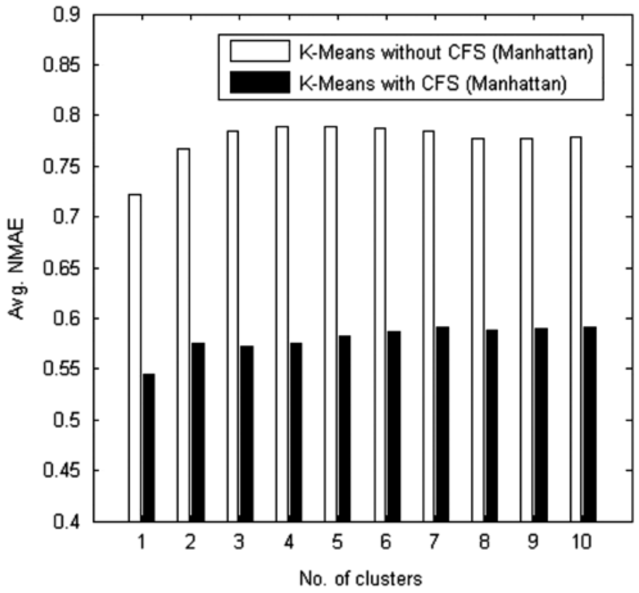

Figure 4. Comparison of average NMAE values of fivefold cross-validation by K-means-based (Manhattan) CF with and without CFS applied to different number of clusters.

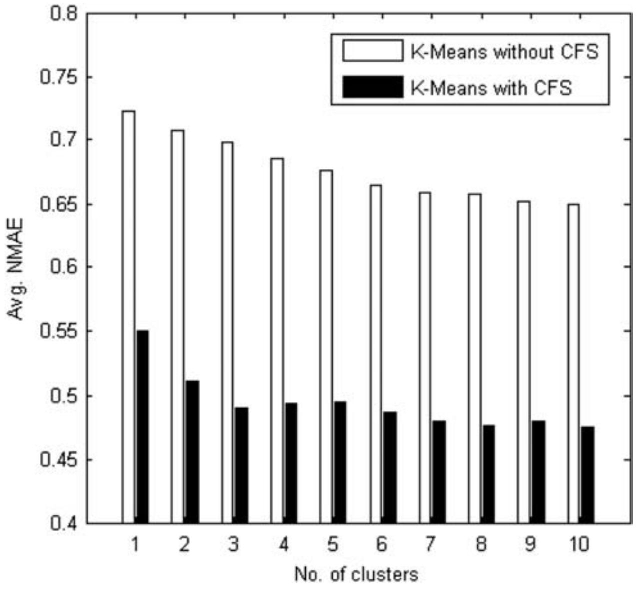

Figure 5. Comparison of average NMAE values of fivefold cross-validation by K-means-based (Euclidean) CF with and without CFS applied to different number of clusters.

data-set's dimensions. The results show that the average NMAE value is minimized when applied to CFS.

The results presented in Figure 5 show that, when comparing all data-sets in different number of clusters, the average NMAE values are better when CFS is applied. In order to reduce the data-set's dimensions, the average NMAE gradually stabilized around 0.474 when the number of clusters reached 10. On the other hand, applying CFS is also efficient to decrease the average NMAE. It yields results lower than for K-means-based CF without CFS in all number of clusters.

This illustrates that the 10th cluster has the lowest average NMAE for K-means-based (Euclidean) CF with CFS. Therefore, our experiment focuses on the 10th cluster with CFS. Figure 6 illustrates the average NMAE values for different data-sets of the 10th cluster. The results are almost the same as the previous results. The minimal average NMAE value is assumed for the third data-set, which is around 0.45 when applying CFS to create a model.

4.4 *Building a CF by K-NN*

In this subsection, experiments are conducted using model-based CF based on K-NN. These experiments aim to compare the performance of K-NN with CFS for different ratios between training and testing data-sets: 20/80, 30/70, 40/60, 50/50, 60/40, 70/30, 80/20, and 90/10, respectively.

It can be observed from Figure 7 that the performance of K-NN-based CF with CFS is enhanced by an increased number of training data-sets.

4.5 *Building a CF by the InCF algorithm*

In this subsection, conventional model-based CF is developed with the InCF algorithm. Figure 8 shows that in our experiments with five data-sets applying fivefold cross-

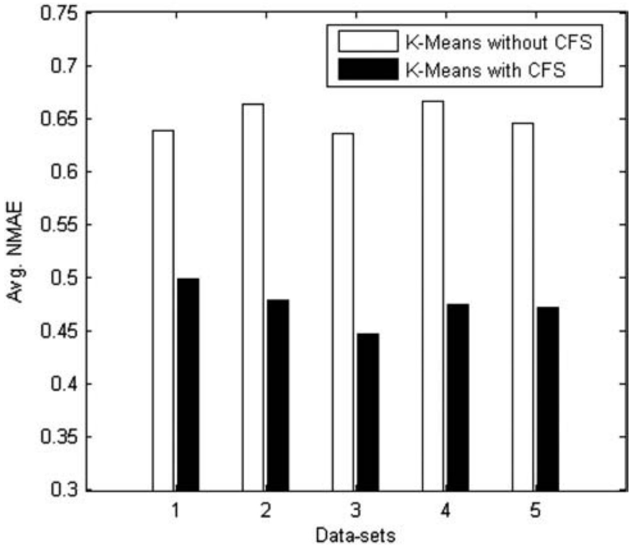

Figure 6. Comparison of NMAE values using K-means-based (Euclidean) CF applied with and without CFS to the 10th cluster for different data-sets.

validation in the benchmark data-set, in the 10th cluster the average NMAE value of InCF is lower than for K-means with CFS and the Manhattan and Euclidean distances. The lowest average value of NMAE for InCF (0.399) is due to the ellipsoidal shapes, special to InCF, better fitting the decision boundaries of feature distributions than spheres. We see that the proposed InCF algorithm has the lowest NMAE (i.e. highest recommendation accuracy) at each cluster size in all cases considered.

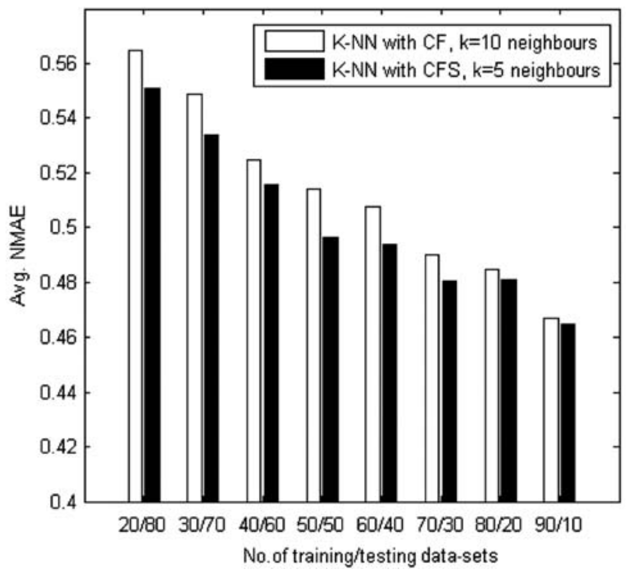

Figure 7. Comparison of average NMAE values by K-NN-based CF with CFS applied to different number of training/testing data-sets.

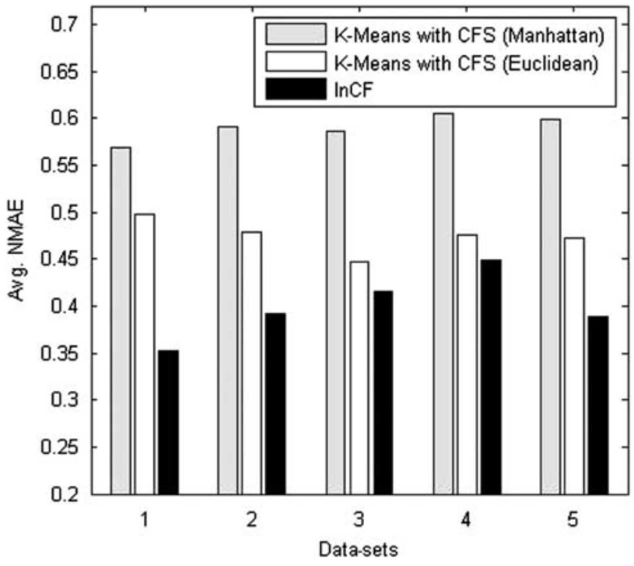

Figure 8. Comparison of average NMAE values between K-means-based CF with Manhattan or Euclidean distance and CFS, and the InCF algorithm in the 10th cluster applied to different data-sets.

Figure 9 shows that the performance of the InCF algorithm is better than K-NN-based CF with CFS, which is due to a higher number of training data-sets and a lower number of testing data-sets. It demonstrates that the proposed InCF algorithm has the lowest NMAE (i.e. highest recommendation accuracy) for all ratios of training/testing data-set sizes at different number of neighbours.

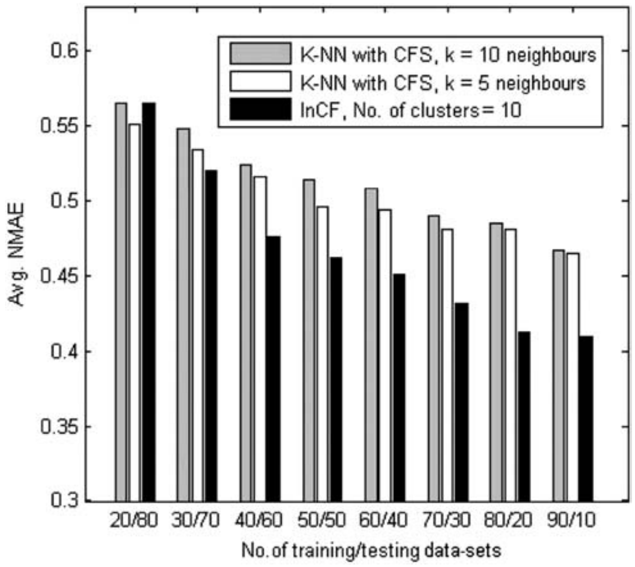

Figure 9. Comparison of average NMAE values between K-NN based-CF with CFS and InCF applied to different number of training/testing data-sets.

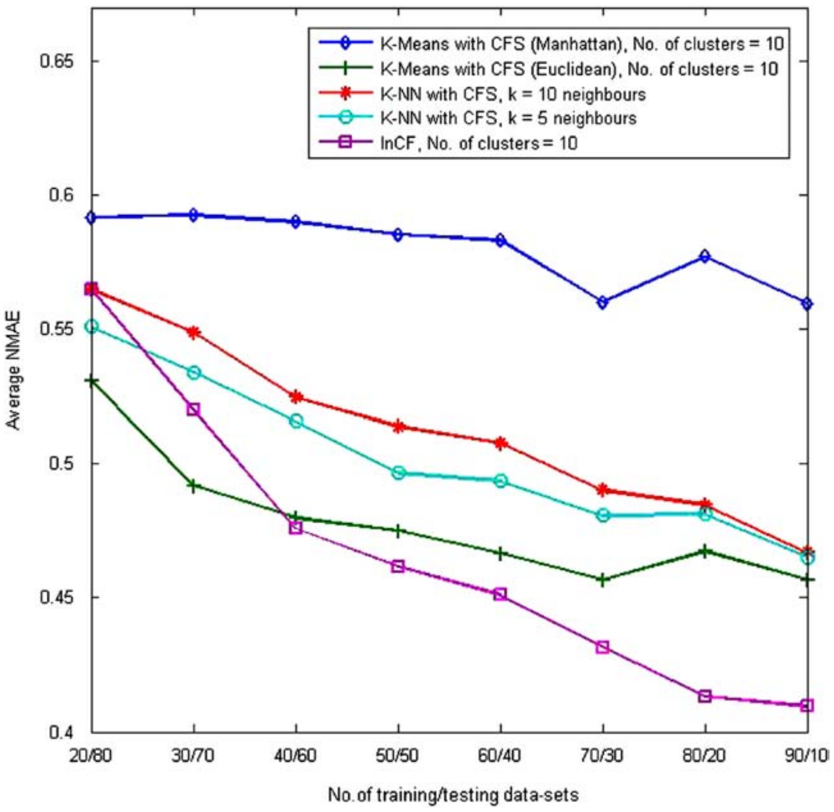

Figure 10. Comparison of average NMAE values between InCF and other clustering algorithms applied for recommender systems.

There is an essential difference in rating prediction between K-means and K-NN clustering. K-means sums up the ratings of all values in a group of similar users and, then, derives values of the centroid to calculate items not rated yet. K-NN takes into account the most similar users as the nearest neighbours, then predicts not yet rated items based on the nearest neighbour. Therefore, we conduct experiments with a different number of training/testing data. Figure 10 shows that the average NMAE for the InCF algorithm is better than for other recommendation algorithms when the ratio between the sizes of training/testing data-sets is not lower than 40/60. Even the average NMAE for ratios of 20/80 and 30/70 is less for InCF than for K-means (Euclidean) and K-NN (k = five neighbours), because the accuracy is affected by the number of training data-sets. For the ratio 90/10, the average NMAE for InCF is 0.40954, which is also better than for other recommendation algorithms.

The experimental results presented in Figure 10 show that the InCF algorithm significantly improves the accuracy and quality of recommendation in comparison to the benchmark clustering algorithms.

4.6 *Recommending a particular movie to interested users by InCF*

In this subsection, the prediction algorithm in InCF is applied to recommend particular movies to interested users. Five interested users are selected randomly, and the developed

Table 3. Information of selected interested users.

ID	*Age*	*Gender*	*Job*
463	48	F	Healthcare
474	51	M	Executive
94	26	M	Student
727	25	M	Student
234	60	M	Retired

algorithm suggests five examples of particular movies to each interested user, which are based on a model created with the learning algorithm in InCF. Table 3 presents the information on selected interested users that consists of age, gender, and career. The results of recommending particular movies to the above users by the prediction algorithm of InCF are shown in Table 4. First, the interested users are categorized by the proposed model based on their similarity. An interested user is categorized according to the model. Based on this, particular movies are recommended to the user in the order of anticipated liking (rating). Here, the experimental results show that five movies are recommended to each interested user. Finally, the movie with the highest rating in the model is recommended to an interested user. For example, interested user '463' is 48 years old and her career is in healthcare. The model assigns the following five movies with highest ratings to her: Vertigo (1958) with rating 4.556, Rear Window (1954) with rating 4.778,

Table 4. Recommended the particular movie to selected interested users.

ID	*Five recommended movies*	*Rating*	*Kind of movie*
463	Vertigo (1958)	4.556	
	Rear Window (1954)	**4.778**	Mystery and Thriller
	North by Northwest (1959)	4.556	
	Casablanca (1942)	4.445	
	Bridge on the River Kwai, The (1957)	4.445	
474	Fish Called Wanda, A (1988)	3.333	
	His Girl Friday (1940)	**4.000**	Comedy
	Gone with the Wind (1939)	3.556	
	Monty Python's Life of Brian (1979)	3.778	
	Harold and Maude (1971)	3.444	
94	Singin' in the Rain (1952)	4.333	
	Bridge on the River Kwai, The (1957)	4.445	
	Secrets and Lies (1996)	4.111	
	Vertigo (1958)	**4.556**	Mystery and Thriller
	North by Northwest (1959)	4.556	
727	Manchurian Candidate, The (1962)	4.445	
	Singin' in the Rain (1952)	4.333	
	Vertigo (1958)	4.556	
	Rear Window (1954)	**4.556**	Mystery and Thriller
	North by Northwest (1959)	4.556	
234	Annie Hall (1977)	3.667	
	Unforgiven (1992)	3.000	
	Trainspotting (1996)	3.111	
	Wings of Desire (1987)	3.000	
	My Left Foot (1989)	**3.889**	Drama

North by Northwest (1959) with rating 4.556, Casablanca (1942) with rating 4.445, and The Bridge on the River Kwai (1957) with rating 4.445. As Rear Window (1954) has the highest rating according to the model, this movie is finally recommended to the interested user '463'. The type of these movies is 'Mystery and Thriller'.

From this experiment, it can be concluded that our proposed approach can predict particular tastes and, thus, can recommend to an interested user the movie with the highest rating assumed to be the most suitable movie.

5. Conclusion and future work

An InCF algorithm based on the Mahalanobis distance is presented. It improves the accuracy of prediction as substantiated by experimental results showing that the average NMAE values for the InCF algorithm are lower than those for ordinary CF based on K-means.

To handle the scalability problem, incremental on-line learning was employed to detect new clusters. Using the InCF algorithm, the interested users do not affect the existing user-item rating matrix, because measuring their similarity with the model clusters does not require time-consuming searches in entire databases. Thus, InCF can cope with the tremendous growth of both items and interested users.

As future work, the development of a medical recommender system is planned, which applies the InCF algorithm to provide diagnostic support to physicians in rural hospitals of developing countries. In this medical recommender system, InCF will be combined with medical knowledge.

Notes on contributors

Maytiyanin Komkhao received a B.Eng degree in civil engineering from Rangsit University, Thailand in 2000, and an M.Sc degree in information technology from King Mongkut's University of Technology North Bangkok, Thailand, in 2007. She is now a doctoral student at Fernuniversität in Hagen, Germany. Her research interests include theory and applications of neural networks, fuzzy sets, pattern classification, machine learning, and recommender systems.

Jie Lu is the Head of School of Software at the University of Technology, Sydney. Her research interests lie in the area of decision support systems and uncertain information processing. She has published five research books and 270 papers, won five Australian Research Council discovery grants and 10 other grants. She received a University Research Excellent Medal in 2010. She serves as Editor-In-Chief for Knowledge-Based Systems (Elsevier), Editor-In-Chief for International Journal of Computational Intelligence Systems (Atlantis), editor for book series on Intelligent Information Systems (World Scientific), and guest editor of six special issues for international journals, as well as delivered six keynote speeches at international conferences.

Zhong Li received his B.Sc, M.Sc, and Ph.D degrees from Sichuan University, Jinan University, and South China University of Technology in 1989, 1996, and 2000, respectively. He obtained his D.Sc (Habilitation) degree from Fernuniversität in Hagen in 2007, where he is now an adjunct professor. His research interests include fuzzy logic and fuzzy control, chaos theory and chaos control, intelligent computation and control, complex networks, and swarm intelligence. He serves as associate editor for six international journals, has published three books with Springer-Verlag, 1 patent, 16 book chapters, 52 journal papers, and 40 conference papers.

Wolfgang A. Halang, born in 1951 in Essen, Germany, received a doctorate in mathematics from Ruhr-Universität Bochum in 1976, and a second one in computer science from Universität Dortmund in 1980. He worked in both industry (Coca-Cola GmbH and Bayer AG) and academia (University of Petroleum and Minerals, Saudi Arabia, and University of Illinois at Urbana-Champaign), before he was appointed to the Chair of Applications-oriented Computing Science and head of the Department of Computing Science at the University of Groningen in the Netherlands. Since 1992, he holds the Chair of Computer Engineering in the Faculty of Electrical and Computer Engineering at Fernuniversität in Hagen, Germany, whose dean he was from 2002 to 2006. He was a visiting professor at the University of Maribor, Slovenia, 1997, and the University of Rome II in 1999. His research interests comprise all major areas of hard real-time computing with special emphasis on safety-related systems. He is the founder and was European editor-in-chief of the journal Real-Time Systems, member of the editorial boards of 4 further journals, co-director of the 1992 NATO Advanced Study Institute on Real-Time Computing, has authored 14 books and over 350 refereed book chapters, journal publications and conference contributions, has edited 24 books, holds 15 patents, has given some 80 guest lectures in more than 20 countries, and is active in various professional organizations and technical committees as well as involved in the programme committees of over 200 conferences. In the International Federation of Automatic Control, he served two terms each as chair of the Technical Committee on Real-Time Software Engineering and as the Co-ordinating Committee on Computers, Cognition, and Communication for Control.

References

Adomavicius, G., and Tuzhilin, A. (2005), 'Towards 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.

Burke, R. (2002), 'Hybrid Recommender Systems: Survey and Experiments,' *User Modeling and User-Adapted Interaction*, 12, 4, 331–370.

Candillier, L., Meyer, F., and Boullé, M. (2007), 'Comparing State-of-the-Art Collaborative Filtering Systems,' in *Proceedings of the 5th International Conference on Machine Learning and Data Mining in Pattern Recognition*, Berlin, Heidelberg: Springer-Verlag, pp. 548–562.

Candillier, L., Tellier, I., Torre, F., and Bousquet, O. (2005), 'SSC: Statistical Subspace Clustering,' in *4th International Conference on Machine Learning and Data Mining in Pattern Recognition*. Lecture Notes in Computer Science. Leipzig: Springer, pp. 100–109.

Cho, Y.H., and Kim, J.K. (2004), 'Application of Web usage Mining and Product Taxonomy to Collaborative Recommendations in E-Commerce,' *Expert Systems with Applications*, 26, 2, 233–246.

Cornelis, C., Lu, J., Guo, X., and Zhang, G. (2007), 'One-and-Only Item Recommendation with Fuzzy Logic Techniques,' *Information Sciences*, 177, 22, 4906–4921.

Dempster, A.P. (1972), 'Covariance Selection,' *Biometrics*, 28, 1, 157–175.

Gong, S. (2010), 'A Collaborative Filtering Recommendation Algorithm Based on User Clustering and Item Clustering,' *Journal of Software*, 5, 7, 745–752.

Hall, M.A. (2000), 'Correlation-Based Feature Selection for Discrete and Numeric Class Machine Learning,' in *Proceedings of the 17th International Conference on Machine Learning*, San Francisco, CA: Morgan Kaufmann Publishers, pp. 359–366.

Han, J., and Kamber, M. (2006), *Data Mining: Concepts and Techniques* (2nd ed.), San Francisco, CA: Morgan Kaufmann Publishers.

Herlocker, J.L., Konstan, J.A., Terveen, L.G., and Riedl, J.T. (2004), 'Evaluating Collaborative Filtering Recommender Systems,' *ACM Transactions on Information Systems*, 22, 1, 5–53.

Hofmann, T. (2003), 'Collaborative Filtering via Gaussian Probabilistic Latent Semantic Analysis,' in *Proceedings of the 26th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York: ACM, pp. 259–266.

Jain, A.K., Murthy, M.N., and Flynn, P.J. (1999), 'Data Clustering: A Review,' *ACM Computing Surveys*, 31, 3, 264–323.

Khoshneshin, M., and Street, W.N. (2010), 'Incremental Collaborative Filtering via Evolutionary Co-Clustering,' in *Proceedings of the 4th ACM Conference on Recommender Systems*, New York: ACM, pp. 325–328.

Lee, S.K., Cho, Y.H., and Kim, S.H. (2010), 'Collaborative Filtering with Ordinal Scale-Based Implicit Ratings for Mobile Music Recommendations,' *Information Sciences*, 180, 11, 2142–2155.

Lekakos, G., and Giaglis, G. (2006), 'Improving Prediction Accuracy of Recommendation Algorithms: Approaches Anchored on Human Factors,' *Interacting with Computers*, 18, 3, 410–431.

Lu, J., Shambour, Q., Xu, Y., Lin, Q., and Zhang, G. (2010), 'BizSeeker: A Hybrid Semantic Recommendation System for Personalized Government-to-Business E-Services,' *Internet Research*, 20, 3, 342–365.

Lu, J., Shambour, Q., and Zhang, G. (2009), 'Recommendation Technique-Based Government-to-Business Personalized E-Services,' in *Proceedings of the 28th North American Fuzzy Information Processing Society Annual Conference*, Piscataway, NJ: IEEE Press, pp. 1–6.

Mahalanobis, P.C. (1936), 'On the Generalised Distance in Statistics,' *Proceedings of the National Institute of Science of India*, 2, 1, 49–55.

Papagelis, M., Rousidis, I., Plexousakis, D., and Theoharopoulos, E. (2005), 'Incremental Collaborative Filtering for Highly-Scalable Recommendation Algorithms,' in *Proceedings of the 15th International Symposium on Methodologies of Intelligent Systems*, Berlin, Heidelberg: Springer-Verlag, pp. 553–561.

Pham, M.C., Cao, Y., Klamma, R., and Jarke, M. (2011), 'A Clustering Approach for Collaborative Filtering Recommendation Using Social Network Analysis,' *Journal of Universal Computer Science*, 17, 4, 583–604.

Polikar, R., DePasquale, J., Mohammed, H.S., Brown, G., and Kuncheva, L.I. (2010), 'LEARN++.MF: A Random Subspace Approach for the Missing Feature Problem,' *Pattern Recognition*, 43, 11, 3817–3832.

Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J. (2000), 'Analysis of Recommendation Algorithms for E-Commerce,' in *Proceedings of the 2nd ACM Conference on Electronic Commerce*, New York: ACM, pp. 158–167.

Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J. (2002a), 'Recommender Systems for Large-Scale E-Commerce: Scalable Neighbourhood Formation Using Clustering,' in *Proceedings of the 5th International Conference on Computer and Information Technology*, San Jose, CA: San Jose State University, pp. 158–167.

Sarwar, B.M., Karypis, G., Konstan, J.A., and Riedl, J. (2002b), 'Incremental Singular Value Decomposition Algorithms for Highly Scalable Recommender Systems,' in *Proceedings of the 5th International Conference on Computer and Information Technology*, San Jose, CA: San Jose State University, pp. 27–28.

Shambour, Q., and Lu, J. (2011), 'A Hybrid Trust-Enhanced Collaborative Filtering Recommendation Approach for Personalized Government-to-Business-Services,' *International Journal of Intelligent Systems*, 26, 9, 814–843.

Su, X., and Khoshgoftaar, T.M. (2009), 'A Survey of Collaborative Filtering Techniques,' in *Advances in Artificial Intelligence*, New York: Hindawi Publishing Corporation, pp. 1–20.

Tanner, R., and Cruickshank, D.G.M. (1998), 'RBF Based Receivers for DS-CDMA with Reduced Complexity,' in *Procceedings of 1998 IEEE 5th International Symposium on Spread Spectrum Techniques and Applications*, Piscataway, NJ: IEEE Press, Vol. 2, pp. 647–651.

Vuskovic, M., and Sijiang, D. (2002), 'Classification of Prehensile EMG Patterns with Simplified Fuzzy ARTMAP,' in *Proceedings of the 2002 International Joint Conference on Neural Networks*, Piscataway, NJ: IEEE Press, Vol. 3, pp. 2539–2544.

Witten, I.H., and Frank, E. (2005), *Data Mining: Practical Machine Learning Tools and Techniques*, San Francisco, CA: Morgan Kaufmann Publishers.

Xu, H., and Vuskovic, M. (2004), 'Mahalanobis Distance-Based ARTMAP Network,' in *Proceedings of the IEEE International Joint Conference on Neural Networks*, Piscataway, NJ: IEEE Press, Vol. 3, pp. 2353–2359.

Xue, G.-R., Lin, C., Yang, Q., Xi, W., Zeng, H.-J., Yu, Y., and Chen, Z. (2005), 'Scalable Collaborative Filtering Using Cluster-based Smoothing,' in *Proceedings of the 28th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, New York: ACM, pp. 114–121.

Yen, G.G., and Meesad, P. (2001), 'An Effective Neuro-Fuzzy Paradigm for Machinery Condition Health Monitoring,' *IEEE Transactions on Systems, Man, and Cybernetics, Part B*, 31, 4, 523–536.