

## Fuzzy prototype classifier based on items and its application in recommender system

Mei Cai<sup>1</sup>, Zaiwu Gong<sup>2,1</sup>, Yan Li<sup>1</sup>,

<sup>1</sup> School of Economics and Management  
Nanjing University of Information Science and Technology  
Nanjing, China, 210044

<sup>2</sup> Collaborative Innovation Center on Forecast and Evaluation of Meteorological Disasters  
Nanjing University of Information Science and Technology  
Nanjing, China, 210044

Received 26 December 2016

Accepted 25 June 2017

### Abstract

Currently, recommender systems (RS) are incorporating implicit information from social circle of the Internet. The implicit social information in human mind is not easy to reflect in appropriate decision making techniques. This paper consists of 2 contributions. First, we develop an item-based prototype classifier (IPC) in which a prototype represents a social circle's preferences as a pattern classification technique. We assume the social circle which distinguishes with others by the items their members like. The prototype structure of the classifier is defined by two 2-dimensional matrices. We use information gain and OWA aggregator to construct a feature space. The item-based classifier assigns a new item to some prototypes with different prototypicalities. We reform a typical data set—Iris data set in UCI Machine Learning Repository to verify our fuzzy prototype classifier. The second proposition of this paper is to give the application of IPC in recommender system to solve new item cold-start problems. We modify the dataset of MovieLens to perform experimental demonstrations of the proposed ideas.

**Keywords:** recommender system (RS), item-based prototype classifier (IPC), prototype structure, feature space

### 1. Introduction

A recommender system (RS) seeks to discover information items (movies, music, books, news, images, web pages, papers, etc.) that are valuable to the user[1, 2]. Content-based[3, 4] or collaborative filtering(CF)[5] techniques are commonly used techniques. CF is the most popular approach used for recommender systems. There are two major branches in CF: neighborhood-based (NB) and matrix factorization (MF)[6]. The recommender system would identify users who share the similar tastes and interests in the past with a group of people, and propose items which the like-minded users favored. But it suffers from complete

cold start (CCS) problem where no rating records are available and incomplete cold start (ICS) problem where only a small number of rating records are available for some new items or users in the system[7]. Bobadilla, et al.[8] distinguish cold-start problems as three kinds: new community, new item and new user. New item and new user problems correspond to the cases when a new item/user enters an already existing system[9]. The approaches for user cold-start can be divided into two categories. The first type refines the similarity metrics used to identify effective reference users. The second type utilizes the social relations between users to identify reference users for cold start new users[10].

Most personalized recommender systems are based mainly on the relation information among users on social network. If users are closely connected or linked to each other in their social circle, there is a high probability that they have similar interests and interact with each other actively[11]. However the elicitation of customers preferences is not always precise either correct, because of external factors such as human errors, uncertainty and vagueness proper of human beings and so on[12]. Recommendations for item cold-start are given through comparing the properties of a new items to the properties of those items that are known to be of liked by some kind users. The main limit of such a solution is the uninterruptable relations between factorization and properties. Sun, et al. [13] propose a hybrid algorithm by using both the ratings and content information to tackle item-side cold-start problem. Aleksandrova, et al.[9] present a hybrid matrix factorisation model representing users and items. Anava, et al. [14] propose efficient optimal design based algorithms to solve budget-constrained item cold-start.

The cold-start problem occurs because of lack of information. But on the other hand, the recent explosion of information has made people lost and confused. With the rapid development of internet, people can now obtain and share information easily with other people. As the web 2.0 has developed, RS has increasingly incorporated social information[8]. Various social circles which are the collection of people with similar interests and tastes are established. These large amounts of social circle generate a number of valuable knowledge for people and companies. There are two points making searching for useful information a difficult task.

(i) Social circles have ill-defined boundaries. Humans are to belong to many associative groups simultaneously, with various levels of affiliation[15].

(ii) In a social circle, users express their opinions in an unconstrained manner, and share these data with others. Users are usually not well-trained decision analysts. They need to spend as little time and cognitive effort as possible in giving their preferences[16].

We need to solve the problems on social big data domain related to different aspects as knowledge representation, data management, data processing, data analysis, and data visualization [17]. In the psychology or the cognitive sciences, the study of human preferences has had a long history. The preferences structure of a social circle is an opaque concept.

High-level descriptions of a social circle's preferences are needed, because of the above two points.

A prototype is an opaque concept in the sense that even though we may be able to define it by exemplification, we may not be able to formulate explicit operational criteria for assessing the degree to which a schema qualifies as a prototype[18]. This character is just what preferences structure of a social circle has. The prototype based classification is intuitive approach based on representatives (prototypes) for the respective classes[19]. Many nearest neighbor algorithms [20] have been applied into prototype based classifier. The prototypes are well positioned to capture the distribution of each class in a feature space. Each prototype has a class label, and classification of each new instance is carried out by finding the closest prototype using some defined distance measures [21]. There are some variations of nearest neighbor algorithms such as nearest prototype classification (NPC) algorithms [22, 23], fuzzy rough prototype selection method [24], prototype-based fuzzy clustering (PFC) algorithm [25], dissimilarity-based classifiers[26]. Yera and Martinez [27] point out that a relatively high amount of research works have proved that collaborative filtering approaches with fuzzy tools can be useful in recommendation scenarios. These algorithms and their extensions are widely applied in handwriting digit recognition[28], personalized recommendations[1], human activity pattern analysis (like Web usage mining, travel behavior, tourists and shopping paths)[29], Web-based Personal Health Records (web PHRs)[30].

The Nearest Prototype Classifier (NPC)[19, 23, 31-35] is perhaps the simplest and most intuitively motivated classifier in the pattern recognition. NPC algorithms decrease classification time by reducing the number of instances in the data set [22]. Since the accuracy of such classification techniques is very important, Deng, et al.[25] apply transfer learning to prototype-based fuzzy clustering (PFC) in order to solve the problem of limitation and scarcity of data for clustering task, and Fischer, et al. [36] present simple and efficient reject options for prototype-based classification to reach a reliable decision.

These researches make improvements in the fuzzy feature distance measure. But choosing discriminative features, as well as constructing the feature space is very important. Attentions must be paid not only to build the representation set (a collection of prototypes), but also to

the feature extraction and the distance measure definition[37]. Most classifiers are based on the assumption that the descriptor of a prototype is well-structured. The feature extraction which helps us to generate prototypes is especially difficult for pattern classification of opaque concepts. In this paper, we try to make improvements in this aspect.

The remainder of this paper is organized as follows: in Section 2 a recommender system scenario is described. In Section 3 the item-based prototype classifier (IPC) is proposed. A numerical example is given to complete the fuzzy concept recognition and classification process. And in Section 4, we propose an item-based recommendation approach. We modify dataset of MovieLens to perform experimental demonstrations of the proposed ideas. The conclusions are stated in Section 5.

## 2. A recommender system scenario description

When a new item attends into a market, it has no previous sales records. That is “item cold-start problem”.

We need to predict the “rating” or “preference” and recommend to the suitable users. The latent features that result from the factorization are not directly interpretable. Providing interpretation for these features is important not only to help explain the recommendations presented to users, but also to understand the underlying relations between the users and the items[9].

Customers are divided into several groups. A group of customers constructs a social circle. They share information easily with other people of the similar interests and tastes. Recommender systems behave as black boxes. There is probably no univocal notion of a social circle’s preference structure in psychology or the cognitive sciences. The difficulty is to find the suitable descriptor to describe the social circle’s preference structure.

Another character of the scenario is based on the present situation. For example, on Amazon, recommended products are listed under the title “Customers who bought this item also bought”. Fig. 1 presents an edited screen shot of an Amazon product page.

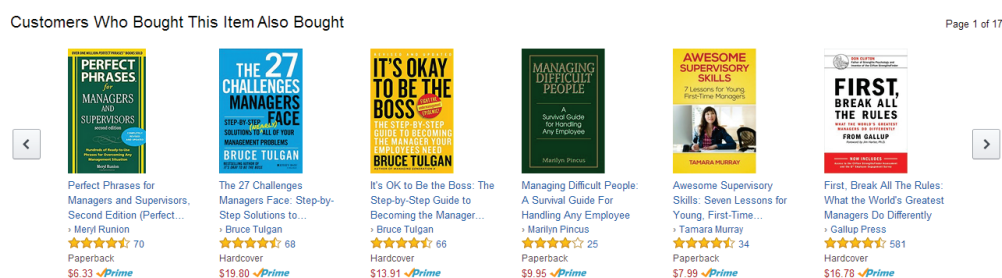


Fig. 1. Amazon product web page

From Fig.1. we can see every item which is recommended is given a total score. We can not know why it is given a higher or lower score. There are several features affecting the total score, such as content, illustration, and price. But these features have different weights in different customers’ minds. On the other hand, this idea is hard to describe. It is tacit knowledge. Customers are not willing to spend much time in giving their preferences. So many recommender systems in e-commerce can only collect preferences in this way.

## 3. IPC: Item-based Prototype Classifier

### 3.1. Prototype structure description

Although prototype classifier techniques in recommender systems are very different, they are based on the same assumptions: (1) structures of prototypes are accuracy, and (2) classifiers are based on nearest neighbour rule. That is to say, (1) a perfect example can be gotten to describe a prototype, and (2) the feature weights are known when deciding the similarity of an unlabelled example. We call these classifiers feature-based prototype classifier. They are not applicable for opaque concepts

like social circles' preferences. In our prototype, we assume the feature space is influenced by items. The differences between two prototypes are reflected by items, not features. We call it item-based prototype classifier (IPC). Firstly, we propose an item-based prototype to describe social circle. In the following paragraphs, the main concepts used in our model are described.

Assuming the recommender system which contains a set of items  $Item := \{u_1, u_2, u_3, \dots, u_n\}$ , a set of features  $Fea := \{F_1, F_2, F_3, \dots, F_m\}$  and a set of social circles which is represented as a prototype set  $Pro := \{A_1, A_2, A_3, \dots, A_h\}$ . A social circle is a labeled prototype  $A_i$  which is composed by a subset of items and the memberships of the items belonging to the prototype. So the prototype  $A_i$  will be denoted by

$$A_i = \mu_{i1}/u_1 + \mu_{i2}/u_2 + \dots + \mu_{in}/u_n \quad (1)$$

which means that  $A_i$  consists of  $u_j$  with grade of membership  $\mu_{ij}$ . Bigger the membership  $\mu_{ij}$  is, the more favourable the item is by the social circle. That is to say, the bigger an item's membership is, the more characteristic it is of the opaque concept  $A_i$ .

The item  $u_j$  is also represented as a fuzzy multiset:

$$u_j = \mu_{1j}/A_1 + \mu_{2j}/A_2 + \dots + \mu_{hj}/A_h \quad (2)$$

which means that  $u_j$  has the relationship with different prototypes.

Each item  $u_j$  is assumed to consist of some features. The feature vector of  $u_j$  is  $Rep(u_j) := \{v_{j1}v_{j2}v_{j3} \dots v_{jm}\}$ .

Humans are well known to belong to many associative groups simultaneously, with various levels of affiliation, which is called overlapping groups[15]. Let's take an example to explain the relations between items and social circles. Junk eaters and healthy eaters compose two social circles. All kinds of food are items. Junk eaters like junk food, but it does not mean that they never buy healthy food. Junk foods are liked by junk eaters, but it does not mean that they never be bought by healthy eaters. Both of these junk food and healthy food have the potential possibility in their shopping baskets, though some are more likely than others. So an item has the relation with all prototypes which was not pointed out by previous literature. The grade of membership  $\mu_{ij}$  defines the degree to which each item  $u_j$  fits  $A_i$ .

The descriptor can be represented by two 2-dimensional matrices, where  $x$  axis is associated to prototypes of the whole descriptor  $A_i \in Pro$ , and  $y$  axis is

associated to items  $u_j \in Item$ . Each position  $(x, y)$  of the matrix is  $\mu_{ij}$  which represents the degree to which each item  $u_j$  fits prototype  $A_i$ .  $z$  axis is associated to feature  $F_p \in Fea$ . Each position  $(y, z)$  of the matrix is  $v_{jp}$  which represents the feature of item  $u_j$ . Fig.2. shows the structure used to keep all this information.

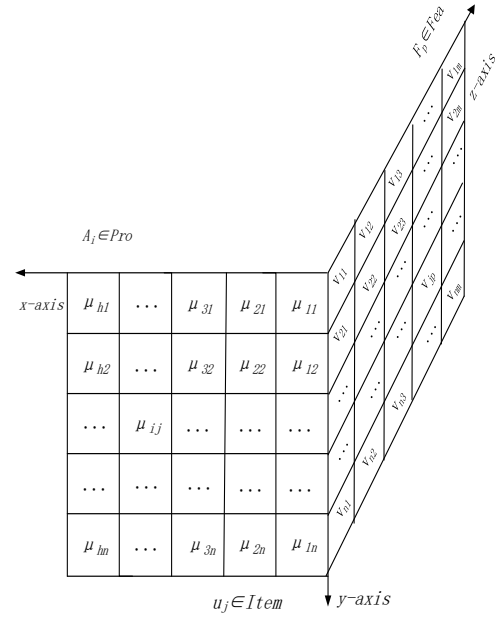


Fig.2. The two 2-dimensionnal matrices of a prototype

If a category is commonly discriminated by features, then decision makers know well the feature's role in deciding which category the item belongs to. Decision makers also know well the best position in the feature space to represent the category. That is to say, we calculate the distance between a new item and an assumed item whose features best represent the category. But in our problem, the perfect item is not existing. We can only use several items to reflect the differences between two prototypes. The complex relationship of features is hard to tell. So we apply Fig.2. to describe a prototype's structure.

From function (2), we can see an item can belong to several prototypes. These items reflect the characteristics of a prototype from different aspects. When an unlabeled item needs classification, it should also be represented as a multiset. The prototypicality of a prototype is calculated by aggregating distances of the unlabeled item and the items in the set of the prototype. The prototype whose degree of prototypicality is smaller than the other reflects

that this prototype is more suitable to label the unlabeled item.

### 3.2. Identification of the feature space

When identifying a feature space of a prototype, we need to identify two things: features and features' weights. There is a feature set  $Fea := \{F_1, F_2, F_3, \dots, F_m\}$  which composes the elements in the feature space of a prototype. The feature  $F_p$  is associated to a group of feature values  $v_{jp} (j = 1, 2, \dots, n)$  of item  $u_j$ . The importance of feature value  $v_{jp}$  in the group is determined by  $w_p$ . An implicit feature weighting is subjective and different for each user. A common feature weighting method based on the Term Frequency-Inverse Document Frequency (TF-IDF) is not good at dealing with multi-valued features. In a more general case, the features can be assessed by multi-valued variables or different domains: numeric, linguistic or nominal, Boolean, etc. [38]. We use **information gain** to decide the relevance of a feature. We take into account the entropy or amount of information for each feature, the more entropy the more weighting should have. This idea is also used by Barranco et al. [38] to assign weights to features in content-based recommender systems.

Information gain,  $Gain(Item, F_p)$  is the measure of the difference in entropy from before to after the set, when  $Item$  is split on a feature  $F_p$ . In other words, how much uncertainty in  $Item$  was reduced after splitting set  $Item$  on feature  $F_p$ . Information gain can be calculated for each remaining feature.

$$Gain(Item, F_p) = E(Item) - EE(F_p) \quad (3)$$

Now we give the definitions of  $E(Item)$  and  $EE(F_p)$  in function (3).

**Definition 1.** Entropy  $E(Item)$  is a measure of the amount of uncertainty in the (data) set  $Item$ .

$$E(Item) = - \sum_{\substack{A_i \in Pro \\ u_j \in Item}} p(A_i) \log_2 p(A_i) \quad (4)$$

where,

- $Item$  - The current (data) set for which entropy is being calculated.

- $A_i$  - Subset of classes in  $Pro$ .  $u_j \in Item$  is classified into subsets  $A_i$  with the grade  $\mu_{ij}$ .
- $p(A_i)$  - The proportion of the number of elements  $u_j$  in class  $A_i$  to the number of elements in set  $Item$ .  $p(A_i) = \sum_{j=1}^n \mu_{ij} / \sum_{i=1}^h \sum_{j=1}^n \mu_{ij}$ .

Feature values may be continuous values or discrete values. If the diversity of feature values is not high, we can separate  $Item$  into  $\{Item_{m_p^1}, Item_{m_p^2}, \dots, Item_{m_p^a}\}$  directly according to feature values  $v_{jp}$ . The items with the same feature value are group into one group. If the diversity of feature values is high, feature values of all items need clustering in order to simplify data and reduce the number of subjects. For the  $p^{th}$  feature, we group  $v_{jp}$  into a target value set  $\{m_p^1, m_p^2, \dots, m_p^a\}$  which contains  $a$  elements. We suppose  $a$  is much smaller than the number of different feature values. Feature values of all items fall into the set. In such a way that feature's values in the same group are more similar to each other than to those in other groups. We separate  $Item$  into  $\{Item_{m_p^1}, Item_{m_p^2}, \dots, Item_{m_p^a}\}$  according to feature values. If target value of  $v_{jp}$  is closest to  $m_p^t$ , then  $u_j$  is fall into set  $Item_{m_p^t}$ . Since  $a$  is much smaller than the number of elements of feature values, the computation load will be reduced. Now we give the definition of Expected entropy.

**Definition 2.** For the  $p^{th}$  feature, Expected entropy is

$$EE(F_p) = \sum_{t=1}^a p(F_p^t) E(F_p^t) \quad (5)$$

where,

- $Item$  - The current (data) set for which entropy is being calculated.
- $E(F_p^t)$  - is a measure of the amount of uncertainty in the set  $Item_{m_p^t}$ .
- $p(F_p^t)$  The proportion of the number of elements  $Item_{m_p^t}$  to the number of elements in set  $Item$ .

$$m^t = \sum_{i=1}^h \sum_{u_j \in Item_{m_p^t}} \mu_{ij}, \text{ where } u_j \in Item_{m_p^t}$$

$$p(F_p^t) = m^t / \sum_{i=1}^a m^i$$

**Definition 3.** Entropy  $E(F_p^t)$  is defined

$$E(F_p^t) = - \sum_{\substack{A_i \in Pro \\ u_j \in Item_{m_p^t}}} p(A_i^{m_p^t}) \log_2 p(A_i^{m_p^t}) \quad (6)$$

where,

- $Item_{m_p^t}$  - The current (data) set for which entropy is being calculated
- $A_i^{m_p^t}$  - Subset of classes in  $A_i$ . If  $u_j \in Item_{m_p^t}$  and target value of  $v_{jp}$  is  $m_p^t$ , then  $u_j$  is classified into subsets  $A_i^{m_p^t}$ .
- $p(A_i^{m_p^t})$  - The proportion of the number of elements  $u_j$  in class  $A_i^{m_p^t}$  to the number of elements in set  $Item_{m_p^t}$ .  $p(A_i^{m_p^t}) = \frac{\sum_{u_j \in A_i^{m_p^t}} \mu_{ij}}{\sum_{u_j \in Item_{m_p^t}} \mu_{ij}}$

The feature with the largest information gain is used to split the set on this iteration. If feature  $F_p$  is more important, its weight  $w_p$  is larger. After gaining the preferred sequence of features according to  $Gain(Item, F_p)$ , we assign weights to these features.

We use OWA (order weighted averaging) operator[39] as the aggregation operator to get the distance between  $u_j$  and an unlabeled item  $u_x$ . OWA has the ability to get optimal weights of the features based on the rank of these feature vectors.

**Definition 4.** An OWA operator of dimension  $n$  is a mapping  $F: R^n \rightarrow R$  having an associated weighting vector  $W = (w_1, \dots, w_n)$  with

$$\sum_{i=1}^n w_i = 1 \text{ where } 0 < w_i < 1, \text{ such that}$$

$$F(a_1, \dots, a_n) = \sum_{i=1}^n b_i w_i \quad (7)$$

where  $B = (b_1, \dots, b_n)$ , and  $b_i$  is the  $i^{\text{th}}$  largest element of the collection of aggregates  $(a_1, \dots, a_n)$ .

There are two characterizing measures related to the weighting vector of an OWA operator. The first one, a measure of orness of the aggregation which is defined as

$$orness(W) = \frac{1}{n-1} \sum_{i=1}^n (n-i)w_i \quad (8)$$

As more of the total weight moves to the weights at the top we are giving more preference to the larger valued arguments. On the other hand by moving more of the total weight to the bottom we are giving preference to the smaller valued arguments in calculating the OWA aggregation. Larger value of  $orness(W)$  indicates the preference for larger argument values. While lower value of  $orness(W)$  indicates the preference for smaller argument values [40].

The second one, the classical measure of information uncertainty, dispersion, called Shannon entropy, is defined by

$$disp(W) = -\sum_{i=1}^n w_i \ln w_i \quad (9)$$

It represents the degree to which the aggregation takes into account all information available in the aggregation, in other words, it measures the average amount of information involved in  $W$ .

Here the idea is to find the set of weights, among those with the desired attitudinal character,  $orness(W)$ , having the largest dispersion. Model(1), suggested by [41] to obtain  $W$ , is based on the solution of the following mathematical programming problem:

Model(1)

$$\begin{aligned} \text{Max} &= \sum_{i=1}^n w_i \ln w_i \\ \text{s.t.} &\begin{cases} \frac{1}{n-1} \sum_{i=1}^n (n-i)w_i = A \\ \sum_{i=1}^n w_i = 1 \text{ where } 0 < w_i < 1 \end{cases} \end{aligned}$$

We require  $A = orness(W) \in [0.5, 1]$  making  $w_i > w_{i+1}$ .

The process of identifying the feature space is as following:

Step 1. Identify the features in the feature space.

Step2. Calculate the information gain for each feature.

Step 3. Get the preferred sequence of features.

Step 4. Assign weight to feature  $F_p$  according to position in the preferred sequence.

The final identified feature space of all prototypes is described as Fig.3.

	F1	F2	F3	F4	F5	F6	
	...	...	...	...	...	...	
U4	$v_{41}$	$v_{42}$	$v_{43}$	$v_{44}$	$v_{45}$	$v_{46}$	... $\mu_{i4}$
U3	$v_{31}$	$v_{32}$	$v_{33}$	$v_{34}$	$v_{35}$	$v_{36}$	... $\mu_{i3}$
U2	$v_{21}$	$v_{22}$	$v_{23}$	$v_{24}$	$v_{25}$	$v_{26}$	... $\mu_{i2}$
U1	$v_{11}$	$v_{12}$	$v_{13}$	$v_{14}$	$v_{15}$	$v_{16}$	... $\mu_{i1}$
	W1	W2	W3	W4	W5	W6	

Fig.3. The description of feature space



### 3.3. Item-based prototype classification process

The fuzzy prototype classification process has three phases. In the preparation phase, we pretreat data set to get a simpler one. This phase is not necessary for all classification process. In the training phase, we construct the feature space for every prototype. The second phase we calculate prototypicality of the unlabeled item to a prototype.

#### 1). preparation phase

As we mentioned in above subsection, diversity of feature values is high, feature values of all items need clustering in order to simplify data and reduce the number of subjects. We use K-means Clustering to group  $v_{jp}$  into set  $\{m_p^1, m_p^2, \dots, m_p^a\}$ . So  $y$  possible values of the feature  $F_i$  are clustered to  $a$  ( $a < y$ ) groups. We calculate the centroid  $m_p^l$  ( $0 < l < a$ ) of the group as the target values of the features. Suppose that we have  $y$  sample feature vectors  $v_{1p}, v_{2p}, \dots, v_{yp}$ , and we know that they fall into  $a$  compact clusters,  $a < y$ . We firstly initiate  $a$  centroids at random for each cluster which is presented as  $m_p^{l(1)}$ . And we assign  $v_{jp}$  into set  $Item_l$  whose centroid is  $m_p^{l(1)}$ . The next step is to associate remainder point to the nearest centroid. This algorithm aims at minimizing an objective function. The objective function

$$J = \sum_{l=1}^a \sum_{v_{jp} \in Item_l^{(i)}} \|v_{jp} - m_p^l\| \quad (10)$$

where  $\|v_{jp} - m_p^l\|$  is a chosen distance measure between a data point  $v_{jp}$  and the cluster centre  $m_p^l$ .  $J$  is an indicator of the distance of the  $y$  data points from their respective cluster centers. Then we recalculate  $a$  new centroids of the clusters resulting from the previous step. A new binding has to be done between the same data set points and the nearest new centroid. A loop has been generated. As a result of this loop we may notice that the  $a$  centroids change their location step by step until no more changes are done. The K-means Clustering [42] algorithm sees Appendix 1.

Now the feature vector of  $u_j$  is  $Rep(u_j) := \{m_1^{j_1} m_2^{j_2} m_3^{j_3} \dots m_m^{j_m}\}$ , where  $m_p^{j_p}$  is the target value of the feature  $F_p$ .  $v_{jp}$  is replaced by  $m_p^{j_p}$ .

#### 2). training phase

In this phase, we obtain a set of items

$Item: \{u_1, u_2, \dots, u_n\}$ .  $u_j$  is represented as function (2). We know its feature values and membership with each prototype. All items can be classified into one of the three subsets  $Item_{\{Good\}}^k$ ,  $Item_{\{Medium\}}^k$  and  $Item_{\{Poor\}}^k$  to represent a prototype, where  $Item_{\{Good\}}^k$ ,  $Item_{\{Medium\}}^k$  and  $Item_{\{Poor\}}^k$  is the corresponding high, medium and low membership degrees respectively.

$$A_k = membership/Item_{\{Good\}}^k + membership/Item_{\{Medium\}}^k + membership/Item_{\{Poor\}}^k \quad (11)$$

We use information gain to obtain two 2-dimensional matrices to describe prototype structure and identification the feature space. More details can be seen in subsection 3.1 and 3.2.

#### 3). performance phase

In this phase we classify the unlabeled item to a prototype by calculating the prototypicality. We only know the feature values of the unlabeled item. Each unlabeled item is carried out by finding the closest prototype using a defined distance measure. The distance between prototype  $A_k$  is as calculated by function(12) (13).

We use items from  $Item_{\{Good\}}^k$  to present prototype  $A_k$ .

$$G_x^k = \sum_{u_j \in Item_{\{Good\}}^k} \mu_{kj} \cdot d(u_j, u_x) \quad (12)$$

$$d(u_j, u_x) = \sum_{p=1}^m w_p \cdot d(v_{xp}, v_{jp}) \quad (13)$$

We can get  $G_x^1, G_x^2, \dots, G_x^h$  for every  $u_x$ .  $G_x^k$  reflects the distance between  $u_x$  and prototype  $A_k$ . Smaller  $G_x^k$  is, closer  $u_x$  is to  $A_k$ .

Then  $u_x$  is represented as a fuzzy set:

$$u_x = G_x^1/A_1 + G_x^2/A_2 + \dots + G_x^h/A_h \quad (14)$$

We assign  $u_x$  into three subsets  $Item_{\{Good\}}^a$ ,  $Item_{\{Medium\}}^b$  and  $Item_{\{Poor\}}^c$  according to  $G_x^l, l = 1, 2, \dots, h$ .

- 1).  $a = i$ , where  $G_x^i = \min_{k=1,2,3}(G_x^k)$
- 2).  $c = i$ , where  $G_x^i = \max_{k=1,2,3}(G_x^k)$
- 3). Otherwise  $b = i$ .

### 3.4. A numerical example

To verify our fuzzy prototype classifier, we choose a fragment of a typical data set—Iris data set in UCI Machine Learning Repository<sup>1</sup>. It has four numerical

<sup>1</sup> UCI machine learning repository <http://www1.ics.uci.edu/~mllearn/MLSummary.html>.

attributes, three classes, and 150 examples. This data set is for crisp classifications. Every example in the data set completely belongs to one class. But in our model, category is required to be a fuzzy class. Item  $u_j$  in our model is represented as a fuzzy multiset:

$$u_j = u_{1j}/A_1 + u_{2j}/A_2 + u_{3j}/A_3 \quad (15)$$

Before the verification we use FCM function in Matlab<sup>2</sup> to generate membership set  $[\mu_{ij}]_{3 \times 150}$  (see Appendix 2). Iris data set in UCI is reformed to fuzzy class data set which allows the fuzzy prototype model to classify each data example as a certain class. In the reformed Iris data set, the membership  $\mu_{ij}$  defines the degree to which each item  $u_j$  fits  $A_i$  as a category. We assign  $u_j$  into three subsets  $Item_{\{Good\}}^a$  (which is the good example set of  $A_a$ ),  $Item_{\{Medium\}}^b$  (which is the medium example set of  $A_b$ ) and  $Item_{\{Poor\}}^c$  (which is the poor example set of  $A_c$ ) according to  $\mu_{ij}, i = 1, 2, 3$ .

- 1).  $a = i$ , where  $\mu_{ij} = \max_{k=1,2,3}(\mu_{kj})$
- 2).  $c = i$ , where  $\mu_{ij} = \min_{k=1,2,3}(\mu_{kj})$
- 3). Otherwise  $b = i$ .

The verification is separated into two stages: in the first stage, we use a training set to discover feature space for each prototype; and in the second stage, we use the prediction method to verify the inference of the fuzzy prototype model.

#### The first stage:

We select ten examples from each set  $Item_{\{Good\}}^i, i = 1, 2, 3$ , to compose the training set at random. Others compose the testing set. The training set contains 30 examples and 4 features.

Samples 1-10 belong to  $Item_{\{Good\}}^3$  of prototype 3.

Samples 11-20 belong to  $Item_{\{Good\}}^2$  of prototype 2.

Samples 21-30 belong to  $Item_{\{Good\}}^1$  of prototype 1.

Feature values are continuous number. Before using the concept of information entropy to find the feature space, we transfer continuous features to discrete features. The 30 sample feature vectors  $v_{1p}, v_{2p}, \dots, v_{30p}$  of  $F_p$  are supposed to fall into 3 compact clusters. We use

K-means Clustering algorithm to complete the process. The process can be seen in Appendix 2.

Then we construct feature space for corresponding fuzzy prototypes.

- 1). calculate the entropy in the whole training set

$$E(Item) = - \sum_{\substack{A_i \in Pro \\ u_j \in Item}} p(A_i) \log_2 p(A_i) = 1.582$$

- 2). calculate the expected entropy for the feature  $F_1$

$$EE(F_1) = \sum_{t=1}^3 p(F_1^t) E(F_1^t) = 0.945$$

calculate the Information gain for the feature  $F_1$

$$Gain(Item, F_1) = 0.637$$

- 3). calculate the expected entropy for the feature  $F_2$

$$EE(F_2) = \sum_{t=1}^3 p(F_2^t) E(F_2^t) = 1.368$$

calculate the Information gain for the feature  $F_2$

$$Gain(Item, F_2) = 0.214$$

- 4). calculate the expected entropy for the feature  $F_3$

$$EE(F_3) = \sum_{t=1}^3 p(F_3^t) E(F_3^t) = 0.602$$

calculate the Information gain for the feature  $F_3$

$$Gain(Item, F_3) = 0.980$$

- 5). calculate the expected entropy for the feature  $F_4$

$$EE(F_4) = \sum_{t=1}^3 p(F_4^t) E(F_4^t) = 0.602$$

calculate the Information gain for the feature  $F_4$

$$Gain(Item, F_4) = 0.980$$

$$\begin{aligned} Gain(Item, F_2) &< Gain(Item, F_1) < Gain(Item, F_3) \\ &= Gain(Item, F_4) \end{aligned}$$

We get the importance ranking of features

$$F_2 < F_1 < F_3 = F_4$$

We use model (1) to obtain the optimal weights of the features (assume  $orness(W) = 0.85$ ).

$$Max = - \sum_{i=1}^4 w_i \ln w_i$$

$$s.t \begin{cases} \frac{1}{4-1} \sum_{i=1}^4 (n-i) w_i = 0.85 \\ \sum_{i=1}^4 w_i = 1 \text{ where } 0 < w_i < 1 \end{cases}$$

We separate the importance ranking of features into two situations:  $F_2 < F_1 < F_3 < F_4$  and  $F_2 < F_1 < F_4 < F_3$ .

#### Situation 1

$w_1(\text{weight of } F_4) = 0.675$ ,  $w_2(\text{weight of } F_3) = 0.225$ ,

$w_3(\text{weight of } F_1) = 0.075$ ,  $w_4(\text{weight of } F_2) = 0.025$ .

<sup>2</sup>We input data which is a 150\*4 matrix contains features information of 150 samples. Then input FCM function [center,U,obj\_fcn] = fcm(data,3). We got the membership matrix U which is just the last three columns of Table 10 in Appendix 2.



## Situation 2

$w_1(\text{weight of } F_3) = 0.675, w_2(\text{weight of } F_4) = 0.225,$   
 $w_3(\text{weight of } F_1) = 0.075, w_4(\text{weight of } F_2) = 0.025.$

### The second stage:

We calculate  $G_x^k$  of  $u_x$  in the testing item set to measure the distance between every prototype.

$$G_x^k = \sum_{j=1}^n \mu_{jk} \times d(u_j, u_x) \quad (16)$$

where  $u_j \in \text{Item}_{\{Good\}}^k$ .

$$d(u_j, u_x) = \sum_{p=1}^4 w_p \cdot d(v_{xp}, v_{jp}) \quad (17)$$

We can get  $G_x^1, G_x^2$  and  $G_x^3$  for every  $u_x$ .

We assign  $u_x$  into three subsets  $\text{Item}_{\{Good\}}^a, \text{Item}_{\{Medium\}}^b$  and  $\text{Item}_{\{Poor\}}^c$ .

Then we give the analysis and comparison of this method. In the first verification, we use partial result in our model to compare the accuracy of classification between our method and Fuzzy K-means Clustering.

The process of Fuzzy K-means Clustering is as follows:

1. Use the FCM function in Matlab to generate membership set  $[\mu_{ij}]_{3 \times 150}$  (see Appendix 2).
2. Assign  $u_x$  into three subsets  $\text{Item}_{\{Good\}}^a, \text{Item}_{\{Medium\}}^b$  and  $\text{Item}_{\{Poor\}}^c$  according to membership  $\mu_{ix}$ .
  - 1).  $a = i$ , where  $\mu_{ix} = \min_{k=1,2,3}(\mu_{kx})$
  - 2).  $c = i$ , where  $\mu_{ix} = \max_{k=1,2,3}(\mu_{kx})$
  - 3). Otherwise  $b = i$ .

Table 1 The performances of the proposed method in crisp classification

	$N_{total}$	$N_{classified}$	$Accuracy_{classification}$
IPC (Situation 1)	120	112	93.3%
IPC (Situation 2)	120	107	89.17%
Fuzzy K-means Clustering	150	134	89.3%

In the second verification, we use the whole result in our model to verify the accuracy of fuzzy classification. We not only verify the accuracy of  $\text{Item}_{\{Good\}}^i$ , but also  $\text{Item}_{\{Medium\}}^i$  and  $\text{Item}_{\{Poor\}}^i$ . The reformed Iris data set is the base of comparison.

The accuracy of  $\text{Item}_{\{Medium\}}^i$  classification is calculated by function (19)

$$Accuracy_{\{Medium\}} = \frac{N_{\{Medium\}}}{N_{total\{Medium\}}} \quad (19)$$

where  $N_{\{Medium\}}$  is the number of data examples

IPC method chooses 30 examples to form the training set in the preparation phase. The rest examples in  $[\mu_{ij}]_{3 \times 150}$  form the testing set. So in Table 10, the total number of data examples in the testing set of IPC is 120, while the number of data examples in the testing set of the Fuzzy k-means approach is 150. Since the Fuzzy k-means approach here doesn't separate the training set and testing set, all examples in Appendix 2 are the data to compare with IPC.

The original Iris data set is the basic of comparison. The accuracy of classification is calculated by function (18)

$$Accuracy_{classification} = \frac{N_{classified}}{N_{total}} \quad (18)$$

where  $N_{classified}$  is the number of data examples successfully classified, and  $N_{total}$  is the total number of data examples in the testing set.

In our model,  $u_x$  is classified to set  $\text{Item}_{\{Good\}}^i$  according to  $G_x^k$ . If original Iris data set also classifies it to class  $i$ , then we think  $u_x$  is successfully classified by our method. In Fuzzy K-means Clustering,  $u_x$  is classified to  $\text{Item}_{\{Good\}}^i$  according to membership  $\mu_{ix}$ . If original Iris data set also classifies it to class  $i$ , then we think  $u_x$  is successfully classified by Fuzzy K-means Clustering. The comparison of the two classification methods is listed in Table 1.

successfully classified to  $\text{Item}_{\{Medium\}}^i$ , and  $N_{total\{Medium\}}$  is the total number of data examples in  $\text{Item}_{\{Medium\}}^i$ . In our model,  $u_x$  is classified to set  $\text{Item}_{\{Medium\}}^i$  according to  $G_x^i$ .

If reformed Iris data set also classify it to  $\text{Item}_{\{Medium\}}^i$  according to  $\mu_{ix}$ , then we think  $u_x$  is successfully classified by our model.

The definition of  $Accuracy_{\{Poor\}}$  is the same.

$$Accuracy_{\{Poor\}} = \frac{N_{\{Poor\}}}{N_{total\{Poor\}}} \quad (20)$$

The result of the accuracy of fuzzy classification by our model is listed in Table 2.

Table 2. The performances of the proposed algorithms in fuzzy classification

	$N_{total}$	$Accuracy_{\{Poor\}}$	$Accuracy_{\{Medium\}}$
Situation 1	120	100%	91.7%
Situation 2	120	100%	97.5%

As a concluding remark, this application shows that:

(1) The performance of IPC in the accuracy of classification is not poorer than Fuzzy K-means Clustering;

(2) The performance of IPC in the accuracy of fuzzy classification is also acceptable.

#### 4. An item-based recommender approach

It is a difficult problem to give a good recommendation for a new item which firstly appears in the market (e.g. movies). There are various methods to solve the item recommendation problem in RS. A common solution to this problem is to have a set of motivated users who are responsible for rating each new item in the system[8]. This kind user-based rating prediction has an important flaw: it does not consider the fact that users may use different rating values to quantify the same level of appreciation for an item [6]. So we choose item-based approaches which similar items are given the same ratings. For example, instead of consulting with my friends, when determining whether the movie “Titanic” is right for me, I consider the similar movies, like “Forrest Gump” and “Wall-E”, which have similar characters with “Titanic”. If I like these movies, then “Titanic” is right for me.

##### 4.1. The descriptor of elements in recommender system

Now we give elements in the recommender system.

(1) item profile

We use item features and evaluation as the common representations of a profile. The common representation is defined as following:

$$\text{Profile} ::= \{\text{ID}, \text{Item features}, \text{evaluation}\} \quad (21)$$

$$\text{Item features} ::= \{\text{feature}_1, \text{feature}_2, \dots, \text{feature}_n\} \quad (22)$$

$$\text{Evaluation} ::= \{\text{score}(0-5)\} \quad (23)$$

“Item feature” is a set of feature values of the item.

“Evaluation” provides a preference grade which is a relevant score.

(2) social circle prototype

Social circle prototype which is represented as  $Pro = \{A_1, A_2, A_3, \dots, A_h\}$ . Labeled Prototype,  $A_i$  is composed by a subset of items and the membership of the item belongs to the prototype. So a prototype  $A_i$  will be denoted by

$$A_i = \mu_{i1}/u_1 + \mu_{i2}/u_2 + \dots + \mu_{in}/u_n \quad (24)$$

The item  $u_j$  is represented as a fuzzy multiset:

$$u_j = \mu_{1j}/A_1 + \mu_{2j}/A_2 + \dots + \mu_{hj}/A_h \quad (25)$$

##### 4.2. Process of the new item recommendation

In the following subsections, we describe the recommendation process. Architecture of the recommendation sees Fig.4.

###### Classification phase:

Firstly, we collect the information of the new item  $u_x$ . Retrieve the feature values to construct the feature vector  $\{\text{feature}_1, \text{feature}_2, \dots, \text{feature}_n\}$ . Then compare the feature vector of the new item with  $u_j$  feature vector in prototype profile. Use item-based prototype classifier to get  $G_x^k$  of prototype  $A_k$ .

$$u_x = G_x^1/A_1 + G_x^2/A_2 + \dots + G_x^h/A_h \quad (26)$$

###### Recommendation plan generation phase:

According to function (26), we can see the new item will be liked by different social circles of users with different favorable degrees. Every social circle has its own effective advertising manner. Then we generate a recommendation plan which contains a set of advertising manners according to different social circles.

Our approach is quite different from previous ones. We use Table 3 to show the novelty of our recommendation system.

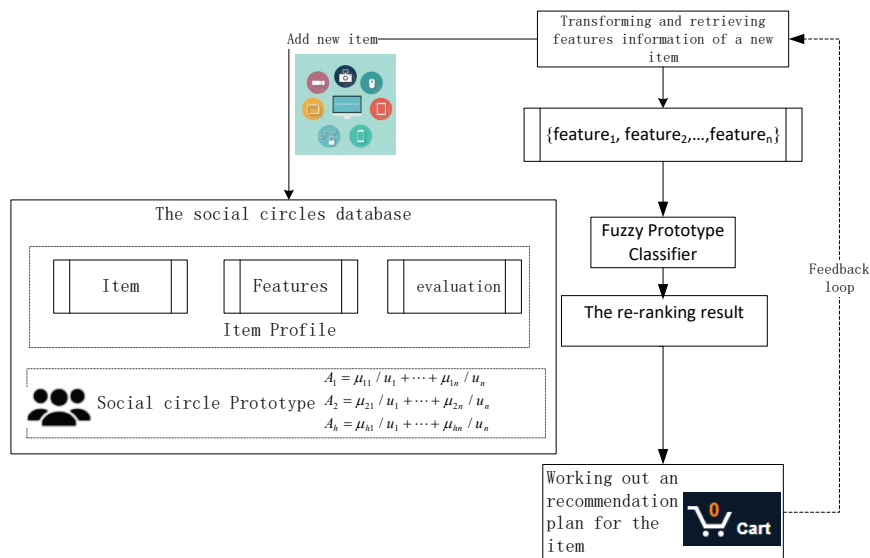


Fig.4. Architecture of the recommendation

Table 3. Comparison of approaches

	Our approach	Previous approaches
Recommendation Fundamentals	Our approach makes recommendation based on both the similarity between items and the similarity between users. We classify items as well as users.	Content-based filtering makes recommendations based on the similarity between items that a user has bought, visited, heard, viewed and ranked positively to new items. Collaborative Filtering makes recommendations to each user based on information provided by those users have the most in common with them[8].
Data quality	Not large	Large
User requirements	Evaluations given by users are fuzzy information. Users only need give a total evaluation of items, not evaluations of items' features.	Evaluations given by users are crisp information. More information about evaluations are needed.
Item requirements	Features spaces of items are not identified before recommendation process. In fact, if we choose different items to construct training data, features spaces will be different. This character makes our approach data driven recommendation.	Features spaces of items are identified before recommendation process. If new item has quite different features compared with training data, the recommendation result will not be good. This character is not good for new item cold-start.
Recommendation way	A new item is recommended to a set of users with different prototypicalities. The prototypicality shows the preference degree of the user. Even if the relation with a prototype is low, we can find the prototype is listed in the recommendation result.	A new item is recommended to a set of users. The new item is recommended to a user or not completely.

## 5. Experiments of item cold-start for MovieLens dataset

### 5.1. Data description and experimental protocol

The beginning of this section starts with an example that clearly demonstrates the new item cold-start problem. We have a RS that includes two sets: users set and items set. When a new item attends into a market, the RS needs to predict the "rating" or "preference" and recommend to the suitable users.

a) Experimental tools: we adopt Collaborative Filtering algorithms as comparative ones, because these kind algorithms have similar application scenario. We have implemented the proposed approach, NB and MF approaches to predict the "rating" of the new item.

b) Experimental datasets: To perform experimental evaluations of the proposed ideas we use a benchmark dataset: MovieLens. This dataset (ml-20m) describes 5-star rating and free-text tagging activity from [MovieLens] (<http://movielens.org>), a movie recommendation service. The data are contained in six files, 'genome-scores.csv', 'genome-tags.csv', 'links.csv', 'movies.csv', 'ratings.csv' and 'tags.csv'. We extract only positive ratings (rating 5) for training and testing. We choose MovieID 1-20 to compose our training set and randomly choose 16 movies besides these 20 movies to make up the new items testing set. The only requirement of the testing set is that these movies also have rating data by the users in the training set and do not contain features which are not in Table 5. The characteristics of the dataset are summarized in Table 4.

c) Evaluation indices: we use three widely used measures to compare the recommendation results, namely Mean Reciprocal Rank (MRR), Mean Absolute Error (MAE) and Root Mean Square Error (RMSE).

- **MRR:** The Reciprocal Rank (RR) for a recommendation list is the multiplicative inverse of the rank of the first "good" item.

$$MRR = \frac{1}{n} \sum_i \frac{1}{rank_i} \quad (27)$$

where  $n$  is the number of users who receive recommendations, i.e., the number of recommendation lists, and  $rank_i$  is the rank of the first correct item in therecommendation list of user  $i$ .

- **MAE and RMSE**

$$MAE = \frac{1}{n} \sum_{u,i} |p_{u,i} - r_{u,i}| \quad (28)$$

$$RMSE = \sqrt{\frac{1}{n} \sum_{u,i} (p_{u,i} - r_{u,i})^2} \quad (29)$$

where  $p_{u,i}(r_{u,i})$  is the predicted (real)rating of user  $u$  for item  $i$ .

Table 4. Information about used data sets

	Training set	Testing set
Users	465	4145
Movies	20	16
Ratings <sup>3</sup>	100%	100%
Sparsity <sup>4</sup>	100%	100%

### 5.2. Empirical results and discussion

First, we state the processes of the proposed, NB and MF approaches. Then we present the three recommendation results of the approaches. Finally, we give analysis and comparison.

- The item-based recommender approach

Users are grouped into several social circles. A social circle is a labeled prototype  $A_i$  which is described by a subset of items and the memberships of the items belonging to the prototype. We solve the new item cold-start problem by identifying a feature space of the prototype. Given a new item and its corresponding features, the distance measure between it and a prototype for the new item is computed. Then the ranking of prototypicalities is obtained for this new item.

In the first stage, we use a training set to discover feature space for each group users. Through data arranging, we extract 10 features of these 20 movies from 'movies.csv' and obtain Table 5 to describe features. We analyze the data in 'ratings.csv' and classify these 20 movies into *Group*<sup>1</sup> or *Group*<sup>2</sup> according to the 5-star rating. Our simplified assumption is that the users giving a same movie 5-star are classified into one group, and a movie liked by one member of the

<sup>3</sup>"Ratings" is expressed as a percentage of the number of movies with 5-stars to the one of all movies. Because every movie in the training set or testing set contains at least one user who give 5-stars to it, the ratings value is 100%.

<sup>4</sup>"Sparsity" describes the sparsity problem in recommender systems. It is expressed as a percentage of the number of users who give 5-stars to movies in the training set or testing set to the one of all users in the training set or testing set. Because every user in the training set or testing set gives at least one movie 5-star, the Sparsity value is 100%.

group will be also liked by other member. Then we get Table 6 to describe information of the training set. In Table6, every movie is classified to a group with high membership which is related with the user's appearance frequency<sup>5</sup>. We also use this process to obtain the membership of movie in testing set. More the members in  $Group^p$  give 5-stars to Movie  $k$ , bigger the membership value of the movie to the group is. According to the membership of movie, we give the ranking as the real rating of user  $i$  for item  $k$ . Higher membership achieves a higher ranking.

The process as follows:

Step1. Choose randomly 20 users from 465 users who gave 5 stares to Movie 1-20.

Step2. User  $i$  has a favorite movie set which is  $U_i = \{u_{i1}u_{i2}, \dots, u_{ij}\}$ .  $u_{ik}$  is the movie that user  $i$  gave 5-stars. Since there are 20 users, there are 20 favorite movie sets, which are  $U_1, U_2, \dots, U_{20}$ .

Step3. If the movie  $u_k$  is in both  $U_i$  and  $U_j$ , then we put user  $i$  and  $j$  in a group. 20 users are grouped in  $Group^1$  or  $Group^2$ .

Step4. Classify movies into  $Group^1$  or  $Group^2$  according to the 5-star rating.

Step5. Set the basic Membership of movie  $k$  belonging to  $Group^p$ ,  $\mu_{pk} = 0.5$ .

Step6. Count the occurrences  $m$  of users who gave movie  $k$  5-stars in  $Group^p$ .

$$\mu_{pk} = 0.5 + m \times 0.05 \quad (30)$$

<sup>5</sup>We used different approaches to obtain membership values to prototypes in Iris example and Movielens case. There are two reasons. The first one is that the amount of Movielens examples is so less than that of Iris example. FCM function in Matlab is good at handling large amounts of data. The samples in Movielens case are not enough to obtain good result when we use FCM function. The second one is that the scenarios of these two data sets are different. Iris dataset is for crisp classifications. Every example in the data set completely belongs to one class. Iris data set in UCI is reformed to fuzzy class data set which allows the fuzzy prototype model to classify each data example as a certain class. We used FCM function to obtain membership values and the original classification can validate its correctness. But the situation in Movielens case is different. The original data set does not tell us the classification of every movie. So we can not validate the correctness of FCM in this case. On the other hand, the process we added conforms to our intuitions that movies liked by more members in a group have more representativeness of this group. Our added process is just to realize this rule.

Step7. The group  $Group^p$  is denoted by

$$Group^p = \mu_{p_1}/u_{p_1} + \mu_{p_2}/u_{p_2} + \dots + \mu_{p_n}/u_{p_n}$$

Calculate the information gain for every feature and get the ranking of features. Use the model (1) to obtain the optimal weights of the features (assume  $ess(W) = 0.85$ ). The result is described in Table 7.

Table 5. Feature description

F <sub>1</sub>	F <sub>2</sub>	F <sub>3</sub>	F <sub>4</sub>	F <sub>5</sub>
Adventure	Animation	Children	Comedy	Fantasy
F <sub>6</sub>	F <sub>7</sub>	F <sub>8</sub>	F <sub>9</sub>	F <sub>10</sub>
Romance	Drama	Action	Crime	Thriller

Table 6. Information of training data

MovieId	GroupID	Membership( $\mu_i$ )
1	1	0.95
2	1	0.9
3	2	0.95
4	2	0.8
5	2	0.8
6	1	0.85
7	2	0.8
8	1	0.8
9	2	0.85
10	1	0.85
11	1	0.9
12	1	0.8
13	1	0.9
14	1	0.8
15	1	0.95
16	1	0.8
17	2	0.9
18	1	0.95
19	1	0.95
20	1	0.95

Table7. Features weights

	F <sub>7</sub>	F <sub>1</sub>	F <sub>6</sub>	F <sub>3</sub>	F <sub>10</sub>
$EE(F_i)$	0.387	0.133	0.116	0.096	0.082
$W_i$	0.419	0.244	0.142	0.083	0.048
	F <sub>9</sub>	F <sub>2</sub>	F <sub>5</sub>	F <sub>4</sub>	F <sub>8</sub>
$EE(F_i)$	0.059	0.038	0.038	0.025	0.019
$W_i$	0.028	0.016	0.010	0.006	0.003

In the second stage, we use the prediction method to recommend a new movie  $u_x$ . We calculate  $G_x^k$  of the 16 movies to measure the distance between two groups (See Table 8).

$$G_x^k = \sum_{j=1}^n \mu_j \times d(u_j, u_x) \quad (31)$$

where  $u_j \in \text{Group}^k, k = 1, 2$ .

$$d(u_j, u_x) = \sum_{p=1}^{10} w_p \cdot d(v_{xp}, v_{jp}) \quad (32)$$

In the experiments, we assume

(a) If  $G_x^1 < G_x^2$ , then we conclude that  $u_x$  is more preferred by group 1. In 'ratings.csv', the movie should be given 5 stars by the users in group 1.

(b) If  $G_x^1 > G_x^2$ , then we conclude that  $u_x$  is more preferred by group 2. In 'ratings.csv', the movie should be given 5 stars by the users in group 2.

We give the  $\text{Rank}_i$  according to the rules:

If  $\frac{G_x^1}{G_x^1 + G_x^2} > \frac{G_y^1}{G_y^1 + G_y^2}$ , then  $\text{Rank}_i < \text{Rank}_j$ ;

If  $\frac{G_x^1}{G_x^1 + G_x^2} = \frac{G_y^1}{G_y^1 + G_y^2}$ , then  $\text{Rank}_i = \text{Rank}_j$ ;

If  $\frac{G_x^1}{G_x^1 + G_x^2} < \frac{G_y^1}{G_y^1 + G_y^2}$ , then  $\text{Rank}_i > \text{Rank}_j$ ;

Fig.5.<sup>6</sup> presents the calculating results of our experiment.

Table 8. Results of prediction

NO.(x)	MovieID	$G_x^1$	$G_x^2$	$\text{Rank}_1$	$\text{Rank}_2$
1	88	0.537	0.685	2	14
2	112	0.373	0.558	1	15
3	185	0.357	0.364	3	11
4	1036	0.357	0.364	3	11
5	2278	0.357	0.364	3	11
6	62	0.501	0.439	10	6
7	318	0.516	0.467	8	7
8	296	0.538	0.513	6	10
9	52	0.603	0.39	14	3
10	34	0.513	0.413	11	5
11	39	0.423	0.238	15	1
12	28	0.602	0.392	12	4
13	36	0.516	0.467	8	7
14	52	0.603	0.390	13	3
15	39	0.423	0.238	15	1
16	514	0.311	0.284	7	9

<sup>6</sup> Series 3,4 present the evaluations of users in 'ratings.csv'. If this movie is given 5 stars by users in  $\text{Group}^1$ , the value of series 3 is 1, otherwise is 0. If this movie is given 5 stars by users in  $\text{Group}^2$ , the value of series 4 is 1, otherwise is 0.

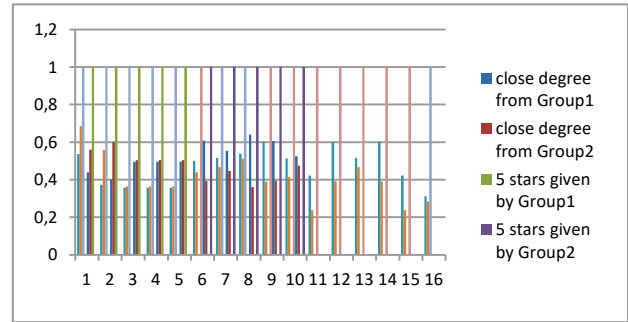


Fig.5. Prediction of our recommendation process and original recommendation results of the dataset

(ii) The NB approach and MF approach

In NB approach, recommendations can be done in two ways known as *user-based* or *item-based* recommendation. We choose *item-based* recommendation as the comparative approach. Item-based approaches predict the rating of a user  $u$  for a new item based on the ratings of  $u$  for items similar to the new one. This idea can be formalized as follows. Denote by  $\mathcal{N}_u(i)$  the items set rated by user  $u$  most similar to item  $i$ . The predicted rating of user  $u$  for item  $i$  is obtained as a weighted average of the ratings given by  $u$  to the items of  $\mathcal{N}_u(i)$ :

$$\hat{r}_{ui} = \frac{\sum_{j \in \mathcal{N}_u(i)} w_{ij} r_{uj}}{\sum_{j \in \mathcal{N}_u(i)} |w_{ij}|} \quad (33)$$

where  $w_{i,j}$  reflects the character similarity between  $i$  and  $j$ .  $r_{uj}$  reflects the preference of user  $u$  to item  $i$ .

In MF approach, recommendations are made by comparing the properties of the new items to the properties of those items that are known to be liked by representative users (RU). Rating value which measures the preference of the new item liked by RU is calculated in function (34).

$$r_{i,j} = \sum_{k=1}^{K-K} w_{k,i} v_{k,j} \quad (34)$$

where  $r_{i,j}$  reflects the item  $j$  relation with user  $u_i$ . The vector  $w_{*,i}$  is a  $K$ -dimensional representation of a user  $u_i$  (that is representation of a user  $u_i$  in the space of latent features), and the vector  $v_{*,j}$  is a  $K$ -dimensional representation of the item  $i_j$  (that is representation of an item  $i_j$  in the same space of latent features). The set of features is interpreted as a set of representative users. Therefore, value  $v_{k,j}$  expresses preferences of a representative user  $k$  on the item  $j$  and  $w_{k,i}$ —closeness of the user  $u_i$  to the representative user  $k$ .

We normalize  $r_{i,j}(\hat{r}_{u,i})$  to get  $r_{i,j}'(\hat{r}_{u,i}')$



$$r_{i,j}' = \frac{r_{i,j}}{\sum_{l=1}^M r_{i,l}} \quad (35)$$

$$\hat{r}_{u,i}' = \frac{\hat{r}_{u,i}}{\sum_{l=1}^M \hat{r}_{u,l}} \quad (36)$$

We give the  $Rank_i$  according to the rules:

If  $r_{i,i}' > r_{i,j}'$ , then  $Rank_i < Rank_j$ ;

If  $r_{i,i}' = r_{i,j}'$ , then  $Rank_i = Rank_j$ ;

If  $r_{i,i}' < r_{i,j}'$ , then  $Rank_i > Rank_j$ .

Fig.6. Fig.7. present the calculating results of NB and MF approach experiment.

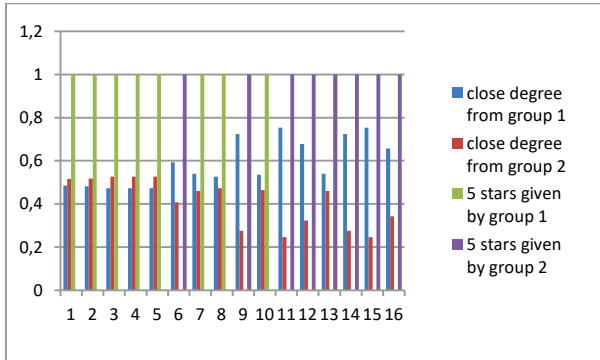


Fig.6. Prediction of the NB recommendation process and original recommendation results of the dataset

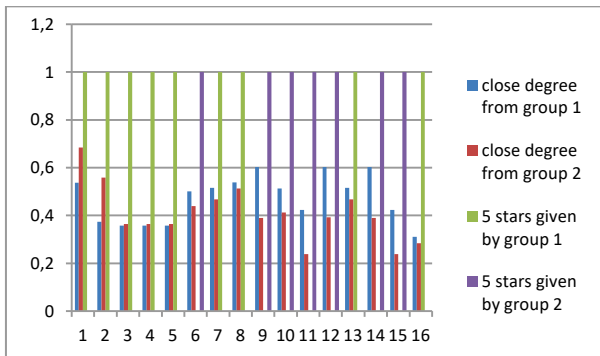


Fig.7. Prediction of the MF recommendation process and original recommendation results of the dataset

### (iii) Analysis and comparison

First, we compare the three approaches in terms of evaluation indices. From Fig.5, Fig.6 and Fig.7, we can see these models give the similar recommendation to the users who just give 5 stars to these movies. Predictions of these models are acceptable. Deeper comparison can be made through MAE and RMSE. From Table 9, we can see the MRR, MAE and RMSE of these approaches. The experimental results have shown that our approach obtains better accuracy than NB and MF approaches in terms of the MAE and RMSE.

Table.9 MAE and RMSE of these three approaches

	Our approach		The NB approach		The MF approach	
	Group <sup>1</sup>	Group <sup>2</sup>	Group <sup>1</sup>	Group <sup>2</sup>	Group <sup>1</sup>	Group <sup>2</sup>
MRR	0.317	0.299	0.410	0.297	0.410	0.297
MAE	1.81	1.88	2.94	2.56	2.94	2.56
RMSE	2.68	2.69	3.68	3.21	3.68	3.21

Then we compare the three approaches in another aspect. Following the general tendency of the cold-start problem solution in CF, there are two major approaches in CF: neighborhood-based (NB) and matrix factorization (MF)[9]. While user-based NB methods rely on the opinion of like-minded users to predict a rating, item-based NB approaches look at ratings given to similar items[6]. The MF approach relies on the opinion that the ratings in the rating matrix can be explained by a small number of latent features[9]. The ideas of these three approaches are similar. The differences among these approaches are how to construct the mapping function. NB approaches use similar items to represent new item. Sparsity problem is obvious. We find the similar items to the new item. But if the user did not give any rates to these similar items, then we can not give accurate recommendation. MF approach makes improvement. MF approach searches seed users, which can represent the interests of the rest of the users and asking their opinion on new items. The key idea of solving the new item cold-start problem is to find a mapping function between latent features and item attributes (from the learning set). Our approach combines features of the two. We present two contributions.

- First, we propose a new way to interpret relations between items and users in RS. We point out that the relation is fuzzy. Preferences of users are presented by memberships. Any new item will be recommended to users, even if the membership is low.
- Secondly, we use reality concept to interrupt MF. Social circle can be viewed as representative users in Matrix factorization. And feature space can be viewed as latent features in Matrix factorization. But our approach improves the representation model of seed users. We apply prototype theory to this problem.

## 6. Conclusion

The goal of our paper is to solve the new item recommendation problem. Assume combination of

several items can reflect different aspects of a social circle. So our prototype structure based on item is an innovation. In our prototype structure, a category is not discriminated by features. It is discriminated by items. Because users do not know well the feature's role in deciding their preferences. This paper applies the novel classification model into recommendation approach which employs item-based prototype classifier to find the best recommendation plans. We believe the prototype structure presented in this paper opens a new situation in knowledge management and also in the recommendation support systems.

## Acknowledgements

This work was supported by the National Natural Science Foundation of China (NSFC) (71401078,71571104), the Natural Science Foundation of Jiangsu (BK20141481) and MOE (Ministry of Education in China) Youth Project of Humanities and Social Sciences (14YJC630072).

## Appendix 1.

Given a set of training samples  $(v_{1p}, v_{2p}, \dots, v_{yp})$ , The K-means Clustering [42] algorithm is composed of the following steps:

1. Given an initial set of k means  $\{m_p^{1(1)}, m_p^{2(1)}, \dots, m_p^{a(1)}\}$ . Let  $m_p^{l(1)}$  be the mean of the cluster  $Item_l^{(1)}$ . We assign  $v_{jp}$  into set  $Item_l$  whose centroid is  $m_p^{l(1)}$ .  $t = 1$ .
2.  $t = t + 1$ . Assign each of training samples to the cluster whose mean yields the least within-cluster sum. Each  $v_{jp}$  is assigned to exactly one  $Item_l^{(t)}$

$$Item_l^{(t)} = \{v_{jp} : \|v_{jp} - m_p^{l(t)}\| \leq \|v_{jp} - m_p^{i(t)}\| \quad \forall j, 1 \leq i \leq a\}$$

3. When all objects have been assigned, recalculate the positions of the K centroids. the new means to be the centroids of the training samples in the new clusters is

$$m_p^{l(t+1)} = \frac{1}{|Item_l^{(t)}|} \sum_{x_j \in Item_l^{(t)}} v_{jp}$$

4. Repeat Steps 2 and 3 until the centroids no longer move. This produces a separation of the objects into groups from which the metric to be minimized can be calculated.

## Appendix 2.

Table 10. The reformed Iris data set by Fuzzy K-means Clustering

No.	$\mu_{1j}$	$\mu_{2j}$	$\mu_{3j}$
1	0.0012	0.0025	0.9963
2	0.0072	0.0159	0.9770
3	0.0063	0.0134	0.9803

4	0.0098	0.0219	0.9683
5	0.0019	0.0041	0.9939
6	0.0211	0.0458	0.9331
7	0.0065	0.0140	0.9795
8	0.0001	0.0003	0.9996
9	0.0215	0.0469	0.9316
10	0.0050	0.0113	0.9837
11	0.0105	0.0223	0.9672

12	0.0024	0.0053	0.9923
13	0.0088	0.0194	0.9717
14	0.0250	0.0513	0.9237
15	0.0381	0.0735	0.8884
16	0.0550	0.1057	0.8393
17	0.0180	0.0365	0.9455
18	0.0012	0.0025	0.9963
19	0.0306	0.0663	0.9031
20	0.0070	0.0149	0.9781
21	0.0095	0.0218	0.9688
22	0.0051	0.0110	0.9839
23	0.0140	0.0279	0.9581
24	0.0060	0.0142	0.9798
25	0.0097	0.0230	0.9673
26	0.0076	0.0175	0.9749
27	0.0016	0.0036	0.9949
28	0.0022	0.0047	0.9931
29	0.0020	0.0043	0.9937
30	0.0061	0.0137	0.9802
31	0.0062	0.0141	0.9798
32	0.0080	0.0178	0.9743
33	0.0209	0.0422	0.9369
34	0.0330	0.0647	0.9024
35	0.0050	0.0113	0.9837
36	0.0047	0.0099	0.9854
37	0.0117	0.0245	0.9638
38	0.0050	0.0113	0.9837
39	0.0190	0.0405	0.9406
40	0.0005	0.0011	0.9983
41	0.0018	0.0038	0.9944
42	0.0464	0.1009	0.8527
43	0.0151	0.0319	0.9531
44	0.0065	0.0147	0.9789
45	0.0169	0.0389	0.9442
46	0.0083	0.0185	0.9732
47	0.0077	0.0165	0.9758
48	0.0080	0.0173	0.9748
49	0.0075	0.0161	0.9764
50	0.0008	0.0018	0.9974

51	0.5011	0.4543	0.0446
52	0.2068	0.7641	0.0292
53	0.5999	0.3688	0.0313
54	0.0805	0.8700	0.0495
55	0.2171	0.7588	0.0241
56	0.0205	0.9738	0.0058
57	0.2972	0.6730	0.0298
58	0.1321	0.5815	0.2864
59	0.2477	0.7210	0.0313
60	0.0947	0.8303	0.0750
61	0.1449	0.6357	0.2194
62	0.0287	0.9621	0.0092
63	0.1011	0.8430	0.0558
64	0.0882	0.8997	0.0121
65	0.0922	0.8159	0.0919
66	0.2684	0.6899	0.0417
67	0.0527	0.9331	0.0142
68	0.0484	0.9256	0.0260
69	0.1374	0.8354	0.0272
70	0.0708	0.8775	0.0517
71	0.2507	0.7217	0.0276
72	0.0462	0.9342	0.0195
73	0.2706	0.7054	0.0240
74	0.0833	0.9028	0.0139
75	0.1012	0.8759	0.0229
76	0.2112	0.7548	0.0340
77	0.4427	0.5237	0.0336
78	0.6725	0.3063	0.0212
79	0.0262	0.9689	0.0049
80	0.1046	0.7666	0.1288
81	0.0899	0.8320	0.0781
82	0.1007	0.7951	0.1042
83	0.0503	0.9186	0.0311
84	0.3197	0.6563	0.0240
85	0.0825	0.8911	0.0264
86	0.1707	0.7972	0.0321
87	0.4111	0.5554	0.0335
88	0.1155	0.8576	0.0269
89	0.0470	0.9289	0.0241

90	0.0624	0.8993	0.0383
91	0.0493	0.9311	0.0197
92	0.0727	0.9157	0.0116
93	0.0418	0.9356	0.0226
94	0.1326	0.5973	0.2701
95	0.0285	0.9588	0.0127
96	0.0377	0.9455	0.0168
97	0.0230	0.9674	0.0096
98	0.0447	0.9439	0.0114
99	0.1243	0.5190	0.3568
100	0.0268	0.9605	0.0127
101	0.8599	0.1207	0.0194
102	0.3551	0.6156	0.0293
103	0.9558	0.0382	0.0061
104	0.8455	0.1419	0.0125
105	0.9576	0.0376	0.0048
106	0.8118	0.1527	0.0355
107	0.1670	0.7598	0.0731
108	0.8629	0.1152	0.0219
109	0.8686	0.1173	0.0140
110	0.8610	0.1146	0.0244
111	0.7734	0.2099	0.0168
112	0.7611	0.2231	0.0158
113	0.9888	0.0100	0.0012
114	0.3057	0.6599	0.0344
115	0.5006	0.4610	0.0384
116	0.8499	0.1361	0.0141
117	0.9132	0.0796	0.0072
118	0.7636	0.1858	0.0506
119	0.7575	0.1933	0.0492
120	0.2571	0.7107	0.0323

121	0.9705	0.0257	0.0038
122	0.2594	0.7069	0.0337
123	0.7832	0.1748	0.0420
124	0.3814	0.5958	0.0228
125	0.9753	0.0217	0.0029
126	0.9119	0.0752	0.0129
127	0.2712	0.7078	0.0210
128	0.3300	0.6469	0.0231
129	0.9086	0.0830	0.0084
130	0.8908	0.0948	0.0145
131	0.8727	0.1075	0.0198
132	0.7601	0.1890	0.0509
133	0.9062	0.0848	0.0090
134	0.4364	0.5402	0.0234
135	0.5761	0.3927	0.0312
136	0.8395	0.1318	0.0287
137	0.8541	0.1287	0.0172
138	0.8802	0.1101	0.0098
139	0.2283	0.7499	0.0217
140	0.9676	0.0290	0.0035
141	0.9573	0.0377	0.0051
142	0.8551	0.1295	0.0154
143	0.3551	0.6156	0.0293
144	0.9610	0.0337	0.0053
145	0.9271	0.0632	0.0097
146	0.8824	0.1064	0.0113
147	0.4667	0.5075	0.0258
148	0.8315	0.1564	0.0121
149	0.7894	0.1890	0.0216
150	0.3913	0.5818	0.0269

## Reference:

1. A. Tejada-Lorente, C. Porcel, J. Bernabé-Moreno and E. Herrera-Viedma, REFORE: A recommender system for researchers based on bibliometrics, *Applied Soft Computing*, 30 (2015) 778-791.
2. S.-M. Chen, S.-H. Cheng and T.-E. Lin, Group decision making systems using group recommendations based on interval fuzzy preference relations and consistency matrices, *Information Sciences*, 298 (2015) 555-567.
3. A. Zenebe and A.F. Norcio, Representation, similarity measures and aggregation methods using fuzzy sets for content-based recommender systems, *Fuzzy Sets and Systems*, 160 (2009) 76-94.
4. Y. Blanco-Fernandez, M. Lopez-Nores, A. Gil-Solla, M. Ramos-Cabrer and J.J. Pazos-Arias, Exploring synergies between content-based filtering and Spreading Activation techniques in knowledge-based recommender systems, *Information Sciences*, 181 (2011) 4823-4846.
5. Y. Cai, H.-f. Leung, Q. Li, H. Min, J. Tang and J. Li, Typicality-Based Collaborative Filtering Recommendation, *IEEE Transactions on Knowledge and Data Engineering*, 26 (2014) 766-779.

6. C. Desrosiers and G. Karypis, A Comprehensive Survey of Neighborhood-based Recommendation Methods, in: F. Ricci, L. Rokach, B. Shapira, P.B. Kantor (Eds.) *Recommender Systems Handbook*, Springer US, Boston, MA, 2011, pp. 107-144.
7. J. Wei, J. He, K. Chen, Y. Zhou and Z. Tang, Collaborative filtering and deep learning based recommendation system for cold start items, *Expert Systems with Applications*, 69 (2017) 29-39.
8. J. Bobadilla, F. Ortega, A. Hernando and A. Gutiérrez, Recommender systems survey, *Knowledge-Based Systems*, 46 (2013) 109-132.
9. M. Aleksandrova, A. Brun, A. Boyer and O. Chertov, Identifying representative users in matrix factorization-based recommender systems: application to solving the content-less new item cold-start problem, *Journal of Intelligent Information Systems*, (2016) 1-33.
10. C.C. Chen, Y.-H. Wan, M.-C. Chung and Y.-C. Sun, An effective recommendation method for cold start new users using trust and distrust networks, *Information Sciences*, 224 (2013) 19-36.
11. S. Nepal, C. Paris, P.A. Pour, J. Freyne and S.K. Bista, Interaction Based Content Recommendation in Online Communities, in: S. Carberry, S. Weibelzahl, A. Micarelli, G. Semeraro (Eds.) *User Modeling, Adaptation, and Personalization: 21th International Conference, UMAP 2013, Rome, Italy, June 10-14, 2013 Proceedings*, Springer Berlin Heidelberg, Berlin, Heidelberg, 2013, pp. 14-24.
12. R. Yera, J. Castro and L. Martinez, A fuzzy model for managing natural noise in recommender systems, *Applied Soft Computing*, 40 (2016) 187-198.
13. D. Sun, Z. Luo and F. Zhang, A novel approach for collaborative filtering to alleviate the new item cold-start problem, in: *International Symposium on Communications and Information Technologies*, 2011, pp. 402-406.
14. O. Anava, S. Golan, N. Golbandi, Z. Karnin, R. Lempel, O. Rokhlenko and O. Somekh, Budget-Constrained Item Cold-Start Handling in Collaborative Filtering Recommenders via Optimal Design, *Computer Science*, (2014).
15. G.B. Davis and K.M. Carley, Clearing the FOG: Fuzzy, overlapping groups for social networks, *Social Networks*, 30 (2008) 201-212.
16. M. Scholz, V. Dorner, G. Schryen and A. Benlian, A configuration-based recommender system for supporting e-commerce decisions, *European Journal of Operational Research*, 259 (2017) 205-215.
17. S. Kaisler, F. Armour, J.A. Espinosa and W. Money, Big Data: Issues and Challenges Moving Forward, in: *2013 46th Hawaii International Conference on System Sciences*, 2013, pp. 995-1004.
18. L.A. Zadeh, A note on prototype theory and fuzzy sets, *Cognition*, 12 (1982) 291-297.
19. F.M. Schleif, T. Villmann and B. Hammer, Prototype based fuzzy classification in clinical proteomics, *International Journal of Approximate Reasoning*, 47 (2008) 4-16.
20. J. Derrac, S. Garcia and F. Herrera, Fuzzy nearest neighbor algorithms: Taxonomy, experimental analysis and prospects, *Information Sciences*, 260 (2014) 98-119.
21. A. Bahrololoum, H. Nezamabadi-pour, H. Bahrololoum and M. Saeed, A prototype classifier based on gravitational search algorithm, *Applied Soft Computing*, 12 (2012) 819-825.
22. R. García-Durán, F. Fernández and D. Borrajo, A prototype-based method for classification with time constraints: a case study on automated planning, *Pattern Anal Applic*, 15 (2012) 261-277.
23. F. Fernández and P. Isasi, Evolutionary Design of Nearest Prototype Classifiers, *Journal of Heuristics*, 10 (2004) 431-454.
24. N. Verbiest, C. Cornelis and F. Herrera, FRPS: A Fuzzy Rough Prototype Selection method, *Pattern Recognition*, 46 (2013) 2770-2782.
25. Z. Deng, Y. Jiang, F.-L. Chung, H. Ishibuchi, K.-S. Choi and S. Wang, Transfer Prototype-Based Fuzzy Clustering, *IEEE Transactions on Fuzzy Systems*, 24 (2016) 1210-1232.
26. E. Pekalska, R.P.W. Duin and P. Paclik, Prototype selection for dissimilarity-based classifiers, *Pattern Recognition*, 39 (2006) 189-208.
27. R. Yera and L. Martinez, Fuzzy Tools in Recommender Systems: A Survey, *International Journal of Computational Intelligence Systems*, 10 (2017) 776-803.
28. S. Impedovo, F.M. Mangini and D. Barbuizi, A novel prototype generation technique for handwriting digit recognition, *Pattern Recognition*, 47 (2014) 1002-1010.
29. P. D'Urso and R. Massari, Fuzzy clustering of human activity patterns, *Fuzzy Sets and Systems*, 215 (2013) 29-54.
30. F.P. Romero, I. Caballero, J. Serrano-Guerrero and J.A. Olivas, An approach to web-based Personal Health Records filtering using fuzzy prototypes and data quality criteria, *Information Processing & Management*, 48 (2012) 451-466.
31. Y. Liao and V.R. Vemuri, Use of K-Nearest Neighbor classifier for intrusion detection, *Computers & Security*, 21 (2002) 439-448.
32. J.-H. Chen, H.-M. Chen and S.-Y. Ho, Design of nearest neighbor classifiers: multi-objective approach, *International Journal of Approximate Reasoning*, 40 (2005) 3-22.
33. A. Arami and B. Nadjar Araabi, A Clustering Method Based on Soft Learning of Model (Prototype) and Dissimilarity Metrics, in: *13th International CSI Computer Conference (CSICC)*, Springer-Verlag, Kish Island, Iran, 2009, pp. 33-40.
34. B. Hammer, B. Mokbel, F.-M. Schleif and X. Zhu, Prototype-Based Classification of Dissimilarity Data, in: J. Gama, E. Bradley, J. Hollmén (Eds.) *Advances in Intelligent Data Analysis X*, Springer Berlin Heidelberg, 2011, pp. 185-197.
35. S.R. Kheradpisheh, F. Behjati-Ardakani and R. Ebrahimpour, Combining classifiers using nearest decision prototypes, *Applied Soft Computing*, 13 (2013) 4570-4578.
36. L. Fischer, B. Hammer and H. Wersing, Efficient rejection strategies for prototype-based classification, *Neurocomputing*, 169 (2015) 334-342.
37. J. Chen, Y. Yan Tang, C.L. Philip Chen, B. Fang, Z. Shang

- and Y. Lin, NNMap: A method to construct a good embedding for nearest neighbor classification, *Neurocomputing*, 152 (2015) 97-108.
38. M.J. Barranco, L. Martinez, A method for weighting multi-valued features in content-based filtering, in: *International Conference on Industrial Engineering and Other Applications of Applied Intelligent Systems*, 2010, pp. 409-418.
39. R.R. Yager, On orderd weighted averaging aggregation operators in multicriteria decision-making, *IEEE Transactions on Systems Man and Cybernetics*, 18 (1988) 183-190.
40. R.R. Yager, On the dispersion measure of OWA operators, *Information Sciences*, 179 (2009) 3908-3919.
41. M. O'Hagan, Aggregating Template Or Rule Antecedents In Real-time Expert Systems With Fuzzy Set Logic, in: *Signals, Systems and Computers*, 1988. Twenty-Second Asilomar Conference on, 1988, pp. 681-689.
42. J. MacQueen, Some methods for classification and analysis of multivariate observations, in: L.M. Le Cam, J. Neyman (Eds.) *Proceedings of the Fifth Berkeley Symposium on Mathematical Statistics and Probability. Volume I: Statistics*, University of California Press, 1967, pp. 281-297.