Managing Natural Noise in Recommender Systems

Luis Martínez*[1], Jorge Castro[23], and Raciel Yera[4]

[1] Department of Computer Science, University of Jaén, Spain
[2] Department of Computer Science and A.I., University of Granada, Spain
[3] Centre for Quantum Computing and Intelligent Systems, University of Technology Sydney, Australia
[4] University of Ciego de Ávila, Ciego de Ávila, Cuba

Abstract. E-commerce customers demand quick and easy access to suitable products in large purchase spaces. To support and facilitate this purchasing process to users, recommender systems (RSs) help them to find out the information that best fits their preferences and needs in an overloaded search space. These systems require the elicitation of customers' preferences. However, this elicitation process is not always precise either correct because of external factors such as human errors, uncertainty, human beings inherent inconsistency and so on. Such a problem in RSs is known as *natural noise* (NN) and can negatively bias recommendations, which leads to poor user's experience. Different proposals have been presented to deal with natural noise in RSs. Several of them require additional interaction with customers. Others just remove noisy information. Recently, new NN approaches dealing with the ratings stored in the user/item rating matrix have raised to deal with NN in a better and simpler way. This contribution is devoted to provide a brief review of the latter approaches revising crisp and fuzzy approaches for dealing with NN in RSs. Eventually it points out as a future research the management of NN in other recommendation scenarios as group RSs.

Keywords: recommender systems, natural noise, fuzzy logic, computing with words, group recommender systems

1 Introduction

The development of the e-commerce has made available huge information amounts of interest for customers in their purchasing processes. With large amount of options, customers usually cannot filter them effectively, hence frustration and early purchasing leaving can happen. Therefore, the support to finding out quick and easily items that fits customers' preferences and needs become an important challenge nowadays. Recommender systems (RSs) are the most successful tool for supporting personalized recommendations [1, 2]. RSs have been broadly used in different scenarios like e-commerce [3], e-learning [4], tourism [5], and so on.

* martin@ujaen.es

Different approaches have been used in RSs, being the content-based (CBRS) [6, 7] and the collaborative filtering (CFRS) [1] the most widespread. CBRS methods are based on items' descriptions to generate the users' recommendations, meanwhile CFRS have performed this task just using users ratings about items. The latter approach is the most popular one in real world RSs because of their good performance even when items descriptions are not available. But, the necessity of customers preferences in CFRSs has produced some problems that limit their performance, such as *cold start* and *sparsity* [1, **?**], and more recently new related problems regarding the quality of the rating data have raised up [8–10]. Specifically, Ekstrand et al. [11] pointed out that the rating elicitation process is not error-free, hence the ratings can contain noise. They mentioned that such a noise, previously coined *natural noise* (NN) in [12], could be caused by human error, mixing of factors in the rating process, uncertainty and other factors. They stated that its detection and correction should provide more accurate recommendations.

The main research stream in RSs has been and still is the development of algorithms that increase the accuracy of the recommendations. Due to the fact that NN biases the recommendations and affect negatively to the RSs performance, it is worthy to research its management for improving the recommendations. For this reason, different proposals in the literature have introduced processes for managing the errors of rating elicitation in recommendation scenarios. The management of NN depends on the data available and some approaches require additional user interactions [9] and others just remove noisy data either ratings or users [13]. Recently, however, new proposals manage the NN by using the current rating values in the user/item rating matrix while keeping as much information as possible and without any new user's interaction [10, 14].

This contribution aims at providing a brief review about the latter approaches for managing NN, which only need current ratings in the RS, by revising both crisp and fuzzy approaches. Additionally, it points out future research challenges.

This contribution is structured as follows. Section 2 provides a brief background for understanding RSs and NN. Section 3 provides a view about the basic model to manage NN with the information stored in the user/item rating matrix. Section 4 focuses on a fuzzy extension of the previous approach for dealing with natural noise in CFRS. Eventually, Section 5 points out the application of NN management in group recommendation scenarios.

2 Preliminaries

This section provides the required background for the current research, including basics about CFRS and a short review of natural noise processing in CFRS.

2.1 Basics in collaborative filtering

A Recommender System (RS) has been defined as *"any system that produces individualized recommendations as output or has the effect of guiding the user*

in a personalized way to interesting or useful objects in a large space of possible options" [15]. Therefore, the main tasks of a RS are: (i) to gather information about the users' needs and interests, and (ii) to present the items that might satisfy such needs and interests. The recommendation problem can be formally defined as finding the most suitable item, or set of items, that maximizes the rating prediction for a target user:

$$Recommendation(I, u_j) = arg \max_{i_k \in I} [Prediction(u_j, i_k)] \quad (1)$$

being I=$\{i_1,\ldots,i_n\}$ the set of all the items and *Prediction*(u_j,i_k) is a function to predict how satisfied would be the user u_j with the item i_k, regarding the data available about u_j and i_k.

Different approaches have been proposed in the literature to recommend, such as content-based [16], knowledge-based [17], or demographic-based [18]. However, collaborative filtering (CFRS) is the most widespread approach in RSs [2, 11], because of its ability to provide effective recommendations only requiring minimal information [19]. This information is usually composed by explicit or implicit feedback from the users. This work is focused on CFRS with explicit feedback preferences, r_{u_j,i_k} (see Table 1), which are given by user's preference values over a subset of items.

Table 1. Users' preferences over items, the rating matrix.

	i_1	$\ldots$	i_k	$\ldots$	i_n
u_1	r_{u_1,i_1}	$\ldots$	r_{u_1,i_k}	$\ldots$	r_{u_1,i_n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_j	r_{u_j,i_1}	$\ldots$	r_{u_j,i_k}	$\ldots$	r_{u_j,i_n}
$\vdots$	$\vdots$	$\ddots$	$\vdots$	$\ddots$	$\vdots$
u_m	r_{u_m,i_1}	$\ldots$	r_{u_m,i_k}	$\ldots$	r_{u_m,i_n}

Among the approaches for CFRS, two pioneer and yet effective methods are the user-based and item-based collaborative filtering approaches [20]. Both methods rely on the nearest neighbours algorithm. Due to the fact that both represent key methods in the CFRS research, both of them will be used as the base for the evaluation of the natural noise approaches revised across this contribution.

2.2 Natural noise in recommender systems

Several authors have pointed out that the user preferences in RS could be inconsistent due to several reasons [12]. These inconsistencies have been classified in two main groups: (i) *malicious noise*, when the inconsistencies are intentionally inserted to bias the recommendation [21], or (ii) *natural noise*, when the inconsistencies appear without malicious intentions [22]. While the malicious noise

received much attention since the beginning of the use of RS [23, 24], natural noise has attracted less attention from researchers.

The concept of natural noise was introduced by O'Mahony et al. [12], in which the authors focus on discovering noisy ratings, both malicious and natural noise. Amatriain et al. [22] performed a user study to obtain a better understanding about how natural noise tends to appear. Afterwards, in [25] they propose strategies to correct the inconsistent preferences. More recently, Pham and Jung [9] proposed the use of item attributes, such as genre, actors, or director, in the case of movies, to detect and correct natural noisy ratings. They focus on finding a preference model based on this information for each user, and then identify as anomalous those positive ratings that do not match the model. The inconsistent ratings are then corrected using the information associated to other users identified as experts.

While the previous methods focus on the detection of noisy ratings, Li et al. [8] propose the discovery of *noisy but non-malicious users* by detecting user's *self-contradictions*, following the principle that highly-correlated items should be similarly rated. This study focuses on noise detection at user level, which could not be detailed enough in some scenarios.

All previous methods manage natural noise in CFRS in different ways, for instance O'Mahony et al. [12] and Li et al. [8] remove noisy information from the dataset. Others use additional information beyond the rating matrix, such as Pham and Jung [9] and Amatriain et al. [25]. Table 2 characterizes these proposals for dealing with NN according to the management of noise and the necessity of additional information beyond current ratings. Table 2 shows a shadowed cell that represents the lack of research centered on noisy ratings correction without additional information that should be filled and it is the aim of this contribution.

Table 2. Related works in Natural Noise

		Information required beyond ratings	
		Additional information	No additional information
Management of noise	Remove noise		O'Mahony et al. [12] Li et al. [8]
	Correct noise	Amatriain et al. [25] Pham et al. [9]	

3 Managing Natural Noise relying just on ratings: A basic approach

Yera, Caballero and Martínez [10] proposed a method for correcting noisy preferences that only relies on the ratings and it processes them at a rating level in

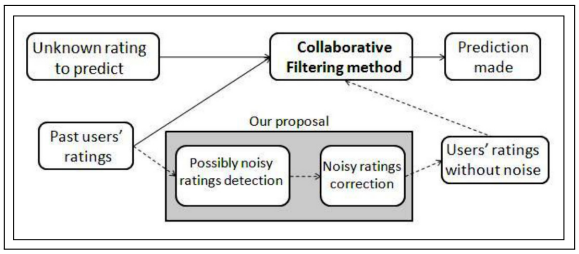

Fig. 1. General scheme of the natural noise approach

order to fill the gray cell in Table 2. This approach takes into account just rating values, in which it was not necessary to use any additional information about users either items.

In such an approach, apart from preferences variation over time, it was considered that erroneous ratings can appear in recommender systems dataset due to several reasons:

- Users can unintentionally express information that does not correspond with their preferences and profiles. This is the classical scenario for natural noise.
- They could also insert anomalous-but-correct information intentionally added, that is not actually aligned with their global profiles and could be considered as noisy.

In all cases, erroneous ratings or ratings that do not represent real user's preferences can cause accuracy decay in collaborative filtering. Incorrect ratings could alter user's profile, and the biased user could fall in a different neighborhood compared with an unbiased one. This could affect current user's predictions, and also predictions for users in the neighborhood.

In order to manage natural noisy ratings, a framework is proposed, which includes two steps (see Figure 1):

1. *Noisy ratings detection*: it classifies user and item profiles into four different classes considering their ratings. Each rating can also be classified into three different classes. A rating is marked as possible noise if there are contradictions between the rating class and the classes associated to its user and item.
2. *Noise correction*: it uses a basic collaborative filtering method to predict a new rating for each possible noisy one. If the difference between both ratings is greater than a predefined threshold, then the old rating is definitively considered as noisy and its value is replaced with the new one.

A further detailed description of previous phases is introduced below:

3.1 Noisy rating detection

This approach follows the principle that users and items have their *own tendency* giving or receiving ratings respectively. Once the tendencies have been identified,

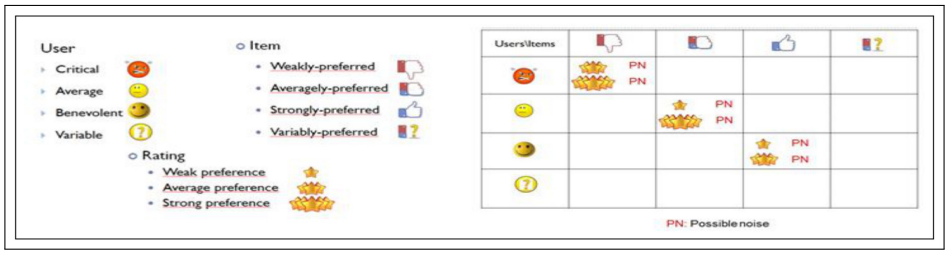

Fig. 2. User, item, rating classification and Possible noisy ratings

the ratings that contradict such tendencies can be classified as possibly noisy. Therefore, to detect natural noisy ratings, a classification for ratings, users, and items is initially performed.

The approach assumes that each user has his/her own tendency when rating items (see Fig. 2): i) A group of users tends to positively evaluate all items, ii) another group provides average values, iii) a third one usually gives low ratings, and also, iv) there is a fourth group of users whose behavior varies among any of the former categories, and do not fall into a specific one.

This classification is also extended for items: i) There is a group that is highly preferred by all users, ii) a group whose items are averagely preferred, iii) a group of items that are not preferred by the majority of users, and like in users, and iv) a group that contains items with divided opinions about their preferences degree.

Eventually, each rating $r(u, i)$ (for a user u and an item i) is classified into three different classes according its value:

1. Weak preference: A rating $r(u, i)$ is a *weak preference* if r(u,i) $< \kappa$.
2. Average preference: A rating $r(u, i)$ is an *average preference* if $\kappa \leq$ r(u,i) $< \nu$.
3. Strong preference: A rating $r(u, i)$ is a *strong preference* if r(u,i) $\geq \nu$.

This classification depends on a weak-average threshold κ and an average-strong threshold ν that are prior defined, satisfying $\kappa < \nu$ (See *Remark 1*).

Considering U and I as whole sets of users and items, preferences for each user u can be grouped in sets W_u, A_u and S_u, :

1. $W_u = \{r(u, i),\ \forall i \in I\ where\ r(u, i)$ is a weak preference$\}$
2. $A_u = \{r(u, i),\ \forall i \in I\ where\ r(u, i)$ is an average preference$\}$
3. $S_u = \{r(u, i),\ \forall i \in I\ where\ r(u, i)$ is a strong preference$\}$

And for each item i in sets W_i, A_i and S_i:

1. $W_i = \{r(u, i),\ \forall u \in U\ where\ r(u, i)$ is a weak preference$\}$
2. $A_i = \{r(u, i),\ \forall u \in U\ where\ r(u, i)$ is an average preference$\}$
3. $S_i = \{r(u, i),\ \forall u \in U\ where\ r(u, i)$ is a strong preference$\}$

Considering rating classes and user and item sets, user profiles can be formally classified into four different categories: *benevolent, average, critical, and variable.*

Table 3. User and item classes proposed.

User classes	
Critical user	Verifies card(W_u) $\geq$ card(A_u) + card(S_u)
Average user	Verifies card(A_u) $\geq$ card(W_u) + card(S_u)
Benevolent user	Verifies card(S_u) $\geq$ card(W_u) + card(A_u)
Variable user	Does not satisfy the others user conditions
Item classes	
Weakly-preferred item	Verifies card(W_i) $\geq$ card(A_i) + card(S_i)
Averagely-preferred item	Verifies card(A_i) $\geq$ card(W_i) + card(S_i)
Strongly-preferred item	Verifies card(S_i) $\geq$ card(W_i) + card(A_i)
Variably-preferred item	Does not satisfy the others item conditions

On the other hand, items can be classified in four categories: strongly-preferred, averagely-preferred, weakly-preferred and variably-preferred (see Table 3).

Table 3 summarizes user and item classification, being, card(A), the cardinality of the set A. Users and items are classified in one of the first three categories in each case if the amount of ratings belonging to the corresponding class rating (following rating classes) exceeds the amount of the other two. Specifically, for each user and item, it is counted the amount of ratings belonging to each preference class.

Once each rating, user and item has been classified, the detection process looks for contradictions among them. In general, it is assumed that if a rating belongs to similar classes regarding its user and item, then it must belong to the similar rating class. Otherwise, the rating could be erroneous. Specifically, the following procedure is presented to determine if a rating could be noisy (see Figure 2):

1. Classify the rating, and classify its corresponding user and item.
2. Mark the rating as possible noise if:
   (a) User class is critical, item class is weakly-preferred, and rating class is an average or a strong preference.
   (b) User class is average, item class is averagely-preferred, and rating class is a weak or a strong preference.
   (c) User class is benevolent, item class is strongly-preferred, and rating class is an average or a weak preference.

Remark 1. The proposal depends on three parameters: κ, ν and δ. These parameters are highly domain-dependant in [10] can be found a further detailed analysis about them.

3.2 Noise correction process

Some authors propose to discard noisy ratings, while others consider that these ratings must be corrected. Yera, Caballero and Martínez [10] proposal adopted the latter view. Before correction, another reason that classifies ratings as noise is

verified. For each value marked as possible noise (PN), another value is predicted using traditional memory-based user-user collaborative filtering with Pearson's similarity and k=60, The predicted value is compared with the current value. If the difference between them exceeds a predefined threshold δ then the new value is set as rating. Otherwise, the initial one is kept.

Summarizing, the whole correction process is:

1. Classify each user and item according definitions.
2. For each rating
   (a) Classify them according definition.
   (b) Following the procedure presented in the previous section, mark it if represents a possible noise.
3. For each rating marked as possible noise
   (a) Predict its value using traditional collaborative filtering
   (b) Calculate the difference between predicted and original value.
      i. If the difference exceeds a threshold, then substitute the original with the predicted value.
      ii. Otherwise, remain the value as the original one.

3.3 Case study

Following the experimental protocol suggested by Gunawardana and Shani in [26] a case study is carried out to evaluate the effects of the previous proposal over Movielens, which is a well-known dataset containing 100,000 movie ratings on 943 users and 1,682 items where each rating is discrete and is given in the range $[1, 5]$.

For the mentioned dataset, it is evaluated the MAE of the CFRS methods *user-user* and *item-item* with and without rating correction. To prepare data for experiments, 900 users were randomly selected conforming the training and test set as aforementioned. This task is then performed five times, selecting each time a different set with the same amount of users. The approach parameters were set as κ=2, ν=4 and δ=1. The Table 4 shows the results obtained.

	User-User CF	UUCF + NN management	Item-Item CF	IICF + NN management
k=10	0.7143	**0.6961**	0.7179	**0.6959**
k=20	0.7054	**0.6887**	0.7111	**0.6909**
k=30	0.7039	**0.6874**	0.7100	0.**6901**
k=40	0.7037	**0.6873**	0.7099	**0.6900**
k=50	0.7037	**0.6873**	0.7098	**0.6900**
k=60	0.7039	**0.6875**	0.7098	**0.6901**

Table 4. MAE values with and without rating correction.

The case study illustrates that the proposed approach obtained positive results for the datasets used in experiments. It proves that the correction approach

decreases MAE value disregarding the algorithm used. It is remarkable that the correction process improves MAE value in a similar degree comparing with the other two traditional recommendation methods (Pearson's user-user and item-item). For a further detailed analysis see [10].

4 A fuzzy approach to detect noisy ratings

In previous section it has been shown that the use of *NN* management approaches in RSs generally improves the accuracy of recommendations. However, existing approaches do not properly manage the inherent uncertainty associated to ratings because they represent and manage the ratings and its noise in a precise way. This precise management lacks of flexibility and robustness, hence a way to properly mitigate this drawback is to use fuzzy tools based on fuzzy logic and fuzzy linguistic approach for modelling the uncertainty and vagueness associated to user's preferences.

Therefore, Yera, Castro and Martínez [14] focused on overcoming the lack of flexibility and robustness by proposing a novel fuzzy *NN* method to deal with the inherent uncertainty of *NN*, improving in such a way the noise management and consequently the recommendation accuracy. This approach extends the approach revised in Section 3 because it focuses on the *NN* management just using ratings. This fuzzy approach for managing *NN* extends the proposal in Figure 1 and it is composed by the following phases (see Figure 3): (i) fuzzy profiling, (ii) noise detection, and (iii) noise correction.

These phases are further detailed in the coming sections.

4.1 Fuzzy profiling

The fuzzy profiling consists of transforming ratings values into a fuzzy linguistic representation. This representation allows to obtain users', items' and ratings' fuzzy profiles. These initial profiles are then transformed by using Computing with Words (CW) [**?**,**?**] into modified profiles to boost their tendencies in a flexible way. Initially, the membership functions that characterize the ratings over its universe of discourse are defined. These functions are respectively associated to

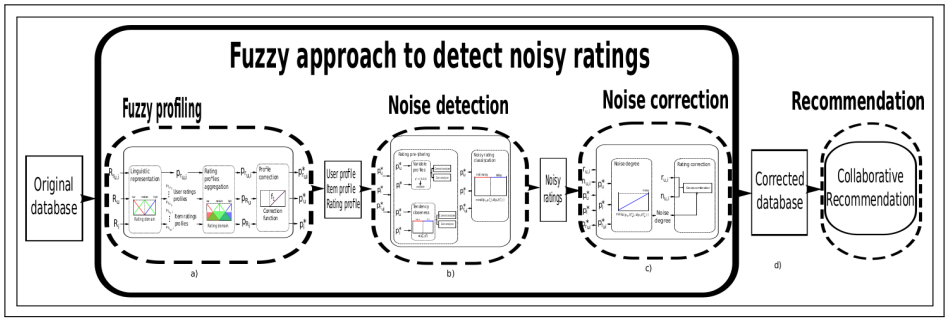

Fig. 3. General scheme of the fuzzy approach to eliminate noise in ratings database.

the fuzzy linguistic labels [**?**], $S = \{low, medium, high\}$, whose fuzzy semantics are shown in Fig. 4.

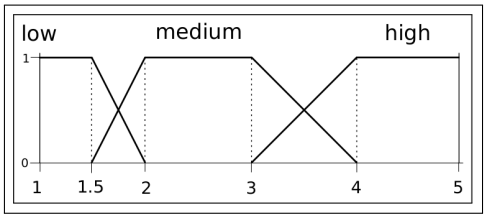

Fig. 4. Fuzzy definition of the ratings domain for the one to five stars domain.

Once the ratings are represented using fuzzy linguistic terms, they are used to build the users', items', and ratings' profiles. Specifically, a user's profile p_{R_u} is built using the fuzzy representation of the user's ratings denoted by R_u (see Eqs. 2 and 3).

$$\mu_{low}(R_u) = \frac{\sum_{r_{ui} \in R_u} \mu_{low}(r_{ui})}{|R_u|} \qquad \mu_{medium}(R_u) = \frac{\sum_{r_{ui} \in R_u} \mu_{medium}(r_{ui})}{|R_u|}$$
$$\mu_{high}(R_u) = \frac{\sum_{r_{ui} \in R_u} \mu_{high}(r_{ui})}{|R_u|} \tag{2}$$

$$p_{R_u} = (\mu_{low}(R_u), \mu_{medium}(R_u), \mu_{high}(R_u)) \tag{3}$$

The items' fuzzy profiles are built similarly to users' fuzzy profiles:

$$p_{R_i} = (\mu_{low}(R_i), \mu_{medium}(R_i), \mu_{high}(R_i)) \tag{4}$$

In the case of rating's fuzzy profile, a fuzzy profile $p_{R_{ui}}$ for rating r_{ui} is built using only the rating r_{ui} itself, i.e., $R_{ui} = \{r_{ui}\}$.

When the ratings do not present a clear tendency, the initial profile has similar membership values for all terms, $p_{R_u} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$, $p_{R_i} = (\frac{1}{3}, \frac{1}{3}, \frac{1}{3})$. In this situation it is difficult to identify the tendencies and perform a successful *NN* management. The proposal applies a CW process [**?**] that transforms the profiles by maintaining high membership values and discarding low ones. The function

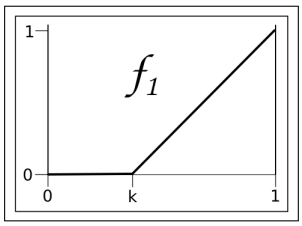

Fig. 5. The fuzzy transformation function to boost the fuzzy profile detected tendency.

f_1 (Fig. 5) is used with this purpose, depending on the parameter k whose value must be greater than $\frac{1}{3}$ to allow certain flexibility, for instance $k = 0.35$.

The transformed profiles are defined as:

$$p^*_{R_x} = (p^*_{R_x,low}, p^*_{R_x,medium}, p^*_{R_x,high}) = (f_1(\mu_{low}(R_x)), f_1(\mu_{medium}(R_x)), f_1(\mu_{high}(R_x)))$$

4.2 Noise detection phase

It classifies each rating of the dataset as noisy or not noisy. To do so, it checks if a given rating r_{ui} is noisy by analysing the rating tendency of its user u and item i, therefore the transformed fuzzy profiles are used to identify whether the rating matches the user's and item's tendencies or it lies out of the detected tendencies. The noise detection phase consists of two main steps:

i) Rating pre-filtering: Each rating is analysed to determine whether it is eligible for the noise classification step. The corresponding user and item fuzzy profiles are compared to determine their closeness. Ratings whose corresponding user or item profiles have zero membership for all terms are discarded. For the remaining ones, their corresponding user and item profiles are compared by using the Manhattan distance, d, and a threshold d_1:

$$d(p^*_{R_u}; p^*_{R_i}) < d_1$$

If a given rating r_{ui} satisfies this inequality, then r_{ui} is eligible for noise classification. The best value for d_1 has been determined empirically [14], concluding that for $d_1 = 1$ the proposal obtains an optimal performance.

ii) Noisy rating classification: For the chosen ratings in the pre-filtering step, its user, item and rating fuzzy profiles are compared to determine if the rating follows the user and item tendency. Thereby, for a rating r_{ui}, the distances among the profiles are computed and a threshold d_2 is used:

$$min(d(p^*_{R_u}; p^*_{R_{ui}}), d(p^*_{R_i}; p^*_{R_{ui}})) \geq d_2$$

If the rating verifies this inequality then r_{ui} is noisy. Several experiments [14] were performed to determine the most suitable value for the threshold, d_2, determining that $d_2 = 1$ is the best value.

4.3 Noise correction phase

Once the noisy ratings have been detected, this approach not only fixes noisy ratings but also it provides a *noise degree* of each rating that makes the rating correction much *more flexible* and *adaptable*. To define the noise degree of a noisy rating, the normalised Manhattan dissimilarity between profiles is used:

$$dissimilarity(p^*_{R_x}, p^*_{R_y}) = (d(p^*_{R_x}, p^*_{R_y}) - d_{min})/(d_{max} - d_{min}),$$

where $p^*_{R_x}$ and $p^*_{R_y}$ are the two profiles being compared, and d_{min} and d_{max} the minimum and maximum distance value between two profiles.

The noise degree of a noisy rating is defined as:

$$NoiseDegree_{r_{ui}} = T(\ dissimilarity(p^*_{R_u}, p^*_{R_{ui}}),\ dissimilarity(p^*_{R_i}, p^*_{R_{ui}})\ )),$$

being T a t-norm. In our context, the minimum is used as t-norm. After this, a new rating value n_{ui} for the same user and item is predicted, using a CF method (in this case the UserKNNPearson prediction approach [11]) using only the not noisy ratings. Afterwards, noisy ratings are corrected using the $NoiseDegree_{r_{ui}}$, original value and prediction:

$$r^*_{ui} = r_{ui} * (1 - NoiseDegree_{r_{ui}}) + n_{ui} * NoiseDegree_{r_{ui}}$$

4.4 Case Study

The proposal is evaluated in MovieLens datset (ml-100k), which is prepared according to the procedure proposed by Gunawardana and Shani [26] to build training and test sets. The method performance is then evaluated through the following steps:

1. Apply a noise correction method over the training set, obtaining the modified training set.
2. Recommend with a given recommendation method using the modified training set.
3. Evaluate the recommendation results using the MAE.

This protocol is used to compare the current proposal (NN-Fuzzy) with related previous works. Specifically, the approaches presented by O'Mahony et al.[12] (DiffBased in Table 5), Li et al. [8] (NNMU) and Yera et al. [10] (NN-Crisp), and a baseline that does not use any noise correction approach (Base) are compared.

Table 5. MAE results on the MovieLens dataset.

Predictor	Base	DiffBased	NNMU	NN-Crisp	NN-Fuzzy
UKNN	0.7647	0.7662	0.7644	0.7632	**0.7608**
IKNN	0.7705	0.7749	0.7699	0.7674	**0.7656**

The results shown in Table 5 demonstrate that NN-Fuzzy obtains the best results for all cases. This evidence proves that it overcomes the performance of all related works.

5 Natural Noise in Group Recommendation scenarios

Group recommender systems (GRSs) filter relevant items to groups of users regarding their preferences. These preferences can be explicitly given by the members, hence natural noise can also affect these scenarios. However, the natural noise problem has not been addressed on GRSs yet.

It seems worthy to research the NN management (NNM) in this scenario but taking into account that the complexity is much higher than in individual scenarios and at least in the group recommendation scenario two levels of data should be considered: (i) *local level*: preferences belonging to the group members, and (ii) *global level*: preferences of all the users in the entire dataset. The level considered most suitable to perform the NNM should then be studied. Four alternatives for NNM in GRSs should then be analyzed in both levels of data:

- First, two approaches that focus the NNM on the local level before the recommendations. The approaches are local NNM based on local information, NNM-LL, and local NNM based on global information, NNM-LG.
- Second, an approach that focuses the NNM on the global level before the recommendation, disregarding the group to which each user might belong. The approach is noted as global NNM approach (NNM-GG).
- Eventually, a cascade hybridization of the previous approaches is presented (NNM-H). It performs a global NNM approach, and then a local NNM that corrects the group ratings by using the information already corrected.

Acknowledgments

This research work was partially supported by the Research Project TIN2015-66524-P, and the Spanish Ministry of Education, Culture and Sport FPU fellowship (FPU13/01151).

References

1. G. Adomavicius, A. T. Tuzhilin, Toward the next generation of recommender systems: A survey of the state-of-the-art and possible extensions, IEEE Transactions on Knowledge and Data Engineering 17 (6) (2005) 734–749.
2. J. A. Konstan, J. Riedl, Recommender systems: from algorithms to user experience, User Modeling and User-Adapted Interaction 22 (1-2) (2012) 101–123.
3. B. Xiao, I. Benbasat, E-commerce product recommendation agents: use, characteristics, and impact, Management Information Systems Quarterly 31 (1) (2007) 137–209.
4. R. Yera Toledo, Y. Caballero Mota, An e-learning collaborative filtering approach to suggest problems to solve in programming online judges, International Journal of Distance Education Technologies 12 (2) (2014) 51–65.
5. J. Noguera, M. Barranco, R. Segura, L. Martínez, A mobile 3d-gis hybrid recommender system for tourism, Information Sciences 215 (2012) 37–52.
6. M. Pazzani, D. Billsus, Content-Based Recommendation Systems, The Adaptive Web 4321 (2007) 325–341.
7. J. Castro, R. M. Rodríguez, M. J. Barranco, Weighting of features in content-based filtering with entropy and dependence measures, International Journal of Computational Intelligence Systems 7 (1) (2014) 80–89.
8. B. Li, L. Chen, X. Zhu, C. Zhang, Noisy but non-malicious user detection in social recommender systems, World Wide Web 16 (5-6) (2013) 677–699.

9. H. X. Pham, J. J. Jung, Preference-based user rating correction process for interactive recommendation systems, Multimedia tools and applications 65 (1) (2013) 119–132.
10. R. Yera Toledo, Y. Caballero Mota, L. Martínez, Correcting noisy ratings in collaborative recommender systems, Knowledge-Based Systems 76 (2015) 96 – 108.
11. M. D. Ekstrand, J. T. Riedl, J. A. Konstan, Collaborative filtering recommender systems, Foundations and Trends in Human-Computer Interaction 4 (2) (2011) 81–173.
12. M. P. O'Mahony, N. J. Hurley, G. Silvestre, Detecting noise in recommender system databases, in: Proceedings of the 11th international conference on Intelligent user interfaces, ACM, 2006, pp. 109–115.
13. X. Amatriain, A. Jaimes, N. Oliver, J. M. Pujol, Data mining methods for recommender systems, in: Recommender Systems Handbook, Springer US, 2011, Ch. 2, pp. 39–71.
14. R. Yera, J. Castro, L. Martínez, A fuzzy model for managing natural noise in recommender systems, Applied Soft Computing 40 (2016) 187 – 198. doi:http://dx.doi.org/10.1016/j.asoc.2015.10.060.
URL http://www.sciencedirect.com/science/article/pii/S1568494615007048
15. R. Burke, Hybrid recommender systems: Survey and experiments, User Modelling and User-Adapted Interaction 12 (4) (2002) 331–370.
16. L. Martínez, L. G. Pérez, M. Barranco, A multigranular linguistic content-based recommendation model: Research articles, International Journal of Intelligent Systems 22 (5) (2007) 419–434.
17. L. Martínez, M. J. Barranco, L. G. Pérez, M. Espinilla, A knowledge based recommender system with multigranular linguistic information, International Journal of Computational Inteligence Systems 1 (3) (2008) 225–236.
18. M. Vozalis, K. Margaritis, Using svd and demographic data for the enhancement of generalized collaborative filtering, Information Sciences 177 (15) (2007) 3017 – 3037. doi:http://dx.doi.org/10.1016/j.ins.2007.02.036.
URL http://www.sciencedirect.com/science/article/pii/S0020025507001223
19. I. Pilászy, D. Tikk, Recommending new movies: even a few ratings are more valuable than metadata, in: Proceedings of the third ACM conference on Recommender systems, ACM, 2009, pp. 93–100.
20. C. Desrosiers, G. Karypis, A comprehensive survey of neighborhood-based recommendation methods, in: Recommender Systems Handbook, Springer US, 2011, Ch. 4, pp. 107–144.
21. C. Dellarocas, Immunizing online reputation reporting systems against unfair ratings and discriminatory behavior, in: Proceedings of the 2nd ACM conference on Electronic commerce, ACM, 2000, pp. 150–157.
22. X. Amatriain, J. M. Pujol, N. Oliver, I like it... i like it not: Evaluating user ratings noise in recommender systems, in: User modeling, adaptation, and personalization, Springer, 2009, pp. 247–258.
23. I. Gunes, C. Kaleli, A. Bilge, H. Polat, Shilling attacks against recommender systems: a comprehensive survey, Artificial Intelligence Review 42 (4) (2014) 767–799.
24. B. Mobasher, R. Burke, R. Bhaumik, C. Williams, Toward trustworthy recommender systems: An analysis of attack models and algorithm robustness, ACM Transactions on Internet Technology 7 (4).

25. X. Amatriain, N. Lathia, J. M. Pujol, H. Kwak, N. Oliver, The wisdom of the few: a collaborative filtering approach based on expert opinions from the web, in: Proceedings of the 32nd International ACM SIGIR Conference, ACM, New York, NY, USA, 2009, pp. 532–539.
26. A. Gunawardana, G. Shani, A Survey of Accuracy Evaluation Metrics of Recommendation Tasks, Journal of Machine Learning Research 10 (2009) 2935–2962.