A fuzzy associative classification approach for recommender systems

Joel Pinho Lucas, Anne Laurent, Maria N. Moreno, Maguelonne Teisseire

DOI : 10.1142/S0218488512500274

International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems

A FUZZY ASSOCIATIVE CLASSIFICATION APPROACH FOR RECOMMENDER SYSTEMS

JOEL PINHO LUCAS

Departamento de Informática y Automática, Universidad de Salamanca, Plaza de la Merced s/n, Salamanca, 37008, Spain *
joelpl@usal.es

ANNE LAURENT

LIRMM, Université Montpellier 2-CNRS UMR 5506, 161 rue Ada Montpellier, 34095, France
laurent@lirmm.fr

MARÍA N. MORENO

Departamento de Informática y Automática, Universidad de Salamanca, Plaza de la Merced s/n, Salamanca, 37008, Spain
mmg@usal.es

MAGUELONNE TEISSEIRE

Cemagref, UMR TETIS-Maison de la télédétection, 500 rue J.F.Breton Montpellier, 34093 Cedex 5, France †
maguelonne.teisseire@teledetection.fr

Received (received date)
Revised (revised date)

Despite the existence of different methods, including data mining techniques, available to be used in recommender systems, such systems still contain numerous limitations. They are in a constant need for personalization in order to make effective suggestions and to provide valuable information of items available. A way to reach such personalization is by means of an alternative data mining technique called classification based on association, which uses association rules in a prediction perspective. In this work we propose a hybrid methodology for recommender systems, which uses collaborative filtering and content-based approaches in a joint method taking advantage from the strengths of both approaches. Moreover, we also employ fuzzy logic to enhance recommendations' quality and effectiveness. In order to analyze the behavior of the techniques used in our methodology, we accomplished a case study using real data gathered from two recommender systems. Results revealed that such techniques can be applied effectively in recommender systems, minimizing the effects of typical drawbacks they present.

Keywords: Recommender Systems; Associative Classification; Fuzzy Association Rules;

*LIRMM, Université Montpellier 2-CNRS UMR 5506, 161 rue Ada, Montpellier, 34095, France
†LIRMM, Université Montpellier 2-CNRS UMR 5506, 161 rue Ada Montpellier, 34095, France

1. Introduction

Nowadays, the world's information base increases dramatically. Additionally, it is technologically feasible to store huge volumes of data with, more and more, lower costs. However, it results in an "information explosion" where there is a lot of useless data in which it is very difficult to find valuable information.

In e-commerce applications such "information explosion" is reflected by loads of products available for sale. In this way, users would probably have difficulty in choosing the products they prefer and, consequently, have difficulty to purchase them. There is a need for new technologies that make able to interpret the entire information available in order to find what is more relevant.[1] Facing the enormous quantity of products, or items, available in current e-commerce systems, the constraint mentioned before become even more critical. Due to such facts and to a more and more competitive industry, these systems need to personalize the presentation of their products to the consumers. A way to reach such personalization is by means of the so called "recommender systems", which are being used by an ever-increasing number of e-commerce sites in order to help consumers to find products to purchase.[2]

Within this context, Recommender Systems appeared aiming at offering products for users of e-commerce applications in a personalized mode. This way, web applications are able to supply their users more facilities in finding those products they prefer. For this reason, such applications may also increase their sales because, in this context, they are able to interact with their users in order to aid them in choosing and finding products and services of their interest. Accordingly, recommender systems aim at adapting themselves for each user. Under this broader definition, recommender systems serve to support a customization of the consumer experience in the presentation of the sold products, thus, recommender systems enable, in a sense, the creation of a new store personally designed for each consumer.[2] Nevertheless, nowadays they may also be employed in other domains further than the e-commerce, such as virtual libraries, news websites, scientific portals, e-learning systems, etc.

Methods employed in recommender systems have their foundations in different domain areas. According to Cheung et al.[3] and Lee et al.[4] the methods implemented in recommender systems can be divided into two main classes: collaborative filtering and content-based methods. Both types of methods make use of information related to evaluations, or ratings, provided by users.

Taking into account that data mining techniques are applied for identifying patterns within data sets, these techniques can be successfully applied for recommender systems,[3] however they need to be extended to deal with common issues of such systems. The induction of association rules is a data mining technique widely applied in decision making processes, which was first introduced by Agrawal et al.[5] in

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

the context of market basket analysis.

In spite of being a non-supervised learning method, association rule induction can also be applied to perform classification tasks. In this work we develop a methodology to be applied in recommender systems, which uses association rules in a prediction perspective (usually referred as associative classifiers). Moreover, we also employ fuzzy logic to enhance recommendations' quality.

In order to analyze the behavior of such classifiers on recommender systems' data, we accomplished a case study using real data gathered from two recommender systems. The key novelty of this work is the use of association rules for classification tasks in recommender systems. Moreover, the use of fuzzy logic can also supply some advances for these systems. Within such methodology, we developed a classification based on association algorithm that applies fuzzy logic, which was named CBA-Fuzzy.

In the next section we describe some foundations relative to our methodology and to recommender systems, as well as some related works. Section 3 depicts the recommendation methodology we propose, as well as the developed algorithm. Subsequently, in section 4, we describe the case study accomplished to evaluate this proposal. After all, we highlight some conclusions obtained through this work.

2. Background and Related Work

In this section we describe general features of association rules for classification and recommender systems. Here we firstly describe some concepts related to classification based on association, as well as on fuzzy association. General aspects and works related to recommendation methods and the main drawbacks they present are also highlighted in this section.

2.1. *Classification Based on Association*

As stated before, association rule induction algorithms can be employed to build recommendation models such as the one in. [4] Association rules were first introduced by Agrawal et al.[5] aiming at discovering consumption patterns in retail databases. Thus, the task of discovering association rules in retail data was termed as "market basket analysis". Agrawal et al.[5] demonstrated that an association rule expresses, in a dataset, the probability that the occurrence of a set of items implies the occurrence of another set of items. These authors defined the following formalization: let $I = \{i_1, i_2, i_3, \ldots, i_{n-1}, i_n\}$ be a set of items (also called attributes), where each element "i" that belongs to "I" can present binary values (true or false) representing their membership to such set. Moreover, there is a transaction set $T = \{t_1, t_2, t_3, \ldots, t_{n-1}, t_n\}$ among elements of I, where each element "t" of "T" expresses a set of items in "I", such as $t \subseteq I$. The representation of an association rule may be declared as $A \rightarrow B$, where A and B are item sets. Such representation states that, in a transaction, the occurrence of all items from "A" (antecedent side of the rule) results in the occurrence of items belonging to "B" (consequent side of

the rule), such as $A \subseteq I$ and $B \subseteq I$. An association rule describes an association relation between item sets that occur together on transactions of a data set. Thus, association is not considered as a prediction task, because it aims at describing data.

On the other hand, a classification method is seen as a prediction task, because it aims at predicting the value of an attribute (label) in a data set. The joining of concepts from classification and association [6] is an alternative approach for performing classification tasks, where association rules are employed as a classification method. Seeing that association models are commonly more effective than classification models, a crucial matter that encourages the use of association rules in classification is the high computational cost that current classification methods present. Several works [7] [8] [9] [11] verified that classification based on association methods presents higher accuracy than traditional classification methods. Differing from association rules, the decision trees, for example, do not consider simultaneous correspondences occurring on different attribute values. Moreover, for human beings, an output provided by association rule algorithms can be much easier comprehended than an output provided by other usual classification techniques, such as artificial neural networks.[12]

According to Thabtah et al.[9], a few accurate and effective classifiers based on associative classification have been presented recently, such as CBA (Classification Based in Association)[6], CPAR (Classification based on Predictive Association Rules)[11], MCAR (Multi-class Classification based on Association Rule)[9], CMAR (Classification based on Multiple class-Association Rules)[10] and GARC (Gain based Association Rule Classification) [13]. Taking into account that for classification rule mining there is one and only one predetermined target, while for association rule mining the target of discovery is not pre-determined,[6] it is necessary to constrain the rules' consequent terms to encompass only one attribute. This way, the consequent term of an association rule will represent the target, or class, attribute. Therefore, such rule can play a prediction role in a given system: in order to classify an item, the rule's properties are matched to every rule's antecedents and the attribute value of the consequent term (from one or more selected rules) will be the predicted class.

Generally, the classification model is presented as an ordered list of rules, based on a rule ordering scheme.[14] In the CBA algorithm, for example, the rules are ordered by means of the confidence measure and it uses only one rule for performing classification. However, in this case some scenarios in which there are multiple rules with similar confidence measures may occur and, at the same time, with very different support measures. Hence, a rule A with much higher confidence than a rule B could be the one chosen for classification even if B had a much higher support.[10] The MCAR algorithm solves such drawback by means of an approach that considers, in addition to the confidence, the rules' support. The CMAR algorithm has a fine approach for selecting association rules for classification, instead of using just one rule it makes use of all rules that match the case to be classified. If the con-

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

sequent term of all selected rules is the same, the predicted class will obviously be the value of the rules' consequent term. Though, in a different scenario, rules are divided in groups according to the consequent terms' values. The value chosen for classification is acquired through the group in which its elements hold the highest correlation value according to the weighted $\chi 2$ measure. Similarly to CMAR, the CPAR algorithm also divides rules into groups, though, instead of using all rules that match to the object to be predicted, it uses the "k" best rules that represent each class. Afterwards, the algorithm chooses a group, by means of the Laplace Accuracy measure, that will be the one used for classification.

The drawbacks presented by association rule induction algorithms are, in general, the same as classification based on association algorithms. A critical drawback of these algorithms arises when they generate poor rules. Such rules generally encompass narrow information and own few attributes (or terms). An object owing few attributes would probably be ineffectively classified. Another critical drawback is due to the large number of rules that algorithms commonly produce,[12] as a consequence, much of them do not supply relevant information or are contradictory. Such drawback is a critical issue related to associative classifiers, because the performance of the algorithm may be affected when retrieving, storing, pruning and sorting a large number of rules.[10] The CMAR algorithm tries to solve such drawback by implementing a FP-Tree data structure to store the association rules' frequent itemsets.

Next subsection presents a variation of classification based on association, in which fuzzy logic concepts are employed for classification.

2.2. *Classification Based on Fuzzy Association*

The use of concepts from Fuzzy Sets in association rule mining, as with data mining in general, has been accomplished by many works in the literature, such as [15] [16] [17] [18], which is commonly reported as Fuzzy Association Rule Mining. Basically, the two major advantages of employing fuzzy association rules are because they allow rules to use vague linguistic expressions and to restrain the unwanted effect that boundary cases might derivate.

The use of vague linguistic expressions, according to Dubois et. al.[19] makes fuzzy association rules very appealing from a knowledge representational point of view, because the idea of fuzzy sets is to act as an interface between a numerical scale and a symbolic scale which is usually composed of linguistic terms. Consequently, rules generated in the mining process might be more comprehensible and present a higher level of abstraction. This way, rule terms can present values like "$< height, tall >$" instead of "$< height, 1.83 >$" or "$< income, medium >$" instead of "$< income, 20,000 >$", for example.

In a real situation, there are often some cases that can be easily handled by a human being, but difficult for a machine, such examples includes face and voice recognition, handwriting verification, etc.[20] In this context, fuzzy association rules may

be employed to build classification models, for example, where human's knowledge could be taken into account effortlessly. Such approach is also valid for recommender systems, because they are introduced in a large variety of distinct domains.

Moreover, as stated before, the use of fuzzy association rules can also restrain the sharp boundary effect between intervals, because fuzzy set concepts provide a smooth transition between intervals' memberships, resulting in fewer boundary elements being excluded.[20] Ordinary association rule mining algorithms usually either ignore or over-emphasize elements near to the boundary of intervals in the mining process. Additionally, the use of sharp boundary intervals is not intuitive with respect to human perception.[21] Fuzzy sets deal with the representation of intervals which has boundaries that are not clearly defined. Such representation is often based on a membership function that defines an element "i" to belong to a certain set. Such function takes values in [0, 1]. Thus, membership in a fuzzy set is a notion intrinsically gradual instead of abrupt or crisp (as in conventional Boolean logic).[22] Therefore, the membership value is not defined by an absolute binary (true or false) value. Moving from set-based (interval-based) to fuzzy associations is formally accomplished by replacing sets (intervals) by fuzzy sets (fuzzy intervals).[19] This way, authors defined the following formalization: given a set of index terms $T = \{t_1, \ldots, t_u\}$ and a set of items $I = \{i_1, \ldots, i_v\}$, where each t_i is represented by a fuzzy set $\mu(t_i)$ of items, which is defined as follows, where $F(t_i, i_j)$ is the membership degree of t_i in i_j.

$$\mu(t_i) = \{F(t_i, i_i) | \forall i_i \in I\} \tag{1}$$

Kuok et. al.[15] were the first ones to employ fuzzy sets concepts in association, they trivially formalized a fuzzy association rule as "*if X is A then Y is B*", where $X = \{x_1, x_2, x_3, \ldots, x_p\}$ and $Y = \{y_1, y_2, y_3, \ldots, y_q\}$ are itemsets and $A = \{f_{x_1}, f_{x_2}, f_{x_3}, \ldots, f_{x_p}\}$ and $B = \{f_{y_1}, f_{y_2}, f_{y_3}, \ldots, f_{y_p}\}$ contain the fuzzy sets associated with the corresponding attributes in X and Y.

In the same way, classification based on association can also make use of "fuzzy association" concepts. According to Jin [20], a fuzzy classifier based on fuzzy if-then rules is more interpretable, because the expert or the user is able to verify the classification paradigm and such verification may be done by judging the plausibility, consistency or completeness of the fuzzy rule sets in the classifiers.

Generally speaking, classifiers based on fuzzy association extend the same concepts of general associative classifiers, however, in order to obtain "fuzzy rules", we first have to partition numerical attributes, usually by means of linguistic terms definition. Some works [21] [22] [23] [24] define general fuzzy classifiers based on rules, where, instead of assigning a class label to each record, a soft class label with a degree of membership in each class is attached to each record (pattern) and the membership value of the class label varies from 0 to 1. According to Au and Chan[24], the use of fuzzy technique allows the prediction of attribute values to be associated with a degree of membership, thus, it is possible to deal with cases that an object can

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

belong to more than one class. In a recommender system of movies, for example, a user may share similar opinions with more than one group of users.

Au and Chan[24] formalized the definition of fuzzy associative classifiers, which is an extension of both fuzzy association and classification based on association. In this case we have the same set of items $I = \{i_1, i_2, i_3, \ldots, i_p, i\}$, where the last element "i" represents the class attribute, which is a categorical attribute with values C_1, C_2, ..., C_q. Let $X = \{x_1, x_2, x_3, \ldots, x_p\}$ be a sample, where x_1, x_2, x_3, ..., x_p are the values taken by attributes i_1, i_2, i_3, ..., i_p. Afterwards, a discriminant function is used to classify each sample, where the class predicted will be the one presenting the maximum value in such function. However, differently from most approaches for fuzzy classification, our approach takes into account all values greater than 0 resulted by the discriminant function in order to classify a sample in more than one class. In subsection 3.2, we depict how we apply and calculate this function to classify a sample in the context of the approach proposed in this work.

2.3. *Recommender System Methods*

As stated before, recommender systems may employ content-based or collaborative filtering methods. The content-based methods were the first approach to be proposed for recommender systems. Such methods compare text documents to user profiles, where web objects are recommended to a user based on those he has been interested in the past.[4] Hence, recommender systems that use such type of methods do not take into account the information acquired from other users. As a result, users of recommender systems employing content-based methods will receive recommendations based on similar items to those he/she has expressed interest before. Thus, besides obtaining data related to documents features, these methods need to obtain data related to the behavior of each user in order to make a recommendation. Information related to users' behavior are explored to build their profiles. In this way, rule-based techniques or keyword-matching are commonly used,[25] nevertheless, techniques based on Machine Learning are used in order to have the users' profiles automatically learned.

Unlike humans, content based techniques have difficulty in distinguishing between high quality and low quality information that is on the same topic.[26] Hence, it might not be feasible to make recommendations when opinions acquired from other users are not taken into account. So, current recommender systems do not employ content-based recommendation methods solely, because according to Balabanović and Shoham[27], in certain domains it is still complex to extract relevant features from items. Such authors also affirms that, in Web pages, for example, it is possible to extract just some content features, because current information retrieving techniques do not take into account multimedia elements, for example. Such features can interfere in user's interest for a certain item.

On the other hand, in collaborative filtering methods, the recommendation process is based on products' opinions collected from other users.[26] These methods

appeared aiming at providing more consistent recommendations by means of information concerning the social context of each user. Thus, the recommendation process is based on ratings of other users who have similar characteristics (preferences concerning system's items).[26]

As the collaborative filtering approach was originally based on nearest neighbor algorithms, the recommended products will be the ones that have been liked by users with similar interests to the "new user", who is commonly referred to as "Active User". Breese et al.[29] classified collaborative filtering methods into two groups: memory-based methods, which are also referred as user-based methods, and model-based methods, which are also referred as item-based methods. The memory-based methods are known as the nearest neighbor method because it was the first technique proposed for these methods. Memory-based methods were the first approach of collaborative filtering, where the whole set of users' opinions is needed to be used, because the opinions of the active user are matched with the ones of all other users of the system in order to find the most similar users to the active user. However, facing the current context of recommender systems, where we basically have sparse backgrounds, these methods are not broadly employed in current recommendation approaches.

On the other hand, model-based methods build a predictive model by means of a training set which comprises opinions acquired from a small portion of the system's users. Such type of method has been developed more recently in order to avoid the sparsity problem, which usually arises when memory-based methods are employed, because e-commerce systems generally offer millions of products for sale, so that it is not feasible to obtain opinions about all of them.[12] In order to build a predictive model, model-based methods usually make use of a training set that takes into account just opinions acquired from a part of the system's users. Nevertheless, the number of ratings available is usually small even to build a prediction model and, hence, new techniques need to be developed in order to solve such shortcoming.

Not surprisingly, machine learning techniques are the most employed techniques in model-based methods. Several machine learning methods that are employed to solve data mining problems are also employed in recommender systems and they may be very diverse, including techniques of clustering, Bayesian learning, neural networks, etc. Furthermore, nowadays numerous systems also employ the agents' technology combined to those techniques.[30] The use of agents in these systems is basically due to their autonomy, learning capability and the possibility of working in cooperation.[31]

In spite of being generally the most efficient approach for recommender systems, model-based collaborative methods also present some shortcomings as well as memory-based and content-based based methods. As a result, current approaches for recommender systems do not employ merely one type of method. Usually, current approaches are likely to employ concepts from both categories of methods in order to take benefit from the strengths of each of them. Several works, like [27] [26] [32] [33] [28] combine collaborative filtering with content-based methods. Such methods

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

may be combined of different ways, more primitive approaches, like the one proposed by Pazzani [33] which tackles content-based and collaborative filtering methods separately and combine the results from each afterwards. More recent approaches, like the one of Condliff et. al.[28] use all the available information, acquired from both approaches, in a single predictive model. The recommendation methodology proposed in this work, like the one described before, uses collaborative filtering and content-based approaches in a joint methodology, as it will be described in section 3.

In the next subsection we will state some of the most critical and general drawbacks related to collaborative filtering and content-based methods employed in recommender systems.

2.4. *Recommender System Drawbacks*

Recommender systems can present two types of errors: false negative and false positive. The first one consists of products that were not recommended, though the consumer would have liked them. The second one consists of recommended products, though the consumer does not like them. According to Sarwar et al.[12], the false positives are more critical because they will lead to angry consumers. In order to avoid false positives, methods employed in recommender systems must avoid the occurrence of some typical drawbacks. Subsequently, it will be described the four most critical drawbacks that may occur in these systems.

The biggest challenge recommender systems probably have nowadays and the most critical drawback is related to data sparsity. Such drawback exists due to the huge amount of data normally available in current recommender systems. Fundamentally, sparsity occurs because the number of ratings needed to build a prediction model is greater than the number of the ratings obtained from users.

Moreover, most recommender system techniques require user explicit expression of personal preferences for items (which means they need user interaction). Nevertheless, methods have been built for obtaining ratings implicitly in order to add more ratings and reduce sparsity However, even with the use of up-to-date methods (including data mining methods), sparsity still remains a critical drawback for recommender systems due to the extensive number of items available. According to Sarwar et. al.[1], even active users may have purchased less than 1% of the items available in a system. This means that in a recommender system of movies owning 500,000 items, for example, an active user would be able to rate 5,000 movies. Nevertheless, we can not expect that all users of the system watch 5,000 movies and provide ratings to all of them. Moreover, rating schemes can only be applied to homogeneous domains [34] and the number of eligible ratings to be used by the system is even more restricted.

The sparsity is more problematic for memory-based collaborative methods, because it may not be feasible to obtain enough ratings from users of a system. Model based methods reduce the shortcomings derived from sparsity, nevertheless it still

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

needs to have a certain minimum number of ratings in order to build an estimation model of ratings (like agents based methods). In any case, owning ratings of just 1% of system items may be, according to the technique used, scarce to build a reliable model.

Another drawback originated from the large number of items available in recommender systems is related to scalability. Scalability in recommender systems includes both very large problem sizes and real-time latency (time taken to give feedback to the user) requirements.[2] An example of such requirement could be a scenario in which a recommender system is connected to a Web page needing to provide recommendations within a few tens of milliseconds while attending thousands of users at the same time. As a result, according to Konstan and Riedl[35], current recommender systems hold the challenge of attending the ruthless demand for low latency and high throughput (information amount passing through the system) in order to serve a massive number of users.

Such drawback may turn into the major concern of the system performance, because it may be unfeasible to deal with huge datasets using all available items. According to Sarwar et. al.[1], a Web-based recommender system using only the nearest neighbor method will probably suffer from serious scalability problems. Generally, scalability is not a drawback for model-based methods, because in such methods, differently from others, main data processing and induction of the predictive model are usually not performed at run time (recommendation time).

Despite the above mentioned drawbacks being able to be brightened up by means of model-based collaborative filtering methods, there are some drawbacks that these methods can not solve. The "early rater problem"[26 28] is an example of drawback that may occur in all type of collaborative filtering methods. This problem is related to the restraint of having few opinions to base the predictions on. This problem refers to the impossibility of providing recommendations about an item that was recently added to a system and, as a consequence, it would have few (or none) ratings of users. In fact, the early rater problem is directly related to sparsity, because a system encompassing a high number of items will probably present many items that have not received any rating.

Sarwar et. al.[36] argue that current recommendation systems depend on the altruism of a set of users who are willing to rate many items without receiving many recommendations. Economists have speculated that even if rating required no effort at all, many users would choose to delay considering items to wait for their neighbors to provide them with recommendations.[37] Therefore, it is necessary to establish means to encourage users to rate available items. Analogously, this drawback also occurs with a new user accessing the system, because as there is no available information on him, it will not be possible to perceive his behavior in order to provide recommendations. An extreme case of the early rater problem is when a recommender system first begins, because every user suffers from the early rater problem for every item.[26]

Conversely, there are drawbacks, such as the "grey sheep problem"[26], that occur

only in collaborative filtering methods. The grey sheep problem refers to the users who have opinions that do not consistently agree or disagree with any group of users. As a consequence, such users would not receive recommendations. However, this problem does not occur in content-based methods, because such methods do not consider opinions acquired from other users of the system in order to make recommendations. Furthermore, the first rater problem neither occurs in such methods since they can provide recommendations based solely on the properties of an item. According to Condliff et. al.[28], since a content-based system does not consider the social background of its users, the system is limited to recommend just items that are similar to those that a user has liked in the past. Therefore, there could be many false negatives, because these methods are not able to distinguish between high and low information quality within the same subject.

2.5. *Fuzzy Associative Classification in Recommender Systems*

In this subsection we describe some related work that influenced the conception of our methodology, which is basically founded in fuzzy logic and classification based on association concepts.

Association rules are already widely employed and consolidated in web mining. Despite not being very popular in recommender systems, association rules can form a very compact representation of preference data that may improve efficiency of storage as well as performance in these systems.[38] Moreover, association rules, as well as other data mining techniques, can be successfully applied for recommender systems as well. However they need to be extended in order to deal with common issues of such systems.[3]

Most current works use the Sarwar et. al. approach[12], where association rules discovery algorithms are applied to find association between rated or co-purchased items by users and then generates item recommendation based on the strength of the association between items. Thus, the use of association rules in recommender systems are basically a collaborative filtering approach, because rule mining is based on data gathered from user opinions. Sun et. al.[39] showed an improvement of accuracy when comparing the use of association rules to classical collaborative filtering methods (e.g. correlation matrix). Such authors used quantitative association rules as they include ratings on rules' attributes. In this case, an association rule in a recommender system of movies could be: "Titanic" triggers "Brave Heart", where it could be interpreted that "rating 4 for the Titanic movie implies in rating 5 for the Brave Heart movie". Fu et. al.[40] have developed a recommending system of web pages using the Apriori algorithm to mine association rules on user navigation histories. Basically, in the recommender systems context, association rules are employed aiming at identifying items frequently found in "association" with items in which the active user has expressed interest.

In general, recommender systems using association rules build their recommendation model starting from a database of all users and all items they have purchased

or rated in the past in order to find association rules using an association rule discovery algorithm and satisfying some required minimum support and confidence constraints. Subsequently, they find all rules whose conditions match the items, left-hand side of rules, previously purchased or rated by the active user. Then, such rules are sorted, usually based on the confidence measure, where items presented by rules with higher confidence are ranked first. Finally, the system provides the recommendation containing the first N items on the consequent term of rules on the top of the list generated before. Besides, items that were purchased or rated by the active user are eliminated from the recommendation.

Nevertheless, current recommender systems can also combine other techniques with association rule mining. Before mining association rules, Sarwar et. al[1] selects a reduced number of collaborative users who are similar to the active user. In our approach, we apply association rule mining in a classification context, where, instead of only considering the occurrence of items users, we consider attributes that describe users and items of the system. The approach proposed by Lin et. al.[41] is, in some points, similar to ours, because it also mines rules for one target item in the consequent term using a variant of the rule generation module of the CBA algorithm (CBA-RG) based on the classical version of the Apriori algorithm. However, it does not use attributes of users or items, because it is based solely on their occurrence. In this way, only the most frequent items are recommended and some possible items of interest, but not frequent, to the active user, are ignored. Nevertheless, they consider relationships between users as well, where association rules for items and users are mined separately and items are distinguished by means of a binary rating scheme (with "like" or "dislike" values). Users are associated according to their preferences (liking or disliking) over certain items on the system. However, just one type of association rule is used: if there are few ratings given by the active user, associations between items will be used, otherwise just associations between users will be considered. Rules are mined at runtime for each specific target user, where it is not required to specify a minimum support in advance. Rather, a target range is given for the number of rules, and the algorithm adjusts the minimum support for each user in order to obtain a ruleset whose size is in the desired range.[41] However, each procedure may be very onerous when dealing with a sparse or high dimensional dataset, because the rule mining is made in an iterative process to be able to define the support automatically. A maximum number of rules threshold needs to be defined previously, where the algorithm executes this loop, increasing the minimum support count, until the number of rules is lower than the defined threshold. In the approach proposed in this work, the set of rules used for classification is constructed off-line, which enable the system to give quick answers to the user.

The use of association rules in recommender systems can also employ concepts from fuzzy logic and allow the recommendation model to deal with numerical results. Berka and Plößnig[42] employed the so called fuzzy association rules in a recommender system designed for the tourism application. According to authors, the smooth tran-

sitions between sets provided by fuzzy logic make them ideal for Tourism applications, where, a user is able, for example, to prefer a restaurant which is within a certain physical distance, but without having a fixed maximum distance for such selection).

Despite being effectively used in many domain areas, fuzzy logic is not widely employed in recommendation systems. However, they can help to minimize, or even solve, typical drawbacks of such systems. According to Dubois et. al.[19], fuzzy logic provides high-value properties to recover items stored in a database and, as a consequence, to provide recommendations for users, because fuzzy sets have the ability to manage concepts such as similarity, preference and uncertainty in a unified way and they also have the ability to perform approximate reasoning. Thanks to such advantages, especially for uncertainty, fuzzy logic can help to minimize the sparsity problem, which is the main drawback current recommender systems suffer from.

According to the context and the type of method considered, fuzzy logic can be used both in content-based and collaborative filtering methods. Some works, such as the one of Durbois and. al.[19], propose a more general use of fuzzy logic, where it is employed as a content-based and a collaborative filtering method. Authors of this work focused their attention on the implementation of case-based decision support, which is the basis to contemplate situations where users do not have absolute preferences or if preferences are expressed relatively to the context in order to be stored.

Nevertheless, there are more specific works, such as the one by Yager[43], where fuzzy sets methods are used to describe information in a content-based recommender system. Cao et. al.[44] have implemented a recommender system based on fuzzy concepts in order to recommend to users products not usually consumed. Moreover, it can always have situations in which there may not be enough information about the customer's past purchases and the customer may have his specific requirements in each single purchase.[44] In this context, authors use fuzzy set's operations in order to define relationships between user requirements and products' features.

On the other hand, there are some works that employ fuzzy logic methods in order to develop recommendation approaches based on collaborative filtering. Nasraoui et. al.[45] define the notion of user session as being a compact temporary sequence of web accesses made by the user. Afterward, these sessions are categorized using fuzzy partitions and, subsequently, recommendations are made in accordance to the categorized sessions. Campos et. al.[46] established a comprehensive approach that combines fuzzy set's theory with Bayesian Networks in order to represent the ambiguity or vagueness in the description of opinions provided by users.

3. Proposed Methodology

In this section we describe the development of a methodology proposed for recommender systems which aims at enhancing recommendation quality and effectiveness. We expect that, with the use of this methodology, it will be possible to minimize

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

the effects of drawbacks described in subsection 2.4. Therefore, we describe a set of procedures and algorithms to be applied in recommender systems of different domain areas.

The main contribution and novelty of this methodology is the employment of classification based on association for recommender systems. Additionally, the use of fuzzy sets concepts provides more value and efficacy for the methodology and allows it to compose a hybrid method taking advantage from the strengths of both collaborative filtering and content-based approaches.

In order to apply the methodology we developed a fuzzy associative classifier, named CBA-Fuzzy, based on the standard CBA algorithm. In the next subsection, we depict the CBA-Fuzzy algorithm and subsequently we describe the recommendation framework of the proposed methodology.

3.1. *The CBA-Fuzzy Algorithm*

The algorithm developed in this work is the basis for the proposed methodology, since it is responsible for generating the class association rules that composes the classification model employed for making recommendations. Nevertheless, besides the existence of few accurate and effective associative classifiers, there are even fewer implementations available nowadays. In fact, LUCS-KDD (The Lucs-KDD Software Library) research group's repository, from the University of Liverpool, was the only software repository we found offering associative classifiers for free use. Such repository holds CBA, CMAR, TFPC and CPAR algorithms.

The algorithm we developed, the CBA-Fuzzy, is an extension of the CBA algorithm version obtained from LUCS-KDD, which is an implementation for the approach proposed by Liu et. al[6]. CBA-Fuzzy, as well as the LUCS-KDD CBA algorithm, was implemented in Java using the Java2 SDK (Software Development Kit), which should therefore make both algorithms highly portable.

The approach proposed by Liu et. al.[6] consists of two components: a rule generator (called CBA-RG) and a classifier builder (called CBA-CB). The rule generator has the same foundations of the Apriori algorithm, which is based on a downward closure property, proposed by Agrawal et. al[5] for association rule mining. Firstly, it obtains the so-called "frequent itemsets" satisfying a minimum support threshold. The support is a measure that assesses the frequency that items of a rule occur in a dataset or, in other words, the number of transactions in which the items of the rule occur at the same time in the dataset. The frequent itemsets are generated by making multiple passes over the data, where the first pass corresponds to an individual itemset (just one item) used as a seed to generate new possibly frequent itemsets (candidate itemsets). In every subsequent pass, it starts from the seed set of the itemsets found to be frequent in the previous pass (it holds a support value greater than a given threshold). In every subsequent pass the frequent itemsets are added by one more item. At this point, each itemset has the same configuration of a classification rule of the classifiers builder, which can be defined as follows, where

condset encompasses the antecedent terms and y is the class.

$$condset \rightarrow y \tag{2}$$

An example of a rule following this definition could be: $\{(att_1 = a)$ AND $(atr_2 = b)\} \rightarrow (class = C_1)$, where "a" and "b" are values of attributes "att_1" and "att_2", respectively.

After obtaining the frequent itemsets, the algorithm will generate class association rules satisfying a minimum confidence threshold. Confidence is a measure that expresses the correspondence between items composing a rule. It is expressed by the occurrence frequency (percentage) of the rule among all the transactions containing the antecedent part. This measure can be obtained by means of the following equation.

$$conf(A, B) = \frac{support(A, B)}{support(A)} \tag{3}$$

On the other hand, the classifier builder component is responsible for producing a classifier out of the whole set of rules, which involves pruning and evaluating all possible rules. Pruning is also done in each subsequent pass of the rule generator. It uses the pessimistic error rate based pruning method proposed by Quinlan[47] in the in C4.5 algorithm. It prunes a rule as follows: If pessimistic error rate of rule r is higher than the pessimistic error rate of rule r_1 (obtained by deleting one condition from the conditions of r), then rule r is pruned.[6]

As stated before, we needed to extend the LUCS-KDD CBA algorithm in order to implement the CBA-Fuzzy. Generally speaking, the integration of fuzzy logic features in our algorithm consists of changing data input format in order to deal with fuzzy values and the support and confidence measures calculation. Actually, the original LUCS-KDD CBA algorithm limits the input data to have only discrete numbers on attributes' values and, in addition, they have to be ordered sequentially commencing with the number 1. Hence, the algorithm demands the analyst lots of efforts for preprocessing input data. Conversely, the CBA-Fuzzy algorithm accepts any type of attribute value, even continuous or categorical attributes. To allow such facility, the algorithm automatically performs dicretization and fuzzyfication processes for continuous attributes. In this way, it avoids some efforts of the analyst during preprocessing, because both processes are integrated in the algorithm.

The discretization process for numerical attributes can be done automatically by CBA-Fuzzy either using equal-width approach, where samples are divided into a set $V = \{v_1, v_2, v_3, \ldots, v_n\}$ of N intervals of the same length, or using equal-depth approach, where the attribute range is divided into intervals containing approximately the same number of samples (same frequency).

The general workflow of the CBA-Fuzzy algorithm is shown below, where "D" is the dataset used as input for the algorithm (training set) and D_f the dataset after the fuzzyfication process.

Algorithm 1 CBA-Fuzzy's workflow

1: D = processCSV($inputFile$);
2: V = discretize(D, $type$, N);
3: **if** $type$ = "equal-width" **then**
4: D_f = applyFuzzyTriang(D, V);
5: **else if** $type$ = "equal-depth" **then**
6: D_f = applyFuzzyTrap(D, V);
7: **end if**
8: CR_s = CBAFuzzy-RG(D_f);
9: CRM = CBA-CB(CR_s);

The input for CBA-Fuzzy and its output (a set of class association rules) are CSV (Comma Separated Value) files, which are managed using the GenJava-CSV library. Line l represents the input process, which transforms the input CSV file into a dataset "D" able to be read by other algorithm's modules. The second input parameter of line 2 represents the type of discretization the analyst wants to perform. Hence, the analyst can set up the number of intervals and the type of discretization he finds more appropriate. Subsequently, between lines 3 and 7, the type of discretization will stipulate the type of membership function used to perform the fuzzyfication process. In order to calculate the membership values of a sample of a dataset discretized using the equal-width approach, we employ a triangular membership function (three parameters). For the datasets discretized using the equal-depth approach, we employ a trapezoidal membership function (four parameters), because in this case some intervals are wider than others and, therefore, they encompass a region with a constant value defining an exclusive membership. During the fuzzyfication process it is assigned one or two membership values to each sample of the dataset, because each sample may belong to one or two intervals at the same time. The assignation of the membership value depends on the proximity of the sample's value to interval range. For example, given a hypothetical dataset containing demographical information about a certain group of people, where there is an attribute "X" referring a person's age and considering a sample "p" of this dataset who holds 24 on the "X" attribute. In order to perform the fuzzyfication process, the system groups the values of "X" in age intervals (like [1-10[, [10-15], [15-20], etc.) and discretizes the value of "p" in one or more intervals. In this case, the value of "p" is 24 and, obviously, it owns to the [20-24[interval, however, it is, at the same time, quite close to the [25-30[interval. If we label [20-25[as "teenager" and [25-30[as "adult", we may say that "p" is a teenager and a bit adult too since he is almost in the boundary of the first interval. Then, we consider that "p" owns, with different grades of membership, both to [20-25[and [25-30[intervals.

By means of the results supplied by the discretization and fuzzyfication processes, the CBA-Fuzzy provides an input dataset, in an appropriate format, to the rule generation component. An example of input dataset is shown in Table 1, where

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

each value for the attributes "Year" and "Age" encompasses two intervals with their respective membership values correlated to every sample.

Table 1. Example of input dataset to rules' generation.

ID	Author	Year	Country	Age	Rating
1	Stephen King	[1996-1998[:0.3	FR	[25-30[:0.3	Good
		[1998-2000[:0.7		[30-33[:0.7	
2	Anne Rice	[1975-1983[:0.5	ES	[30-33[:0.2	Bad
		[1983-1988[:0.5		[33-35[:0.8	
3	Stephen King	[2000-2001[:0.6	UK	[33-35[:0.5	Bad
		[2001-2003[:0.4		[35-37[:0.5	
4	Nora Roberts	[1983-1988[:0.8	BR	[40-47[:0.7	Good
		[1988-1993[:0.2		[47-50[:0.3	

On table 1 we may say that the last sample (ID=4), for instance, owns 80% to [1983-1988[and 20% to the [1988-1993[interval. That means such sample is close to the right border of the first interval (its year value must be almost 1988). A similar situation happens the Age attribute on the same sample, however, in this case, we may deduce that it is a bit more distant to the interval on the right ([47-50[) than there was on the Year attribute.

Afterwards, the CBA-Fuzzy starts the rule generation process (line 8), which is performed by an adapted version of the CBA-RG module developed by Liu et. al[6] and then the classifier is obtained (line 9) by means of a classifier builder (CBA-CB), which was basically the same version developed by such authors.

Taking into account the rule generation process, CBA-Fuzzy calculates the support measure, and consequently the confidence as well, differently from the "crisp version of CBA". In general association rules, the support of an item is calculated by counting the number of transactions it appears on data by summing 1 in every time the interval it belongs appear. However, the computation of the support of fuzzy association rules is not so trivial since it considers partial memberships represented by continuous values between 0 and 1(instead of simply summing binary values). Several methods for computing the fuzzy support of a pattern, such as the ones described in [48] and [49], may be extended for fuzzy association rules. Nevertheless, in this work we employed a classical approach for generalizing the quality measures for fuzzy association rules as it is more appropriate for the scope of recommender systems. Thus, set-theoretic operations, namely Cartesian product and cardinality, are replaced by equivalent fuzzy set-theoretic operations. The fuzzy support of a rule $A \rightarrow B$ is obtained as follows:

$$sup(A \rightarrow B) = \sum_{(i_1,i_2) \in I} A(i_1) \otimes B(i_2) \tag{4}$$

where $\otimes$ is a t-norm with $\otimes = min$. Taking into account equation 3, the fuzzy

confidence may be given as follows:

$$conf(A \rightarrow B) = \frac{\sum_{(i_1,i_2)\in I} A(i_1) \otimes B(i_2)}{\sum_{(i_1)\in I} A(i_1)} \quad (5)$$

In algorithm 2 we depict the pseudo code of our adapted version (CBAFuzzy-RG) of the CBA-RG module referred previously, where "k-itemsets" denotes an itemset having "k" items, "$Freq_k$" denotes the set of frequent "k-itemsets" and "C_k" is the set of candidate "k-itemsets".

Algorithm 2 CBAFuzzy-RG's pseudo code

1: $Freq_1 = \{$large 1-itemsets$\}$;
2: $CR_1 =$ genRules($Freq_1$);
3: $prCR_1 =$ pruneRules(CR_1);
4: $k = 2$;
5: **while** $Freq_{k-1} \neq \emptyset$ **do**
6: $C_k =$ candidateItemsetsGen($Freq_{k-1}$);
7: **for all** data case D_i such that $D_i \subset D_f$ **do**
8: $C_d =$ ruleSubset(C_k, d);
9: **for all** candidateItemset C_i such that $C_i \subset C_d$ **do**
10: **if** D_i.class $= C_i$.class **then**
11: **for all** attribute a such that $a \in D_i$ **do**
12: $lineSupport = lineSupport \times a$.support;
13: **end for**
14: $C_i.rulesupCount = C_i.rulesupCount + lineSupport$;
15: **end if**
16: **end for**
17: **end for**
18: $Freq_k = \{c \in C_k | c.rulesupCount \geq minsup\}$;
19: $CR_k =$ genRules($Freq_k$);
20: $prCR_k =$ pruneRules(CR_k);
21: k++;
22: **end while**
23: $CR_s = \bigcup_k CR_k$;
24: $prCR_s = \bigcup_k prCR_k$;

Firstly, the algorithm counts the item and class occurrences to determine the frequent "1-itemsets" (line 1). From this set of "1-itemsets", a set of CRs (called "CR_1") is generated and subjected to a pruning operation (lines 2 and 3), which was previously described. Actually, pruning is also performed in each subsequent pass

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

"k" to "CR_k" (line 20). In each subsequent pass (line 5 to 22), the algorithm generates the frequent itemsets using the same downward closure property previously described. Moreover, in each pass, the input dataset is scanned and the support count of each possibly rule is updated. The support of each rule is calculated (line 11 to 14) taking into account the membership value of each attribute (or item) composing the itemset, which represents the rule, where the membership values are multiplied to obtain a minimum value. After those new frequent itemsets have been identified to compose $Freq_k$ (line 18), then the algorithm generates the rules CR_k (line 19). Finally, rule pruning is performed (line 24) on these new rules.

3.2. *Recommendation Framework*

By means of the algorithm previously described, which is the central feature of our methodology, we are able to build the hybrid recommendation methodology as this algorithm generates a set of class association rules used for classification. Basically, the proposed methodology is composed of three main components: conception of groups of users, rule set generation and recommendation. The first two components are built off-line and are responsible for inducing the estimation model (including the class association rules generated by CBA-Fuzzy) used for making recommendations. The third component is responsible for classifying, at runtime, the active user in order to provide him personalized recommendations. In the next subsections we describe these three components.

3.2.1. *Building Groups of Users*

For the purpose of providing recommendations to a specific user, the proposed methodology needs to compare his preferences and features with the ones of other users. Hence, we find out groups of users with similar preferences and characteristics to the user we want to provide recommendations. To do so, as it will be explained in subsection 3.2.3, we classify the active user in one or more groups of users. However, before performing this task we need to obtain the groups of users. In order to obtain groups of users, we take into account information gathered from attributes containing demographic characteristics of users (such as age, postal code, level of education, etc.) and also from attributes concerning items of the system (such as year of launching, price, etc.) that users have rated or purchased. In addition to that, it may be considered users' past interactions with the system by means of implicit actions, such as time spent seeing an item, number of mouse clicks, etc. In this sense, this process may be considered as a collaborative filtering approach to provide recommendations, because the system makes use of the information gathered from other users of the system (instead of only from the active user). However, such information of other users need to be represented by shared transactions in order to delineate samples containing data attributes describing their features. Such samples are provided as an input training set to a clustering algorithm in order to generate

groups of transactions, which represent users. Figure 1 shows a diagram of activities for building such groups, which is the first step of our methodology.

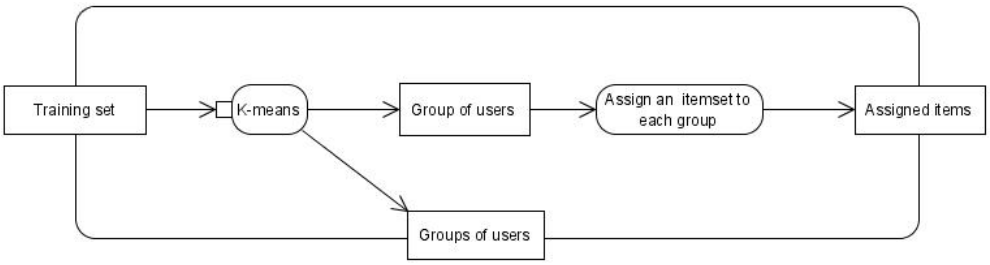

Figure 1. Building groups of users

The input of the diagram of activities is the training set to be provided to the K-Means algorithm. The training set is a selected portion of the dataset encompassing a set of significant samples enclosing the same configuration as described previously. Through this adaptation of K-Means, employing an unsupervised learning approach, we obtain a set of groups of users. Thus, the output provided by the K-Means algorithm will be a set $G = \{g_1, g_2, g_3, \ldots, g_N\}$ of groups of users, where N is a predefined number of groups. In fact, such parameter depends directly on the domain area of the system and also on the aim of the analyst. A recommender system of tourist monuments on a certain city, for example, would probably present much more distinct user profiles than a recommender system of books for a gospel community, although the first one does not necessarily have more users than the latter one. Nevertheless, it would be reasonable that the number of groups encompass a proportion between the number of users and the number of available in the system. Therefore, the number of groups is an intrinsic parameter that depends on the system requirements. Each group is represented by a register containing, for each attribute, the mean value of all samples composing the group (in case of a numeric attribute) or the most frequent value (in case of a categorical attribute).

The output (N groups of users) provided by the *K-Means* is supplied as input to the next step of our methodology (which will be described in the next subsection). Therefore, we integrated the adapted version of *K-Means* in order to have its output in the format required as input for the proposed algorithm. At this time, the initial dataset is appended with a new attribute containing the assigned cluster.

Afterwards, an ordered list of items (or products) $P = \{p_1, p_2, p_3, \ldots, p_m\}$ is assigned to each group g_i, where $i \in \{1, 2, 3, \ldots, N\}$. The top items in each list will be the ones who better represent each group, where the distance of each item to the centroid is taken into account. Alternatively, the top items may be the ones who received better evaluation from the users of the group or the most frequent ones (taking into account the number of purchases or ratings received) or any other

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

criterion defined by an expert in the domain area involving the system. In fact, the ranking of items, as well as other parameters, is directly related to the domain area of the system. Since we propose a general recommender methodology, we do not deep in this question in this work. In the end of this first phase, the sets of items assigned to users' groups will be supplied as input to the runtime process (the third step of our methodology).

3.2.2. *Generating Classification Rules*

The second step in the construction of the estimation model is the conception of the classification rules by means of the CBA-Fuzzy algorithm. Subsequently, at runtime, such rules will be responsible for classifying every new user. Figure 2 shows a diagram of activities of the rule generation process, which has two input sets: the groups of users provided as output by the K-Means algorithm and the same training set used as input in the last step.

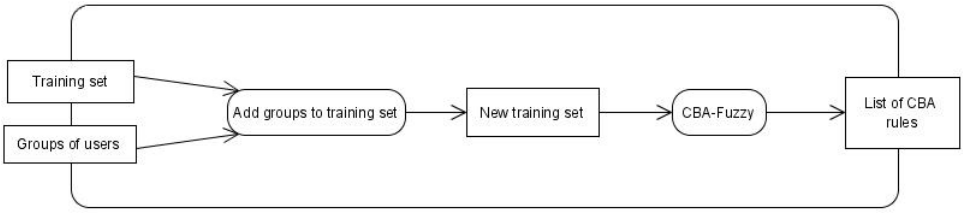

Figure 2. Obtaining CBA-rules

The first activity for generating the list of classification rules is to combine the two inputs. At this point, we add to the training set the label attribute for classification. To do so, we take into account the registers composing the groups of users and compare their values to the ones of the training set in order to fulfill the label attribute. Therefore, each sample of the training set will have an identification corresponding to a group of users. In this way, we have a new training set, which will be the input for the CBA-Fuzzy algorithm. The output provided by the algorithm will be a set of class association rules (with the same configuration described in definition 3) $R(g_i) = \{r_1, r_2, r_3, \ldots, r_p\}$, $\forall g_i \in G$. Thus, the classification model will be composed of a set of rules available for each group of users. These rules will be responsible for classifying the active user in one of those groups, because the system compares his' attributes to the rules' antecedent terms.

Each classification rule encompasses a support and a confidence value, which can be obtained according to what was stated in subsection 2.1. In this way, the confidence value expresses the degree of reliance of each rule. Therefore, before running the CBA-Fuzzy algorithm, the analyst should set up a minimum threshold value for both measures (support and confidence). It is recommended to set a high

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

value for confidence and a low value for the support, especially in a scenario evolving recommender systems, where we usually have sparse data and frequent itemsets might be less likely to occur. Moreover, as we consider all rules generated in order to classify a new user, a rule ordering scheme is not taken into account. It is important to remark that if the active user's data was in the training set (he is already part of a group of users), the class association rules would not be taken into account, because he is already classified.

3.2.3. *Providing Recommendation*

In order to provide recommendations to the active user at runtime, it is required to classify the active user using the class association rules generated before (in the second step). To do so, we take into account data gathered from active user's last interaction with the system, which is represented by a transaction "y". The data gathered from such interaction has the same configuration (same attributes) of those used to build groups of users. Since preferences might change as time goes by, such data is collected from the most recent interaction of the active user. Thus, recommendations given will be well-suited to his current preferences. Moreover, as we are using past information of the active user in order to provide him recommendations, in this context the proposed methodology may be considered as a content-based approach. Figure 4 shows a diagram of activities of the runtime phase of our methodology, where the last transaction of the active user and the list of class association rules, obtained in the second step of the methodology, are supplied as input at this point.

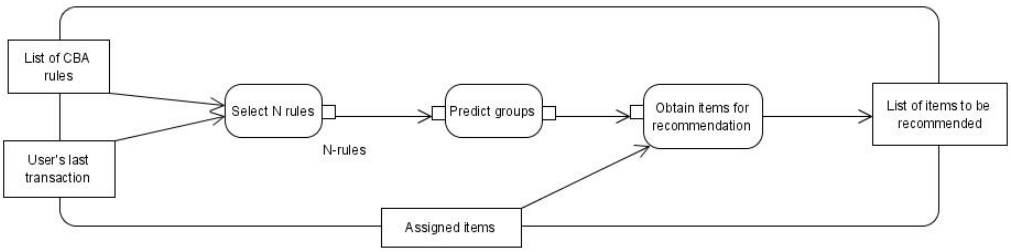

Figure 3. Runtime process

After recent information about the active user being available, the system compares the attributes' values of such data to the ones available on the class association rules generated before. Thus, we obtain a new set $R_c = \{r_1, r_2, r_3, \ldots, r_N\}$ of selected rules with "N" rules satisfying such condition, where we only consider the rules matching the values of all attributes in the antecedent terms of the first rules. Subsequently, the values of the label attribute (consequent term) of the rules in Rc are considered in order to obtain the possible groups to which the active user

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

owns to. At this point, we calculate his membership function to every class (group of users) found on R_c's rules consequent terms. To do so, a discriminator function "g" is defined in order to calculate the degree of truth that the active user owns to every class found in R_c. Considering that the active user is represented by a transaction "y" and "h" represents a group of users, the discriminator function can be calculated by means of the following equation

$$g_h(y) = \sum_{1 \leq k \leq M, i = C_h} \prod_{j=1}^{l_k} B(j,k)[X(j,k)(y)] \tag{6}$$

where l_k is the number of terms (attributes) in each rule, $X(j,k)(y)$ the value taken by the attribute $X(j,k)$ in the sample "y" and $F(j,k)[X(j,k)(y)]$ its degree of membership. Hence, such function calculates the product of attributes' degrees of membership for all the rules in R_c and then sum the results obtained in each rule.

After obtaining a discriminator value for each class (or group of users), the system compares them in order to find the greatest values. In this context, our approach is different from most classification based on fuzzy association approaches, because, generally, they only predict one single class label for a given instance. Conversely, our methodology might consider more than one class, because the analyst defines a minimum discriminator threshold value. Those classes satisfying such threshold will be the ones taken into consideration. In this way, we have "t" groups of users related to the active user. At this point, we make use of the sets of items, obtained in the first step of the methodology, assigned to the groups of users, which is the third component provided as input to the runtime step. The recommendation provided to the active user will be a suggestion containing the "n" best ranked items of each list. In order to have a constant number of recommended items, "n" will be inversely proportional to "t", because as more classes we have, less items will be considered in each list.

4. Case Study

In this section we describe a case study in order to evaluate the methodology proposed in this work. Therefore, we led two experiments to compare classifiers' accuracy and another one to compare the number of false positives issued. The first experiment considers the purpose of employing associative classifiers in recommender systems. Besides considering algorithms' accuracy, we also analyzed how the performance of associative classifiers may be affected by sparsity. The second experiment compares a general classification based on association approach with the one based on fuzzy logic that was implemented in this work. The last experiment compares the false positive rate of the proposed algorithm with other classifiers commonly used in recommender systems. The last experiment compares the false positive rate of the proposed algorithm with other effective supervised learning classifiers. In or-

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

der to accomplish these experiments, we analyzed algorithm's performance in five datasets: one obtained from the MovieLens database and four obtained from the BookCrossing database. The database of MovieLens consists of movies ratings made by users in 2000, which is a recommender system based on the GroupLens technology. Such database is freely available for research purposes on the GroupLens Web page. The database of BookCrossing consists of book ratings gathered by Ziegler et. al.[50] from the Book-Crossing community. Users from this community exchange books and experiences all around the world. For both databases the WEKA[51] tool was used to perform data transformation and pre-processing.

For all experiments done in this case study, we applied the ten-fold cross-validation method to estimate algorithms' classification accuracy. This method randomly divides the dataset into ten disjoint subdatasets, each one containing roughly the same number of records. For each subdataset, a classification model is built using the records that were not employed for building it. The classification model is then evaluated on the pending subdataset to obtain a cross-validation estimate of its accuracy. At last, the method calculates the average of all classification models' accuracy and then provides an approximation for the classifier built from the whole dataset.

In the next subsections we detail both databases and how they were used and, subsequently, the experiments accomplished on such databases.

4.1. *MovieLens Data*

Initially, the MovieLens dataset contained approximately 100,000 ratings for 1,682 movies made by 943 users, where we integrated the data related to users and movies into one file, which was the input provided for the algorithms analyzed in this case study.

However, before supplying such input we changed the rating attribute in order to have only two values: "Not recommended" (score 1 or 2) and "Recommended" (score 3, 4 or 5). The first one refers to an item the user may like and the second refers to the opposite case. Such changes were performed to simplify classification, because the main aim in a recommendation task is to determine if an item should be offered to the user or not. Taking into account users' data, we used the following attributes: gender, age and occupation. The age attribute was discretized in five age ranges The user's occupation attribute is a nominal variable with 21 distinct values.

Taking into account movies' data, the file provided by MovieLens originally contained 19 binary attributes related to movie genres. An instance with value 1 expressed that the movie belongs to a specific gender and 0 otherwise. The association model's consistency would be compromised if 19, among the 23 attributes on the dataset, were binaries. Thus, these 19 binary attributes were reduced to just one attribute representing the movie genre's name. However, since some movies may belong to different film genres, we only used the records containing ratings

about movies with just one genre. Afterwards, 7 film genres were not considered in this study. Hence, association rules generated by the model could express relationships between user profiles' characteristics and film genre features. After data pre-processing and transformation, 14,587 records were remained in the input file for the algorithms used in this study.

4.2. *BookCrossing Data*

Initially, the Book Crossing data contained 1,149,780 ratings about 271,379 books provided by 278,858 users. However, such ratings include explicit (an assigned mark from 1 to 10) and implicit (written reviews) ratings. Thus, the implicit ratings were not considered for this study and the dataset remained with 433,671 records.

In order to simplify the classification, the rating attribute was modified in the same way as the MovieLens was: "Not recommended" (score from 1 or 6) and "Recommended" (score from 7 to 10). For books' data, we used two attributes from the dataset: publication Year and Author. The first was discretized in five ranges. The Author attribute was also modified, because at first it encompassed 48,234 distinct values. Thus, the dataset was reduced in order to this attribute encompasses only 40 distinct values (the ones that appear on more records). This decision is based in the fact that a rule based classification model considers the most representative portions of the dataset in order to generalize data patterns in it, therefore most non frequent attributes values are not relevant to the learning model conception.

On the other hand, taking into account users' data, we also used two attributes: Age and Place where the user inhabits. Though some of these attributes could not seem relevant for recommendation, we believe that most attributes of any dataset may be a trace of user's preference, where such relation is not usually intuitive. To users from a certain city, for example, books of a particular genre may be very popular due to some unknown reason. Therefore, such patter may enrich the classification model.

The age attribute was discretized in nine age ranges. The Place attribute originally contained the name of the city, the state or province, and the name of the country. However, in this way such attribute presented 12,952 distinct values. Therefore, we changed this attribute in order to encompass only 40 distinct values. For that reason and noticing that 75% of the places were from USA, we divided the dataset, based on this attribute, in two: places grouped by states of USA and places grouped by countries excepting USA. Afterwards, the first dataset (states of USA) remained with 25,523 records and the second one (countries) remained with 8,926 records.

In order to try the algorithms' accuracy facing a smaller range of distinct values we also used two more datasets derived from those ones mentioned before. Thus, we copied both datasets and kept only 10 distinct values (the most frequent) for

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

author and Country/State attributes. In this way, we obtained two more datasets that contain 6,270 records (on the dataset of states of USA) and 3,238 records (on the dataset of countries).

4.3. *Analyzing Associative Classifiers*

The first part of the case study described in this work consists of analyzing the performance of associative classifiers in comparison to traditional classifiers commonly employed in recommender systems and also to a fuzzy rule-based classification algorithm called FURIA (Fuzzy Unordered Rule Induction Algorithm)[52], which is based on the RIPPER algorithm.

At this point, we analyzed the following classifiers: C4.5, BayesNet, FURIA, CBA, CPAR and CMAR. The first three were run through WEKA and the other three were obtained from the LUCS-KDD repository. The main aim of this study was to compare the algorithms accuracy using data gathered from recommender systems and to verify if it is feasible, in terms of precision, to employ associative classification in these systems.

In order to analyze how the performance of associative classifiers may be affected by sparsity, we needed to address some questions related to sparsity: how can we nominate whether a dataset is sparse or not and if it is possible to measure the degree of sparsity of a dataset. Due to practical reasons sometimes industry and academy evaluate sparsity considering the number of NULL/NA values presented by a certain dataset. In this sense, sparsity may be seen as density, which reflects both the overall size of the recommendation's item space and the degree to which users have explored it.[53] In this context, an example is described in [54], in which there is a dataset with four attributes in a retail market scenario: store, week in a year, costumer and product. If there is 1000 stores, 52 weeks in a year, 500,000 customers and 10,000 products, the dataset has 1,000x52x10,000x500,000 = 260,000,000,000,000 potential cells. However, it might only have 1,872,000,000 populated cells, because there are 450,000 customers purchasing on average 26 times a year, buying 40 products at just 2 stores. Thus, such dataset is 0.00036% sparse ((936,000,000 / 260,000,000,000,000) x 100).

Before analyzing classifiers' accuracy, we analyzed how sparse the five datasets used in the case study, thus we used the approach taken in [54]. In Table 2 we depict the density of each dataset.

As shown in Table 2, the "BCrossing World" and "BCrossing USA" are the sparsest datasets. Not surprisingly, the two datasets with reduced number of distinct values, "BCrossing USA10" and "BCrossing World10", are less sparse than their correspondent datasets with more distinct values. Conversely, the MovieLens is the densest dataset used in this study (0.86). As expected, the MovieLens dataset is used in mostly recommender systems' works, because its density makes it easier to develop trustful case studies.

In order to perform the experiments proposed in this case study and to evaluate

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

Table 2. Datasets' Density.

Dataset	Records' Number / Product of Distinct Values
BCrossing USA	25,523 / 144,000 = 0.18
BCrossing USA10	6,270 / 9,000 = 0.7
BCrossing World	8,926 / 144,000 = 0.062
BCrossing World10	3,228 / 9,000 = 0.36
Movielens	14,587 / 17,052 = 0.86

the classification based on association algorithms, we define the same values for the support and confidence thresholds on every algorithms. Due to the sparse nature of the databases used in this case study we defined a very low support threshold value (5%), for experiments made on associative classifiers, in order to obtain enough frequent itemsets. Conversely, we defined high values of confidence threshold: 70% for the sparsest datasets ("BCrossing World" and "BCrossing USA"); 75% for "BCrossing World 10" and "BCrossing USA 10", which are less sparse than the other two; and 85% for the densest dataset (MovieLens). For the datasets of BookCrossing containing 10 distinct values for Author and Country/State attributes, we increased the support threshold to 10% due to its reduced number of records.

It is also important to highlight the number of positives ("Yes") values of the databases employed. Originally, the positives rate for Movielens was 56% and for Book Crossing was 65%.

In Table 3 we show the results obtained after running the algorithms mentioned above. Each line, except on the last column, depicts the average accuracy, on the ten-fold cross validation, on each classifier, which is defined as the percentage of the correct classified samples among the whole data taken into account. The last column shows the density of each dataset.

Table 3. Comparison of Classifiers.

Dataset	Bayesnet	C4.5	FURIA	CBA	CPAR	CMAR	Density
Movielens	81.95%	82.88%	82.72%	81.4%	74.07%	**83.28%**	0.86
BCrossing World	80.87%	80.21%	**81.31%**	79.47%	73.25%	59.55%	0.062
BCrossing World10	80.51%	79.98%	80.9%	**81.28%**	79.86%	33.51%	0.36
BCrossing USA	80.23%	81.31%	**81.33%**	80.23%	78.15%	67.30%	0.18
BCrossing USA10	81.53%	80.82%	81.35%	**81.56%**	76.71%	56.86%	0.7

Results revealed that the associative classifiers reached similar accuracy, except CMAR on Book Crossing data, to traditional classifiers (supervised learning). Actually, in some cases associative classifiers reached higher accuracy. Despite the fact of being the first method of classification based on association, the CBA algorithm reached the highest accuraccy on two of the four datasets of Book Crossing. On MovieLens data, the CMAR reached the highest accuracy, which was the best

result obtained over all the experiments.

Since rules provided by the associative classifiers hold a high confidence value (70% or 85%), the rules used for building the classification models are reliable. The ninth rule generated by CMAR on MovieLens data is an example of this kind of rule: "age=*[25-34]* & genre=*drama* ⇒ rating=*yes*". This rule states that if a user is older than 25 and younger than 34, he will probably rate positively a drama movie.

In spite of presenting the highest precision over all experiments (83.28% on MovieLens data), CMAR did not present satisfactory results on Book Crossing data. MovieLens and Book Crossing data drastically differ in relation to sparsity. In general, for sparser datasets, CMAR presented worse results, because, as shown in Table 2, the datasets of the countries are sparser than the datasets of states of USA and, as shown in Table 3, there was a great loss of accuracy on "BCrossing World" compared to "BCrossing USA" and on "BCrossing World 10" compared to "BCrossing USA 10". On the other hand, CBA did not present a great loss (less than 1%) of accuracy on the datasets of countries. A similar scenario occurred for the datasets with 40 distinct values compared to the ones with 10 distinct values, to which CMAR presented a loss of accuracy. These datasets own substantial less records (around 75%) than the ones with 40 distinct values, which means they are less likely to present frequent itemsets and to identify relationships to build rules. Conversely, CBA did not lose accuracy in these datasets. Due to such outcomes, we argue that CMAR is more effective on datasets that encompass attributes with less distinct values.

At last, the CPAR algorithm also presented acceptable results, even though its precision was slightly lower than ones of other classifiers. Such algorithm is more effective for scenarios of very large datasets where processing time may be a critical issue, because the classifier construction and the rules induction are made in just one processing step. However, in the context of this work, the response time to the user is not a critical issue, because the recommendation model we propose is built off-line.

In order to compare the classifiers employed in this case study, we have used some statistical tests and procedures. According to García et. al.[55], statistical analysis is highly demanded in any research work and allows us to determine whether the obtained results are significant with respect to the choices taken and whether the conclusions achieved are supported by the experimentation. In this context, we employed some non-parametric statistical procedures to compare the algorithms' accuracy over multiple datasets. To do so, we employed the software provided by García and Herrera[56], which provides Java implementations of statistical tests and procedures. In this way, we firstly considered the Friedman[57] test for the five classifiers analyzed above. Such statistical test computes the average rank R_i of each one of the "k" classifiers over "N" datasets.

By means of the software implemented in [56], we were able to obtain the ranks of the five classifiers under study through the Friedman statistic, which was distributed according to chi-square with 5 (or $k-1$) degrees of freedom. The approximate

value computed for the Friedman statistic corresponding to the input datasets, where $N = 6$, was 11.97. On table 4 we show the average ranking obtained by the software mentioned above for the five classifiers under study.

Table 4. Average ranking of Classifiers.

Algorithm	Ranking
BayesNet	3.0
C4.5	3.0
FURIA	2.0
CBA	2.8
CPAR	5.2
CMAR	5.0

Through table 4, we can verify that the null hypothesis, which states that all classifiers are equivalent, is rejected, because the five ranks computed are not equal. By means of the algorithms ranking we can clearly identify two groups of classifiers. One group with lower performance (ranking greater than 5) and another one with greater performance (ranking between 2 and 3). The latter encompasses four classifiers, where FURIA's average ranking is slightly greater than the other three. Thus, through this first statistical test, we are able to verify that fuzzy rule learners are able to be effective in recommender systems in the same way they are in other domains. Moreover, we are able to verify that the CBA algorithm may be as effective as other known machine learning classifiers. However, in order to assure that fuzzy rules and associative classifiers may be employed effectively in recommender systems, more experiments and statistical tests need to be performed.

According to Demšar[58], as the null-hypothesis was rejected, we can proceed with a post-hoc test. At this point, we want to perform $n \; x \; n$ comparisons of classifiers. In order to perform multiple comparisons we employ the Shaffer's statistic procedure [59], which, according to García et. al.[55], is one of the most powerful statistical procedures available. To do so, we also employed the same software mentioned above to find the p-values (corresponding probabilities) associated to comparisons between two classifiers. A p-value represents the lowest level of significance of a hypothesis that results in a rejection.[56] Thus, we may estimate how different two classifiers are, because the null-hypothesis affirms that such classifiers have similar performance and the p-value expresses the probability error of such comparison.

However, in this experiment we employed adjusted p-values (APV), which according to García and Herrera [56], provide more information in a statistical analysis. In table 5 (obtained through the same software mentioned above) we show all possible pairwise comparisons (second column), the non-adjusted p-values (third column) and the adjusted p-values for the Shaffer's static procedure (last column). The results displayed in table 5 were all obtained through the software mentioned above,

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

where the Shaffer procedure rejects those hypotheses that have an adjusted p-value lower or equal to the significance level previously defined ($\alpha = 0.05$).

Table 5. Adjusted p-values for the Shaffer's procedure.

i	hypothesis	unadjusted p	p_{Shaf}
1	FURIA vs .CPAR	0.0068	0.1026
2	FURIA vs .CMAR	0.0112	0.1123
3	CBA vs .CPAR	0.0425	0.4252
4	C4.5 vs .CPAR	0.063	0.6298
5	BayesNet vs .CPAR	0.063	0.6298
6	CBA vs .CMAR	0.063	0.6298
7	C4.5 vs .CMAR	0.0909	0.6368
8	BayesNet vs .CMAR	0.0909	0.6368
9	BayesNet vs .FURIA	0.398	2.786
10	C4.5 vs .FURIA	0.398	2.786
11	FURIA vs .CBA	0.4996	2.786
12	BayesNet vs .CBA	0.8658	3.463
13	CPAR vs .CMAR	0.8658	3.463
14	C4.5 vs .CBA	0.8658	3.463
15	BayesNet vs .C4.5	0.999	3.463

Results on table 5 shows that all null hypotheses were accepted, which means that every two algorithms compared on the ten hypotheses are not significantly different. This assures that the three associative classifiers analyzed may present similar performance to the other classifiers analyzed. Nevertheless, the last seven comparisons present a considerable greater adjusted p-value (2.79 and 3.46) than the first eight. Seeing that the higher the p-value is, the stronger the evidence is in favour to the null hypothesis, we may say the last seven comparisons are much farer to the threshold limit for hypotheses rejection (0.05) than the first ones. Thus, we can clearly identify two groups of classifiers: BayesNet, C4.5, FURIA and CBA; and CPAR and CMAR. Such scenario reinforces the conclusion obtained through the Friedman test, where the same groups of classifiers were found. In this way, we may presume that the CBA algorithm is powerful as other known and broadly employed machine learning methods.

Through these experiments we were able to conclude that classification based on association methods can be employed effectively in recommender systems, because they can reach similar, or even greater, precision to traditional classifiers, and also because rules obtained by the classification model are feasible due to the high value of confidence they present. Moreover, by means of the FURIA's results, we may conclude that fuzzy rules are able to provide good results in recommender systems, what encourages more us to develop a fuzzy associative classifier for these systems.

Besides allowing to offer recommendations on real time (because the estimation model is built off-line), associative classifiers provide, as they are composed of trivial class association rules, an estimation model easy to be interpreted. It allows the

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

analyst to interfere and interpret easily the classification model generated. Moreover, a recommendation model made of class association rules allows us to deal with effects caused by the "grey sheep" problem.

4.4. *CBA vs. CBA-Fuzzy*

Taking into account the results obtained in subsection 4.3, we argue that the CBA algorithm is the most suitable associative classifier to be employed in recommender systems, because, as seen in subsection 2.4, these systems are generally affected by the sparsity drawback. And on the other hand, CMAR presented poor results for sparse data. This scenario was the main reason to make use of the CBA algorithm to implement the CBA-Fuzzy algorithm described in subsection 3.1. Actually, the main contribution of CMAR, as well as other newer associative classifiers, is relative to memory usage and processing time, because accuracy was slightly the same (5% is not considerable in recommendation scenario). Thus, current associative classifiers would not supply many benefits to our methodology, because the estimation model is built off-line. In this context, CBA may be employed efficiently in recommender systems as a model based collaborative filtering method.

Here we present an experiment that compares the performance of the general CBA algorithm, obtained from LUCS-KDD repository, with the CBA-Fuzzy algorithm implemented in this work. We basically want to verify if the proposed algorithm is able to present at least similar results to its ancestor (CBA) when applied to recommender systems data. We evaluate both discretization approaches performed by CBA-Fuzzy: equal-width (ew) and equal-depth (ed). In this experiment we did not employ statistical tests, because here we are evaluating just two algorithms and, what is more, they are quite similar, except for the fact that one considers fuzzy sets.

For both approaches and in both MovieLens' and Book Crossing's datasets, we set ten intervals (parameter N of CBA-Fuzzy) for the discretization process of the numerical attributes (Age in MovieLens; Age and Year of Publication in BookCrossing), because such attributes do not own great discrepancies. Then, in the fuzzyfication process, CBA-Fuzzy applied, in total, 20 membership functions on the datasets of Book Crossing and 10 on MovieLens data.

In order to provide discretized data to the CBA version of LUCS-KDD, the WEKA tool was employed. In this experiment we also consider the number of rules generated by each algorithm, because we want to obtain a classification model that is reliable and then applied to a real scenario. Actually, the number of rules generated can define the interpretability of an associative classifier. So that, the support and confidence thresholds have been established aiming to obtain an acceptable number of rules to build a classification model suitable for a recommender system. In this case study, we wanted to generate between 80 and 140 classification rules. An excessive number of association rules is a critical interpetability issue, however, in a recommendation scenario, where even active users may have purchased less than

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

1% of the items available, data is usually very sparse and sometimes it is difficult to obtain a minimum quantity of rules. Hence, if we have more rules, we are able even increase the rules' confidence.

We considered a threshold value of 1% for the support measure to run both algorithms. For the dataset of Movielens and to the datasets of BookCrossing with 10 distinct values, we defined a threshold value of 80% for the confidence measure. However, for the datasets owing more distinct values ("BookCrossing World" and "BookCrossing Usa"), we reduced such threshold to 75% because they are the sparsest datasets employed in this study (see Table 2) and, as a consequence, they are likely to present less frequent itemsets. Table 6 shows the results after running CBA and CBA-Fuzzy on the datasets mentioned above. Each line depicts the average accuracy and the average number of rules obtained on each dataset. It should be noted that the CBA-Fuzzy algorithm classifies each instance in only one class, because the membership values defined on the fuzzyfication process are used at runtime, as seeing in subsection 3.2.3, to classify the active user in two or more classes.

Table 6. Comparison between CBA and CBA-Fuzzy.

Dataset	CBA(ew)	CBA-Fuzzy(ew)	CBA(ed)	CBA-Fuzzy(ed)
BCrossing USA	47.01% - 44.4R	17.01% - 16.4R	82.1% - 85R	**82.22% - 89.7R**
BCrossing USA10	79.77% - 85.6R	80.4% - 80R	79.92% - 94.1R	**81.04% - 135.6R**
BCrossing World	**79.65% - 91.5R**	75.24% - 84.1R	79.04% - 105.6R	70.78% - 117.8R
BCrossing World10	78.2% - 72.9R	78.19% - 73.9R	78.65% - 69.2R	**80.1% - 102.3R**
Movielens	79.29% - 93.4R	78.82% - 88.9R	**79.68% - 92.1R**	78.62% - 139R

Table 6 shows that the best results were obtained on the datasets that were discretized using ranges of values of the same frequency (equal-depth). This scenario is more obvious on Movielens data, because accuracy has not overcome the threshold of 50% for the equal-width approach. The poor results obtained by equal-width in such dataset are reasonable since intervals automatically generated with this method do not reflect the real context of data.

Moreover, it should be remarked that on "BCrossing World 10" dataset (the one with less records), the equal-depth approach were not superior. It can be justified because intervals of values in this dataset were less frequent due to its reduced number of records and, consequently, both algorithms have been unable to benefit from such approach. This scenario is more evident on CBA-Fuzzy, because in such dataset obtained the lowest accuracy with this approach.

Comparing the two algorithms employed in this case study, the CBA-fuzzy, in almost all cases, generated more rules. This might be considered a positive scenario to the CBA-Fuzzy, sparse data requires a sufficient number of rules to cover the majority of the values of its attributes. Moreover, the rules obtained encompass a high confidence value. Furthermore, CBA-Fuzzy obtained greater accuracy than CBA in three of the five datasets evaluated with equal-depth discretization.

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

This scenario lets us conclude that the algorithm developed in this work can be used effectively in recommendation systems. Despite the fact of not having a major difference of accuracy compared to other methods, our purpose benefits, as being a hybrid methodology, from advantages of both categories of methods, mainly aiming at minimizing drawbacks of collaborative filtering methods. The first-rater problem, for example, is minimized through the use of a content-based method, because every new item added to the system can be related to a group simply analysing values of its attributes. Shortcomings related to data sparsity are also reduced, because both estimations models employed are built off-line. Moreover, the problem of the "grey sheep" is also drastically minimized when using fuzzy logic, because the active user can be member of more than one group at the same time. There are many situations where the user does not have a clear relationship with any group, but it can present vague (or even ambiguous) relationships with two (or more) groups. On the other hand, the dataset employed needs to encompass a reasonable number of records to build a feasible classification model.

4.5. *Analyzing False Positive Occurrence*

In order to analyze the number of false positives the proposed algorithm presents in comparison to other classifiers, in this subsection we took into account the false positive rate presented by each algorithm on the same datasets used previously. Therefore, we analyzed the same "non associative classifiers" employed in subsection 4.3 (FURIA, Bayesnet and C4.5). We basically wanted to verify if the classification based on association algorithms, especially the CBA-Fuzzy, would not make more false positives than other classifiers, because, as stated is section 3, the occurrence of false positives in recommender systems may cause very negative effects.

In order to calculate the false positive rate, we employed the same approach defined by Fawcett[60], which is stated in Definition 1.

Definition 1. FP rate = negatives incorrectly classified / total negatives

The false positive rate is also called "false alarm rate", because it counts negative instances (samples not owning to the class c_1 being analyzed) that were incorrectly classified. On the other hand, the true positive represents the instances owing to c_1 that were correctly classified (also called hits). An ideal "conservative classifier" is the one which classifies positives instances (the ones owning to c_1) only with strong evidence, as consequence they make few false positive errors, conversely its true positive rates is reduced as well, and therefore, the precision is also reduced. Nevertheless, conservative classifiers are appropriate for a recommender system scenario, in which false positives need to be avoided.

It should be noted that, in our approach, we do not consider a default rule to classify an instance which would not match any rule generated, as would be done in the other associative classifiers. This would indeed lead to recommend an item that does not match the user's needs. In this way, the methodology proposed in this

Author-produced version of the article published in International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems, 2012, 20(4), 579-617.
The original publication is available at http://www.worldscientific.com
DOI : 10.1142/S0218488512500274

work does not classify a user whose data does not match with any rule generated, so that the versions of the CBA-Fuzzy and CBA, which was used as an intermediate step for the implementation of CBA-Fuzzy, follow such approach in the case study. Conversely, other classifiers always classify the active user as they classify every sample provided as input. Moreover, instances classified in our approach are based on rules holding high confidence values (from 70% to 85%).

Table 7 shows the false positive rates obtained by Bayesnet, C4.5, FURIA, CBA and CBA-Fuzzy on the same datasets employed in the previous subsection, where we also set the same confidence and support values of the experiment described on such subsection. We considered the e-depth approach for discretization because, as shown in subsection 4.4, it accomplished better results.

Table 7. Example of input dataset to rules' generation.

Dataset	BayesNet	C4.5	FURIA	CBA	CBA-Fuzzy
Movielens	47.4%	42.6%	37%	33.22%	**32.08%**
BCrossing World	45.15%	39.9%	34.9%	**23.03%**	31.89%
BCrossing World10	40.3%	42.45%	37.65%	33.83%	**32.88%**
BCrossing USA	47.45%	41.55%	36.36%	20.43%	**18.89%**
BCrossing USA10	48.25%	44%	36.35%	34.66%	**28.63%**

Table 7 shows that associative classifiers, especially CBA-Fuzzy, obtained a false positive rate around 10% lesser than the other two classifiers. The false positive rate for the "BCrossing USA" dataset, for example, was 21.12% and 27.02% lesser on CBA-Fuzzy in comparison with Bayesnet and C4.5. The FURIA, which is a fuzzy rule-based classifier, also obtained less false positives rates when compared to BayesNet and C4.5. However, it did not present less false positives than the associative classifiers analyzed, especially than CBA-Fuzzy. The big differences of false positives rates among classifiers is related to the high confidence threshold values set for the rule-based classifiers. In accordance with the Fawcett [60] defines this kind of classifiers as "conservatives", because they make positive classi

cations only with strong evidence (just rules with high confidence were accepted on the classification model), so they make few false positive errors and, on the other hand, tend to have low true positive rates as well.

In order to compare the false positives rates of the four classifiers analyzed within this experiment, we also employed some statistical tests. However, since in this case lower false positives rates are preferred instead of greater ones, we considered inverse rate values for the statistical tests' input, where, instead of considered the real value (45%, for example), we subtracted the real value from 100%. In table 8 we show the algorithms' average ranks obtained through the Friedman test, which was distributed according to chi-square with 4 degrees of freedom and the approximate value computed corresponding to the input datasets was 16.8.

In table 8 we can verify that the null hypothesis also is rejected, because the

Table 8. Average ranking of Classifiers.

Algorithm	Ranking
BayesNet	4.8
C4.5	4.2
FURIA	2.6
CBA	2.0
CBA-Fuzzy	1.4

five ranks computed are not equal. The proposed algorithm, CBA-Fuzzy, is the first at such ranking. As the null-hypothesis was rejected, we can proceed with a post-hoc test. At this point, we want to perform a *1 x n* comparison of classifiers, because, according to Demšar[58], pairwise comparisons should not be made in the case of testing whether a newly proposed method is better than the existing ones. Therefore, the best ranked algorithm (or control classifier) is compared to the others. To do so, we employed the Hochberg's procedure [61], which, according to García et. al.[55], is one of the most powerful statistical techniques for multiple comparisons. To compute the values related to the Hochberg's procedure and the Friedman test, we also employed the software provided by García and Herrera[56]. Table 9 displays the adjusted p-values for the Hochberg's procedure (last column), the non-adjusted p-values (third column) and the four algorithms compared to CBA-Fuzzy (second column). Such procedure rejects those hypotheses that have an adjusted p-value lower or equal to the significance level previously defined ($\alpha = 0.05$).

Table 9. Adjusted p-values for the Hochberg's procedure.

i	algorithm	unadjusted p	p_{Hoch}
1	BayesNet	6.74E-4	0.0027
2	C4.5	0.0051	0.0153
3	FURIA	0.2301	0.4603
4	CBA	0.5485	0.5485

Results revealed that the first two hypotheses were rejected, which means that CBA-Fuzzy is significantly better than BayesNet and C4.5 when the false positives rate is analyzed. Nevertheless, the p-values obtained for FURIA and CBA were not considerably high, which means that CBA-Fuzzy is slightly better than both considering false positives generation.

5. Conclusions

In this work we propose a new methodology of recommendation employing fuzzy logic and classification based on association. By means of the experiments described

in section 4, we show that both techniques can be effectively applied to recommender systems and might also improve recommendations' consistency. Through the methodology we described in section 3, we intend to decrease errors on recommender systems, because they still encompass several shortcomings.

Associative classification provides a fast and comprehensible learning model, differing from the majority of traditional classifiers. Another major achievement of associative classifiers in this work was the few false positives that would be made in recommendations, because developing an associative classifier adapted to recommender systems allowed us to drastically reduce false positive rate in comparison with other classifiers. On the other hand, false negatives might occur easier, because certainly all rules do not include information about all datasets' relations. However, if an item of interest to a user was classified as "Not Recommended", it would not be a critical error like a false positive would be.

As we could see in the case study described in this work, the accuracy of classification based on association methods has a straight correlation to data's attributes characteristics. Therefore, the CBA-Fuzzy algorithm achieves some advances for a recommendation scenario, because it performs automatic discretization and definition of degrees of membership to the generated intervals and, hence, it brings more significance and value to data.

Moreover, since we employ collaborative filtering and content-based approaches, the proposed methodology may be considered as a hybrid method. Firstly, as it employs historical data from other users, the characterization of the groups and the classification model are collaborative filtering methods. On the other hand, as our methodology considers active user's past behaviour to determine which group he belongs to, it can be viewed as a content-based method as well. In addition to that, the definition of the list of items in each group is a content-based approach. Seeing that our methodology consists of a hybrid approach, it can benefit from advantages of both categories of methods in order to minimize common drawbacks of recommender systems.

In spite of the efforts made for dealing with recommender systems' limitations and drawback, there are still numerous challenges in this area. In addition, the number of users and items are constantly increasing in web systems due to their popularization. In this way, there are innumerous perspectives associated to this work as it may be possible to reduce recommender systems' limitations even more. The proposed methodology could be extended to deal with other less frequent drawbacks and new limitations that might emerge in future (generally resulted by sparsity). Additionally, the threshold value defined for the support measure for the CBA-Fuzzy could be automatically set up according to specific conditions of a certain recommender system, with special attention to the sparsity property of its database. Moreover, other methods for computing the fuzzy rules support may be tested or even designed specifically for recommender systems in order to reach recommendations with even higher precision. Furthermore, more data mining methods, using supervised or non supervised learning, may be combined in order to try to reduce

even more the number of false positives.

References

1. B. M. Sarwar, G. Karypis, J. A. Konstan, J. Reidl, Item-based collaborative filtering recommendation algorithms, in: World Wide Web, 2001, pp. 285–295.
2. J. B. Schafer, J. A. Konstan, J. Riedl, E-commerce recommendation applications, Data Mining and Knowledge Discovery (2001).
3. K.-W. Cheung, J. T. Kwok, M. H. Law, K.-C. Tsui, Mining customer product ratings for personalized marketing, Decision Support Systems 35 (2) (2003) 231–243.
4. C.-H. Lee, Y.-H. Kim, P.-K. Rhee, Web personalization expert with combining collaborative filtering and association rule mining technique. Expert Systems and Applications. 21 (3) (2001) 131–137.
5. R. Agrawal, T. Imielinski, A. Swami, Mining association rules between sets of items in large databases, in: P. Buneman, S. Jajodia (Eds.), Proceedings of the 1993 ACM SIGMOD International Conference on Management of Data, Washington, D.C., 1993, pp. 207–216.
6. B. Liu, W. Hsu, Y. Ma, Integrating classification and association rule mining, in: Knowledge Discovery and Data Mining, 1998, pp. 80–86.
7. Y. Sun, A. K. C. Wong, Y. Wang, An overview of associative classifiers, in: S. F. Crone, S. Lessmann, R. Stahlbock (Eds.), Proceedings of the International Conference on Data Mining, Las Vegas, Nevada, USA, CSREA Press, 2006, pp. 138–143.
8. Y. Sun, Y. Wang, A. K. C. Wong, Boosting an associative classifier, IEEE Trans. Knowl. Data Eng. 18 (7) (2006) 988–992.
9. F. Thabtah, P. Cowling, Y. Peng, Mcar: multi-class classification based on association rule, in: AICCSA '05: Proceedings of the ACS/IEEE 2005 International Conference on Computer Systems and Applications, IEEE Computer Society, Washington, DC, USA, 2005, pp. 33–I.
10. W. Li, J. Han, J. Pei, CMAR: Accurate and efficient classification based on multiple class-association rules, in: ICDM, 2001, pp. 369–376.
11. X. Yin, J. Han, Cpar: Classification based on predictive association rules, in: SIAM International Conference on Data Mining (SDM03), 2003, pp. 331–335.
12. B. M. Sarwar, G. Karypis, J. A. Konstan, J. Riedl, Analysis of recommendation algorithms for e-commerce, in: ACM Conference on Electronic Commerce, 2000, pp. 158–167.
13. G. Chen, H. Liu, L. Yu, Q.Wei, X. Zhang, A new approach to classification based on association rule mining, Decision Support Systems 42 (2) (2006) 674–689.
14. Y. Wang, Q. Xin, F. Coenen, A novel rule ordering approach in classification association rule mining, in: The 5th International Conference on Machine Learning and Data Ming (MLDM'2007), Springer LNAI 4571, 2007, pp. 339–348.
15. C. M. Kuok, A. Fu, M. H. Wong, Mining fuzzy association rules in databases, SIGMOD Record 27 (1998) 41–46.
16. T.-P. Hong, Y.-C. Lee, An overview of mining fuzzy association rules, in: Fuzzy Sets and Their Extensions: Representation, Aggregation and Models, 2008, pp. 397–410.
17. J. Alcalá-Fdez, R. Alcalá, M. J. Gacto, F. Herrera, Learning the membership function contexts for mining fuzzy association rules by using genetic algorithms, Fuzzy Sets Systems 160 (7) (2009) 905–921.
18. G. Chen, Q. Wei, Fuzzy association rules and the extended mining algorithms, Information Sciences 147 (2002) 201–228.
19. D. Dubois, E. Hüllermeier, H. Prade, Fuzzy methods for case-based recommendation

and decision support, Journal of Intelligent Information Systems 27 (2) (2006) 95–115.
20. W. JIN, Fuzzy classification based on fuzzy association rule mining, Ph.D. thesis, Graduate Faculty of North Carolina State University (2004).
21. Z. Chen, G. Chen, An approach to classification based on fuzzy association rules, in: T. Li, Y. Xu, D. Ruan (Eds.), Proceedings of the International Conference on Intelligent Systems and Knowledge Engineering, 2007.
22. C. Haruechaiyasak, M.-L. Shyu, S.-C. Chen, X. Li, Web document classification based on fuzzy association, in: COMPSAC '02: Proceedings of the 26th International Computer Software and Applications Conference on Prolonging Software Life: Development and Redevelopment, IEEE Computer Society, Washington, DC, USA, 2002, pp. 487–492.
23. J. Lu, B. Xu, H. Yang, A classification method of fuzzy association rules, in: the Second IEEE International Workshop on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, 2003, pp. 248–251.
24. W.-H. Au, K. C. Chan, Classification with degree of membership: A fuzzy approach, Data Mining, IEEE International Conference on Data Mining (2001) 35.
25. K. Lang, Newsweeder: learning to filter netnews, in: Proceedings of the 12th International Conference on Machine Learning, Morgan Kaufmann publishers Inc.: San Mateo, CA, USA, 1995, pp. 331–339.
26. M. Claypool, A. Gokhale, T. Miranda, P. Murnikov, D. Netes, M. Sartin, Combining content-based and collaborative filters in an online newspaper (1999).
27. M. Balabanović, Y. Shoham, Fab: content-based, collaborative recommendation, Commun. ACM 40 (3) (1997) 66–72.
28. M. K. Condliff, D. D. Lewis, D. Madigan, C. Posse, Bayesian mixed-effects models for recommender systems (1999).
29. J. S. Breese, D. Heckerman, C. Kadie, Empirical analysis of predictive algorithms for collaborative filtering, 1998, pp. 43–52.
30. M. Chau, D. Zeng, H. Chen, M. Huang, D. Hendriawan, Design and evaluation of a multi-agent collaborative web mining system, Decision Support Systems 35 (1) (2003) 167–183.
31. M. N. Moreno, F. J. García, M.J. Polo, V. F. Lopez, Using Association Analysis of Web Data in Recommender Systems, Vol. 3182, Springer, 2004, Ch. E-Commerce and Web Technologies, pp. 11–20.
32. R. D. T. Júnior, Combining collaborative and content-based filtering to recommendresearch papers, Master's thesis, Universidade Gederal Do Rio Grande Do Sul (2004).
33. M. J. Pazzani, A framework for collaborative, content-based and demographic filtering, Artificial Intelligence Review 13 (5 - 6) (1999) 393–408.
34. P. S. Yu, Data mining and personalization technologies, in: DASFAA '99: Proceedings of the Sixth International Conference on Database Systems for Advanced Applications, IEEE Computer Society, Washington, DC, USA, 1999, pp. 6–13.
35. J. A. Konstan, J. Riedl, Research resources for recommender systems, in: CHI 99 Workshop Interacting with Recommender Systems, 1999.
36. B. M. Sarwar, J. A. Konstan, A. Borchers, J. L. Herlocker, B. Miller, J. Riedl, Using filtering agents to improve prediction quality in the grouplens research collaborative filtering system, in: Computer Supported Cooperative Work, 1998, pp. 345–354.
37. C. Avery, R. Zeckhauser, Recommender systems for evaluating computer messages, Commun. ACM 40 (3) (1997) 88–89.
38. J. B. Schafer, The application of data-mining to recommender systems, in: J. Wang

(Ed.), Encyclopedia of Data Warehousing and Mining, Information Science Publishing, 2005.
39. X. Sun, F. Kong, H. Chen, Using quantitative association rules in collaborative filtering, in: W. Fan, Z. Wu, J. Yang (Eds.), WAIM, Vol. 3739 of Lecture Notes in Computer Science, Springer, 2005, pp. 822–827.
40. X. Fu, J. Budzik, K. J. Hammond, Mining navigation history for recommendation, in: IUI '00: Proceedings of the 5th international conference on Intelligent user interfaces, ACM Press, New York, NY, USA, 2000, pp. 106–112.
41. W. Lin, S. Alvarez, C. Ruiz, Efficient adaptive-support association rule mining for recommender systems (2002).
42. T. Berka, M. Plößnig, Designing recommender systems for tourism, in: The 11th International Conference on Information Technology in Travel and Tourism, 2004.
43. R. R. Yager, Fuzzy logic methods in recommender systems, Fuzzy Sets Systems 136 (2) (2003) 133–149.
44. Y. Cao, Y. Li, X. Liao, Applying fuzzy logic to recommend consumer electronics, in: ICDCIT, 2005, pp. 278–289.
45. O. Nasraoui, H. Frigui, A. Joshi, R. Krishnapuram, Mining web access logs using relational competitive fuzzy clustering, in: In Proceedings of the Eight International Fuzzy Systems Association World Congress, 1999.
46. L. M. de Campos, J. M. Fernández-Luna, J. F. Huete, A collaborative recommender system based on probabilistic inference from fuzzy observations, Fuzzy Sets Systems 159 (12) (2008) 1554–1576.
47. R. J. Quinlan, C4.5: Programs for Machine Learning (Morgan Kaufmann Series in Machine Learning), Morgan Kaufmann, 1993.
48. D. Dubois, H. Prade, A note on quality measures for fuzzy association rules, in: Proceedings IFSA-03, 10th International Fuzzy Systems Association World Congress, number 2715 in Lecture Notes in Artificial Intelligence, Springer-Verlag, 2003, p. 677.
49. C. Fiot, A. Laurent, M. Teisseire, From crispness to fuzziness: Three algorithms for soft sequential pattern mining., IEEE T. Fuzzy Systems 15 (6) (2007) 1263–1277.
50. C.-N. Ziegler, S. M. McNee, J. A. Konstan, G. Lausen, Improving recommendation lists through topic diversification, in: 14th International World Wide Web Conference, 2005.
51. I. H. Witten, E. Frank, Data Mining: Practical machine learning tools and techniques, 2nd Edition, 2005.
52. J. C. Hühn, E. Hüllermeier, Furia: an algorithm for unordered fuzzy rule induction., Data Mining and Knowledge Discovery 19 (3) (2009) 293–319.
53. J. L. Herlocker, J. A. Konstan, L. G. Terveen, J. T. Riedl, Evaluating collaborative filtering recommender systems, ACM Trans. Information Systems 22 (1) (2004) 5–53.
54. M. Rittman, "what is sparsity, and why should i be concerned with it?", Oracle news (2005).
55. S. García, A. Fernández, J. Luengo, F. Herrera, A study of statistical techniques and performance measures for genetics-based machine learning: accuracy and interpretability, Soft Comput. 13 (10) (2009) 959–977.
56. S. García, F. Herrera, An extension on "statistical comparisons of classifiers over multiple data sets" for all pairwise comparisons, Journal of Machine Learning Research 9 (2009) 2677–2694.
57. M. Friedman, The use of ranks to avoid the assumption of normality implicit in the analysis of variance, Journal of the American Statistical Association 32 (1937) 675–701.
58. J. Demšar, Statistical comparisons of classifiers over multiple data sets, The Journal

of Machine Learning Research 7 (2006) 1–30.
59. J. P. Shaffer, Modified sequentially rejective multiple test procedures, Journal of the American Statistical Association 81 (1986) 826–831.
60. T. Fawcett, Roc graphs: Notes and practical considerations for data mining researchers, Technical Report HPL-2003-4, HP Laboratories (2003).
61. Y. Hochberg, A sharper bonferroni procedure for multiple tests of significance, Biometrika 4 (1988) 800–802.