Expert Systems With Applications 58 (2016) 1–9

Contents lists available at ScienceDirect

Expert Systems With Applications

journal homepage: www.elsevier.com/locate/eswa

Improving the performance of inductive learning classifiers through the presentation order of the training patterns

Gonzalo A. Ruz*

[a] Facultad de Ingeniería y Ciencias, Universidad Adolfo Ibáñez, Av. Diagonal Las Torres 2640, Peñalolén, Santiago, Chile
[b] Center of Applied Ecology and Sustainability (CAPES), Santiago, Chile

ARTICLE INFO

Article history:
Received 28 February 2015
Revised 1 April 2016
Accepted 2 April 2016
Available online 5 April 2016

Keywords:
Inductive learning
Rules family
Clustering
Classification

ABSTRACT

Although the development of new supervised learning algorithms for machine learning techniques are mostly oriented to improve the predictive power or classification accuracy, the capacity to understand how the classification process is carried out is of great interest for many applications in business and industry. Inductive learning algorithms, like the Rules family, induce semantically interpretable classification rules in the form of if-then rules. Although the effectiveness of the Rules family has been studied thoroughly and new and improved versions are constantly been developed, one important drawback is the effect of the presentation order of the training patterns which has not been studied in depth previously. In this paper this issue is addressed, first by studying empirically the effect of random presentation orders in the number of rules and the generalization power of the resulting classifier. Then a presentation order method for the training examples is proposed which combines a clustering stage with a new density measure developed specifically for this problem. The results using benchmark datasets and a real application of wood defect classification show the effectiveness of the proposed method. Also, since the presentation order method is employed as a preprocessing stage, the simplicity of the Rules family is not affected but instead it enables the generation of fewer and more accurate rules, which can have a direct impact in the performance and usefulness of the Rules family in an expert system context.

1. Introduction

Pattern classification consists in assigning (automatically) a class label to a vector of attributes. These attributes, hopefully, have sufficient discriminatory power in order for the classifier to correctly assign each pattern to its appropriate class. The construction or learning of a classifier from data, consists in a *training* phase, where a training set of attribute vectors are used to adjust the parameters of the classifier, for example in a neural network, it would be the weights and the biases, or in a naive Bayes classifier, the distribution of each class and class conditional probabilities of each attribute. Then, the performance of a trained classifier is measured during a *testing* phase, using what is called a test set, which contains attribute vectors that were not used during the training. Throughout the years we have seen new or improved classifiers coming from the machine learning community that are more powerful, in the sense of accuracy, such as support vector machines (SVM) and its variants (Cortes & Vapnik, 1995) and random forests (RF) (Breiman, 2001). In general, one can find many applications of classifiers in different domains, for example, SVM have been used for mechanical faults diagnostic in induction motors (Baccarini, Rocha e Silva, Rodrigues de Menezes, & Caminhas, 2011), neural networks for fraud detection in medical claims (Ortega, Figueroa, & Ruz, 2006), Bayesian network classifiers in the recognition of control chart patterns (Ruz & Pham, 2009) and policy making for broadband adoption (Ruz, Varas, & Villena, 2013), a naive Bayes classifier is used to predict job performance in a call center (Valle, Varas, & Ruz, 2012), and customer churn prediction using random forests (Xie, Li, Ngai, & Ying, 2009).

While research in some areas of machine learning is devoted to generate more powerful classifiers, there is a trade-off between the quantitative aspect of the classifier, correct classification or accuracy, and the qualitative aspect of the classifier, i.e., understand how the classification process is carried out. Most of the more powerful classifiers are considered as black-box models, where although achieving high quantitative performances, they tend to have a rather low qualitative aspect of the model.

For real world expert systems and data mining applications, in different areas such as retail, banking, finance, health, etc. many of these machine learning techniques are still considered as *black*

* Corresponding author. Tel.: +56223311200.
E-mail address: gonzalo.ruz@uai.cl

http://dx.doi.org/10.1016/j.eswa.2016.04.003
0957-4174/

magic, in the sense that they are not able to explain or give explicit classification rules. Therefore, this skepticism for these techniques can in many cases, even after a successful prototype or trial period, never consolidate in a real working application in the company. One approach to overcome this problem, is to work with more simple classifiers, known as white-box models, where the classification process is made explicit through rules, such as IF-THEN rules or decision trees. In this case, quantitative performance is usually reduced, but the qualitative aspect of the model is enhanced. Therefore, in many applications where one can compromise the quantitative performance of the model in order to gain more qualitative characteristics of how the classification is conducted, are in much need in real industry applications. Of course one could always approach the problem by training a black-box model for high accuracy and a white-box model to use to explain in a more broader sense, in a board meeting for example, how the classification is carried out.

Amongst the many white-box classifiers, in this paper, we revisit *inductive learning*, or the generation of IF-THEN rules for automatic classification purposes. There have been a number of inductive learning programs developed throughout the years, like the well-known programs ID3 (Quinland, 1983), which is a divide-and-conquer program, and the AQ program (Michalski, 1990). For this paper, the simple inductive learning algorithms belonging to the RULES (RULe Extraction System) family developed by Pham and Aksoy (1993); 1995a); 1995b) are of interest.

A common drawback that the Rules family algorithms have is that the generalization power of the rules generated during the training process is affected by the presentation order of the patterns used in this process. Random presentation orders of the training data can yield to different results, which may not be always efficient and accurate. This issue arises from the fact that the Rules family during its rule induction process considers the class of a selected seed example as the target class. Then it tries to induce rules that cover as many examples of the target class as possible through a rule evaluation function. No formal criterion is given on efficient ways to select the seed example.

The problem of the presentation order for the Rules family was first considered in Pham, Dimov, and Salem (2000), where a clustering approach was used to generate different presentation orders with the aim of reducing the RULES-4 execution time and the generation of fewer rules. Later in Pham and Salem (2004), a reorganization of the training data is proposed that essentially consists in introducing an example of each different class in the sequence of the training examples, this way the algorithm would learn from examples of at least two classes from the beginning of the training. This would improve the performance of the Rules algorithm by avoiding the generation of one default rule from the initial examples in the training data.

Apart from these two works, there have been no other recent efforts to analyze and propose new methods as a preprocessing stage to minimize the effects of the presentation order of the training patterns. In fact, more recent research concerned with the Rules family algorithms have been concentrated in handling continuous class labels (ElGibreen & Aksoy, 2015a). For example in ElGibreen and Aksoy (2015b), a new version of the Rules family y proposed called RULES-3C which incorporates properties from reinforcement learning to handle continuous classes. Another extension to the Rules family is RULES-IT (ElGibreen & Aksoy, 2014b), which is an incremental covering algorithm that integrates transfer learning to enable the use of past experiences from different domains. RULES-IT has proven to be effective for handling incomplete data and missing labels (ElGibreen & Aksoy, 2014a). It is important to point out that none of these new versions of the Rules family address the issue of the effect of the presentation order of the training examples in their results, which is the main contribution of this paper.

Another preprocessing approach alternative to changing the presentation order of the training examples is to apply instance reduction before carrying out the rule induction as in Othman and Bryant (2015). It was found that by applying instance reduction methods fewer rules were generated without compromising the classification performance.

Fuzzy min-max neural networks (Simpson, 1992) also can be affected by the presentation order. FMMN grow decision boundaries called hyperboxes based on the presentation of the training examples one by one, thus, the resulting hyperboxes and the predictive power may vary depending on the sequence of the training data. To overcome this problem two training algorithms have been proposed for this type of neural networks (Rizzi, Panella, & Mascioli, 2002). Nowadays, new versions have focused in handle mixed attributes (continuous and discrete) such as in Shinde and Kulkarni (2016) or a modified version for clustering (Seera, Lim, Loo, & Singh, 2015).

More recent techniques have been developed for fuzzy ART-type networks such as fuzzy ARTMAP (FAM) (Pourpanah, Lim, & Saleh, 2016), that also suffer from data presentation order issue (Carpenter, Grossberg, Markuzon, Reynolds, & Rosen, 1992). One alternative to overcome this problem has been the use of genetic algorithm. Like in Loo, Liew, Seera, and Lim (2015), where the presentation order is coded from 1 to N, where N is the number of instances in the training set. Then the training sequence, together with some other parameters to be optimized were coded in a chromosome. The fitness function corresponds to the average classification accuracy of the test set, within a ten-fold cross validation scheme. Overall the results using five benchmark data sets showed the effectiveness of the proposed approach. Other examples using a genetic algorithm for the selection of the training pattern order can be found in Baek, Lee, Lee, Lee, and Kim (2014); Palaniappan and Eswaran (2009). In the case of a FAM ensemble, in Oong and Isa (2014), a data presentation method is proposed based on the ascending order of the value from the most uncorrelated input features. The results using benchmark data sets showed that the proposed ordering algorithm obtained better generalization performance in seven out of the eleven data sets.

It is desirable to overcome this presentation order problem without compromising the simplicity of the Rules family algorithms, therefore, in this work, a preprocessing stage is proposed to reduce the variability of the generalization of the Rules family algorithms. In particular, the contributions of this work are

- The quantification in terms of the number of rules and generalization power of the Rules family when random order presentation of the training data is used.
- A new density measure that uses a fuzzy membership function to select representative seed examples from each class.
- A presentation order method of the training examples that combines clustering techniques with the proposed density measure.
- The evaluation of the proposed approach not only on benchmark data sets but also on a real application of wood defect classification.

The outline of this paper is as follows. A brief description of the Rules family and the clustering techniques employed by the proposed presentation order of the training patterns are specified in Section 2, the density measure used to rank the data is developed in Section 3. Section 4 introduces the presentation order of the training patterns method, whereas Section 5 describes the data sets used to test the proposed technique and how the simulations were conducted. Results and discussions are carried out in

Section 6. Finally, the conclusions and future work are outlined in Section 7.

2. Background

In this section, a brief description is given of the Rules family and the clustering techniques that will be used for generating the presentation order of the training patterns.

2.1. Rules family algorithms

The RULES (RULe Extraction System) algorithm was developed by Pham and Aksoy (1995b). The pseudo code can be summarized as follows:

1. Initialize the rule set to empty
2. While all the examples in the training set are not marked as covered
   (a) Select an example E that has not yet been marked
   (b) *Induce a rule* to cover E and a small portion of negative examples
3. Return rule set

The procedure induce a rule generates one rule at a time through a seed example and the application of a specialization process to find the best rule.

This simple inductive learning procedure has been used as a base for generating better and improved versions of this algorithm. Nowadays, the Rules family expands more than ten versions or upgrades from this original version. The reader can refer to a recent review of the Rules family algorithms and its applications in Almana and Aksoy (2014). For this paper, we will consider RULES 3 Plus (Pham & Dimov, 1997), RULES 5 (Pham, Bigot, & Dimov, 2003), and RULES 5 with pruning (Pham, Bigot, & Dimov, 2004).

2.2. Maximin-distance algorithm (MMD)

The Maximin-Distance algorithm (Tou & Gonzalez, 1974) is an heuristic for data clustering, it can be introduced as follows. Let us consider the data set defined by $A = \{x_1, \ldots, x_N\}$ and let us assume that at least two clusters are expected to exist. We set a threshold value τ that at each step determines whether a new cluster should be created. Let $c_1, \ldots, c_m$ be the existing cluster centers (they are the points in A). Let us denote the mean of the distances between the centers by μ and let x_i^{new} be the data point most likely to be chosen as the center of a new cluster. The center of a new cluster is determined by the following procedure. For each remaining point $x_i \in A - \{c_1, \ldots, c_m\}$, compute its distance to each existing center of $\{c_1, \ldots, c_m\}$ and save the shortest distance for each x_i. Then assign x_i^{new} as the point x_i, which obtained the maximum value of the shortest (minimum) distance. This can be represented as

$$x_i^{new} = \arg\max_{x_i} \min_{1 \le j \le m} dist(x_i - c_j). \tag{1}$$

If the value

$$\theta = \min_{1 \le j \le m} dist(x_i^{new}, \mu) \tag{2}$$

is less than $\tau\mu$, then no new cluster center is created, and this part of the process terminates. Otherwise $c_{m+1} = x_i^{new}$ and continue. After all the cluster centers have been found each remaining sample is assigned to its nearest cluster center. Finally, the cluster centers are adjusted so that each centre is the mean of the cluster's samples. The complete MMD algorithm can be summarized as follows:

1. Randomly select a point, say x_1 as the first center c_1, and then calculate the distances between all the remaining x_i and the first center c_1. Suppose that the point x_i^{new} has the largest distance. Then choose x_i^{new} as the second center c_2.
2. Set $m = 2$, $\mu = dist(c_1, c_2)$ (the mean of the centers).
3. Find the most likely new center x_i^{new} from the remaining examples by the Max-Min criterion (1) and calculate the maximal distance by (2).
4. If the maximal distance $\theta \le \tau\mu$, then go to step 6.
5. Increase m by 1, and $c_{m+1} = x_i^{new}$; and calculate μ the mean of the distances between all the centers $c_1, \ldots, c_{m+1}$. Then go to step 2.
6. For each example in A, assign it to the nearest cluster.
7. Calculate the mean of each cluster and replace c_j by this mean. Stop!

2.3. k-means

The k-means clustering algorithm (Tou & Gonzalez, 1974) is a method for finding k vectors μ_j $(j = 1, 2, \ldots, k)$ that represent an entire dataset. The data is considered to be partitioned into k clusters, with each cluster represented by its mean vector and each data instance assigned to the cluster with the closest vector. The k-means algorithm works iteratively. At each stage, the N data examples $A = \{x_1, \ldots, x_N\}$ are partitioned into k disjoint clusters A_j each containing N_j instances. A cost function (or an objective function) of dissimilarity (or distance) is defined as

$$J = \sum_{j=1}^{k} J_j = \sum_{j=1}^{k} \sum_{x_i \in A_j} \|x_i - \mu_j\|^2, \tag{3}$$

where μ_j is the center of the jth cluster, given by the mean of the data instances belonging to that cluster

$$\mu_j = \frac{1}{N_j} \sum_{x_i \in A_j} x_i. \tag{4}$$

The initial partition of the data is random. Then the following two steps are iterated until there is no further change to the cost function J.

1. The mean vectors μ_j for each cluster are calculated using (4).
2. Rearrange the clusters: each data instance x_i is assigned to the jth new cluster if x_i is closer to μ_j than to the other mean vectors.

3. Density measure

The presentation order of the training patterns described in the next section, starts of by clustering the data using one of the techniques described in the previous section. Once the data is clustered, it is desirable to rank the data from each cluster from the most dense to the least dense. This way the Rules algorithm will be using more representative data in its training process, which will yield in more accurate and compact rule sets. In order to rank the data a density measure is required. In this work, the following expression is proposed

$$D_j = \sum_{i}^{M} d_j(x_i) \tag{5}$$

where D_j is the jth pattern density value, x_i $(i = 1, \ldots, M)$ is the ith pattern belonging to the same cluster as x_j, and $d_j(x_i)$ is a distance function (dissimilarity) between x_j and x_i. When the Euclidean distance is chosen as the dissimilarity measure, the densest pattern for a given cluster will be the pattern that obtains the lowest value for D.

Due to the fact that the Rules family algorithms generate if-then rules to classify data, the partitioning of the input data space is done by hyperplanes which are parallel to the main reference system. In many cases these hyperplanes intersect to form compact

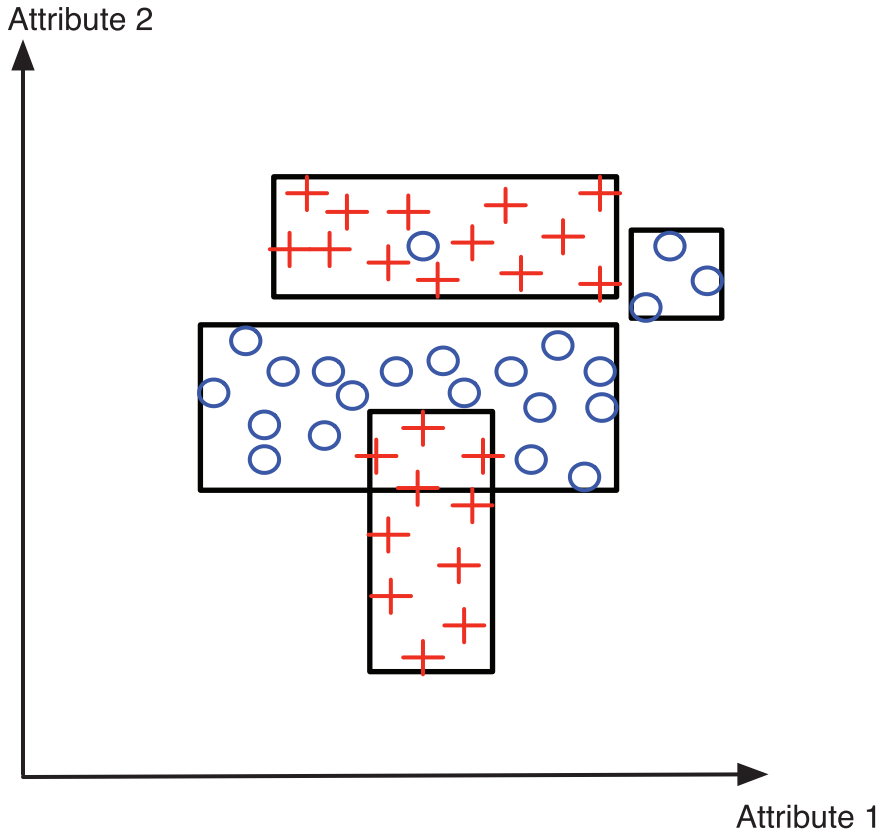

Fig. 1. A 2D example of the rule sets formed by RULES 5.

hyperrectangles rule sets, as shown in Fig. 1 for a 2D example using RULES 5 in a 2 class (o and +) classification problem. In this example you can see that there are two rule sets (hyperrectangles) for class + and two rule sets for class o. Also you can appreciate that a certain level of noise is permitted.

So taking into account that the rules cover the patterns with hyperrectangles, it is desirable to measure the density level of a pattern amongst the other patterns in the same cluster following a rectangular distribution as well. This is why the Euclidean distance is not used as the dissimilarity function in this case, since the Euclidean distance describes a circular or globular boundary and that is not how the patterns are classified by the Rules family algorithms.

To be able to measure the similarity between patterns, a fuzzy membership function is proposed. This membership function is a special case of the Fuzzy Hyperbox membership function used in the General Fuzzy Min-Max (GFMM) Neural Network for clustering and classification (Gabrys & Bargiela, 2000). The Fuzzy Hyperbox membership function has also been used for image processing in the Fuzzy Min-Max neural network for Image Segmentation (FMMIS) (Ruz, Estévez, & Perez, 2005; Ruz, Estévez, & Ramírez, 2009).

Following Simpson (1993), let the jth hyperbox fuzzy set B_j be defined by the ordered set

$$B_j = \left\{A_h, V_j, W_j, b_j(A_h, V_j, W_j)\right\} \tag{6}$$

for all $h = 1, \ldots, m$, where $A_h = (a_{h1}, a_{h2}, \ldots, a_{hn}) \in I^n$ (n-dimensional unit cube) is the hth pattern in the data set, $V_j = (v_{j1}, v_{j2}, \ldots, v_{jn})$ is the min point for the jth hyperbox, $W_j = (w_{j1}, w_{j2}, \ldots, w_{jn})$ is the max point for the jth hyperbox, and the membership function for the jth hyperbox is $0 \leq b_j(A_h, V_j, W_j) \leq 1$.

The membership function measures the degree to which the hth input pattern A_h falls within the hyperbox formed by the min point V_j and the max point W_j. The membership function b_j is defined as (Gabrys & Bargiela, 2000):

$$b_j(A_h, V_j, W_j) = \min_{i=1,\ldots,n}\Big(\min\Big([1 - f(a_{hi} - w_{ji}, \gamma)], \times[1 - f(v_{ji} - a_{hi}, \gamma)]\Big)\Big), \tag{7}$$

where f is a two-parameter ramp threshold function

$$f(x, \gamma) = \begin{cases} 1, & \text{if } x\gamma > 1 \\ x\gamma, & \text{if } 0 \leq x\gamma \leq 1 \\ 0, & \text{if } x\gamma < 0 \end{cases} \tag{8}$$

The parameter γ is a sensitivity parameter that regulates how fast the membership values decrease when an input pattern is separated from the hyperbox core. When γ is large, the fuzzy set becomes more crisp, and when γ is small the fuzzy set becomes less crisp. Throughout the simulations, $\gamma = 1$ was used.

3.1. Proposed fuzzy membership function

If we consider $V_j = W_j = A_j$ where A_j is the jth input pattern, then (7) can be rewritten as

$$b_j(A_h, A_j) = \min_{i=1,\ldots,n}\Big(1 - f(|a_{hi} - a_{ji}|, \gamma)\Big). \tag{9}$$

So now we can define $d_j = b_j$ in (5) as a membership function measuring the degree of similarity between two patterns, thus the densest pattern for a given cluster will be the pattern that obtains the highest value for D.

A two-dimensional example is shown in Fig. 2, where it is clear that the membership values decrease steadily with increasing distance from the pattern been analyzed ($A_h = [0.5, 0.5]$).

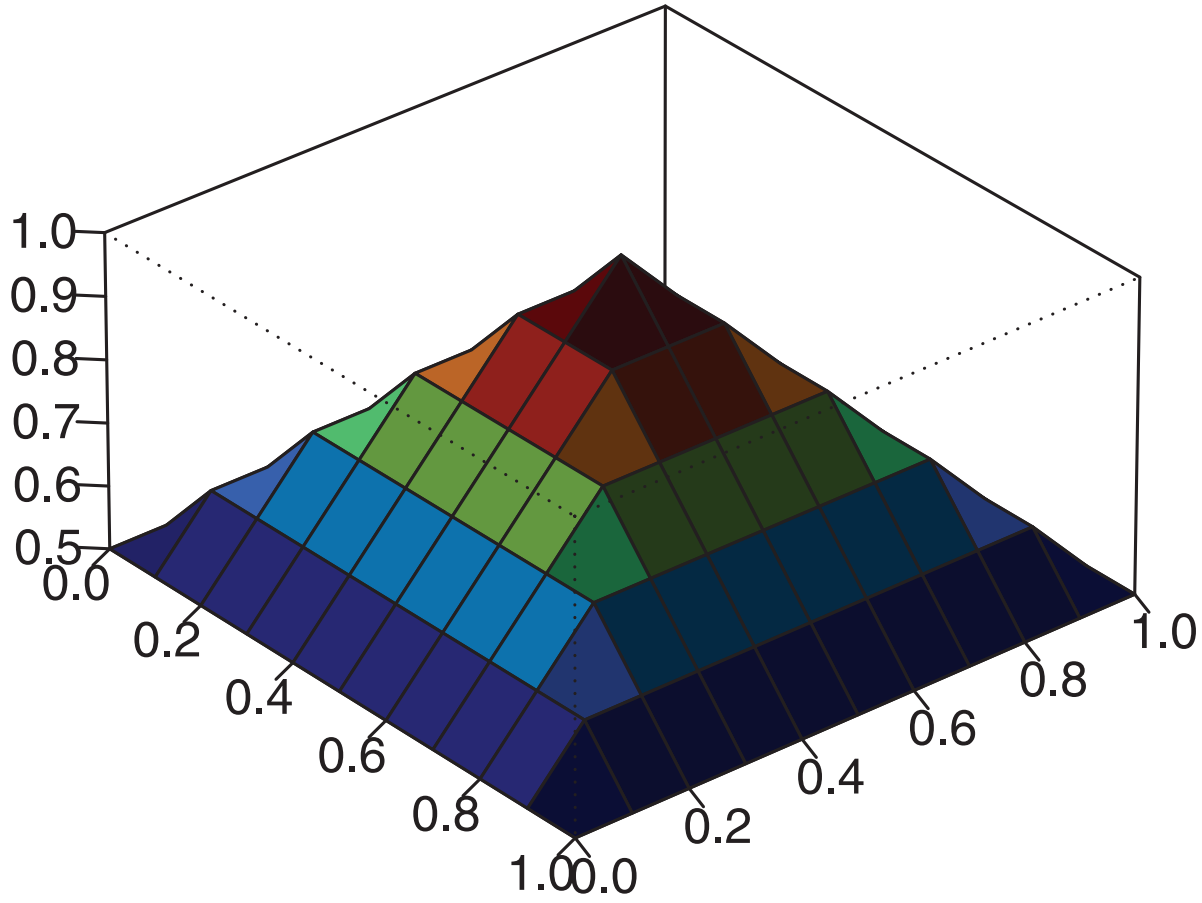

Fig. 2. 2D example of membership function used in the density function.

4. Presentation order of the training patterns

The presentation order (PO) of the training patterns proposed in this work is shown in Fig. 3. The first stage is to form clusters with the training data set using the MMD algorithm or the *k*-means. Initially the number of clusters is fixed to the number of classes for a given data set, this is obtained by setting an appropriate value of τ in the MMD or the value of k in the *k*-means algorithm. But as we will see further on, it is recommended to explore a higher number of clusters than the number of classes. Once, the clusters are formed it is desirable to rank each data from the most dense to the less dense from each cluster. To accomplish this, the density measure in (5) is used with the proposed fuzzy membership function in (9). The following stage is the data selection which in this case is simply selecting one data form each cluster (considering its ranking). Finally, an ordered data set is obtained and used for the training process in one of the Rules family algorithm. To measure the Rules family generalization power, a separate data set is used called the test set which has not been used in the training process to measure the algorithm's classification accuracy.

The basic idea behind the PO is that a possible cause of the significant variation in the generalization performance for different orders of training pattern presentations for covering inductive learning algorithms, for example the Rules family, is giving importance to outlier data and/or data with only a few number of neighbors in the patterns input space. This happens when this type of data is presented at the beginning of the rule forming process instead of the end, causing the inductive algorithm to produce less compact and accurate rule sets.

Table 1
Description of data sets used to test the proposed presentation order (PO) method.

Data sets	Attributes	Classes	Training set	Test set	Total
Breast cancer	30	2	313	256	569
Haberman	3	2	169	137	306
Ionosphere	34	2	193	158	351
Iris	4	3	80	70	150
Wine	13	3	98	80	178
Glass	9	6	118	96	214
Liver	6	2	190	155	345
Zoo	16	7	56	45	101
Pima	8	2	422	346	768
Tae	5	3	84	67	151
Wood	17	13	127	105	232

5. Simulations setup

The proposed PO was tested using 10 benchmark data sets (breast cancer, haberman, ionosphere, iris, wine, glass, liver, zoo, pima indians diabetes, and tae) from the UCI Machine Learning Repository (Lichman, 2013) and one data set (wood) belonging to a real world problem corresponding to feature values extracted from grey-level images of wood veneer for the task of defect classification (Packianather & Drake, 2000). A list with these data sets describing the number of attributes, classes and partitioning of the training and test sets are shown in Table 1. To measure how sensible the rule generation process (training stage) is to different orders of pattern presentations, 30 random ordered presentation of the training data sets were used in the Rules family algorithms and

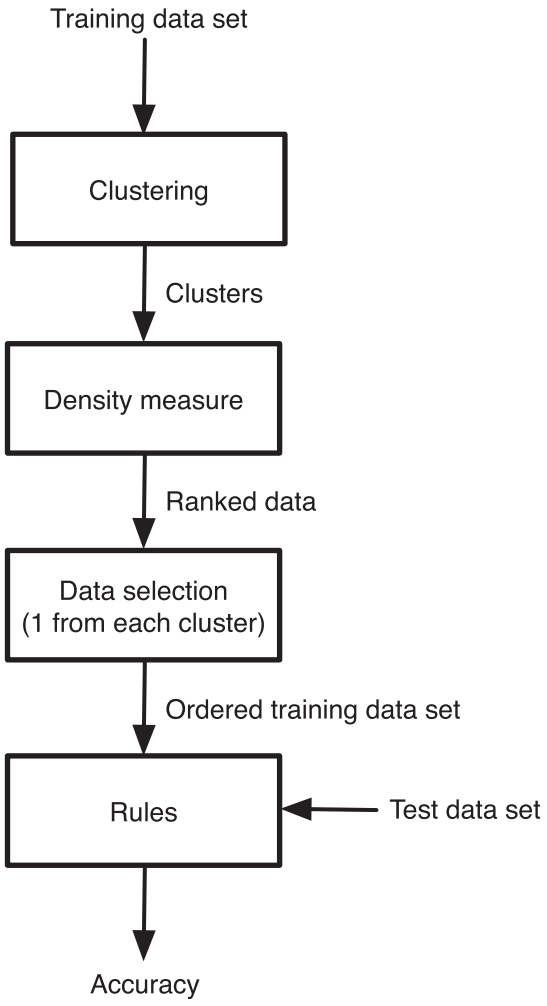

Fig. 3. Flow chart of the method proposed to improve the presentation order of the training patterns.

compared to the performance obtained by the PO. The Rules family algorithms that were used in these simulations were RULES 3 Plus and RULES 5 (with and without pruning).

6. Results and discussions

The results of the simulations carried out on the data sets are shown in Tables 2 and 3 for the RULES 3 Plus algorithm, Tables 4 and 5 for the RULES 5 algorithm, and Tables 6 and 7 for the RULES 5 with pruning algorithm. For the RULES 3 Plus, Table 2 shows on the left side the worst classification accuracy, the best classification accuracy, and the average classification accuracy using the test set for 30 random presentation orders of the training data sets. On the right side of the table, we can see the classification accuracy obtained when the PO was used on the training patterns, results using PO with MMD (PO_{MMD}) clustering and k-means ($PO_{k-means}$) clustering as well as the number of clusters used by each clustering technique. In Table 3, you can analyze the minimum, the maximum, and the most frequent number of rules that were obtained by the 30 random presentation orders of the training sets as well as the number of rules generated by ordered data using the PO.

Table 2 shows that for RULES 3 Plus the performance obtained by the PO_{MMD} and $PO_{k-means}$ is higher than the average value of the classification accuracy of 30 random presentation orders of the training set. In particular, we can see that for the haberman, ionosphere, iris, wine, liver, zoo, and tae data sets either both or one of the PO obtained the same value as the best case found amongst the random presentation orders.

Analyzing Table 4, we notice that once again the PO_{MMD} and $PO_{k-means}$, in this case for the RULES 5, obtained a higher performance (equal in the zoo and the iris data set which did not present any variations) than the average value of the 30 random presentation orders of the training sets. It is also interesting to notice that in the case of the $PO_{k-means}$ with the breast cancer, ionosphere, wine, and the pima indians data set and for the PO_{MMD} with the pima indians data set they obtained a performance higher than the best case found by the 30 random presentation orders.

Table 6 shows the results for RULES 5 with pruning. Here, we see again that the PO_{MMD} and $PO_{k-means}$ have a higher (equal in the zoo data set which did not present any variations) performance than the average of the random presentation orders. The PO_{MMD} with breast cancer data set and the $PO_{k-means}$ with the wine data set obtained a higher classification percentage than the best case found amongst the random presentation orders. Also, for three cases (iris, liver, and zoo) the results are equal to the best case found.

From Tables 2, 4, and 6 we see that the number of clusters used by the PO is higher than the number of classes for each data set analyzed. The reason for this, is that it was found that by considering a higher amount of clusters than the number of classes in each case analyzed better results were obtained.

Not only in the classification accuracy does the random presentation of the training pattern generate significant variations in the results but also if we look at Tables 3, 5, and 7, it is clear that the number of rules is strongly affected as well.

In general, by studying all the tables it was found that the random presentation order of the training patterns generates a variation in the classification percentage in the test set of approximately 7% and approximately 20% in the number of rules formed.

7. Conclusion

The Rules family rule induction procedure generates one rule at a time through a seed example. If this seed example is an outlier or not a good representative of its class, then the inferred rules will not be compact, and more rules would be needed to cover the training examples, affecting also the generalization power of the resulting classifier. Therefore, the presentation order of the training examples is vital for these type of algorithms.

The search for the optimum presentation order is a combinatorial problem. Indeed, for N data points in the training set the number of different presentation orders of the training set corresponds to $N!$. Therefore, even for a rather small training set, for example $N = 100$, the number of different presentation orders to test would be about 9.3×10^{157}, which is intractable. Thus, a brute force approach is not feasible. One way to approach this problem is to employ a more intelligent search approach by evolutionary computation like in Baek et al. (2014); Loo et al. (2015); Palaniappan and Eswaran (2009). A drawback with this approach is that the convergence to a good result could be quite time consuming especially when the training set contains thousands of examples, remember that under an evolutionary computation approach the typical coding is that each element of the chromosome represents the position in the training sequence of the training set.

In this paper, we present another approach through clustering. By clustering the training examples, we are able to find the natural groupings of the data, and which data points are more representative than others from each cluster. This is accomplished through a density measure which is computed through a proposed fuzzy membership function that considers the fact that the rules cover the training data with hyperrectangles.

Table 2
Classification performance in the RULES 3 Plus with random presentation v.s. RULES 3 Plus with PO method.

Data Sets	Worst case %	Best case %	Average %	# Clusters	PO_{MMD} %	k	$PO_{k-means}$ %
Breast cancer	93.75	95.31	94.66	3	94.92	3	94.92
Haberman	48.91	75.18	63.33	4	75.18	3	75.18
Ionosphere	87.97	91.14	89.75	6	91.14	3	89.88
Iris	94.29	95.71	95.57	6	95.71	4	95.71
Wine	91.25	96.25	95.00	3	96.25	3	95.75
Glass	56.25	59.38	57.92	10	58.33	12	60.42
Liver	58.06	60.65	59.23	4	60.65	7	60.00
Zoo	86.67	86.67	86.67	7	86.67	7	86.67
Pima	71.68	73.41	72.60	6	72.83	7	73.12
Tae	46.27	47.76	46.87	3	47.76	3	47.76
Wood	50.48	54.29	53.24	13	53.33	15	53.33

Table 3
Number of rules formed by the RULES 3 Plus with random presentation v.s. RULES 3 Plus with PO method.

Data sets	min-rules	max-rules	mode	PO_{MMD}-rules	$PO_{k-means}$-rules
Breast cancer	36	44	43	38	43
Haberman	43	43	43	43	43
Ionosphere	41	48	45	39	41
Iris	11	13	12	11	11
Wine	22	27	24	23	25
Glass	51	54	53	55	52
Liver	80	84	84	85	80
Zoo	7	9	8	7	8
Pima	183	193	186	190	185
Tae	33	34	33	33	33
Wood	66	73	73	73	70

Table 4
Classification performance in the RULES 5 with random presentation v.s. RULES 5 with PO method.

Data sets	Worst case %	Best case %	Average %	# Clusters	PO_{MMD} %	k	$PO_{k-means}$ %
Breast cancer	92.58	96.48	94.60	6	96.06	6	97.27
Haberman	58.39	75.18	71.82	4	74.45	4	74.45
Ionosphere	89.87	93.67	91.65	6	93.67	5	94.30
Iris	95.71	95.71	95.71	6	95.71	3	95.71
Wine	87.45	97.50	92.54	11	97.50	9	98.75
Glass	38.54	61.46	46.98	12	47.82	11	50.00
Liver	63.87	72.26	67.10	5	71.61	4	70.32
Zoo	88.89	88.89	88.89	7	88.89	7	88.89
Pima	73.12	77.17	74.94	5	78.61	4	77.75
Tae	38.80	47.76	43.43	4	47.76	3	46.27
Wood	48.57	62.86	57.71	13	60.95	16	58.10

Table 5
Number of rules formed by RULES 5 with random presentation v.s. RULES 5 with PO method.

Data sets	min-rules	max-rules	mode	PO_{MMD}-rules	$PO_{k-means}$-rules
Breast cancer	12	20	19	18	17
Haberman	45	58	50, 51	47	50
Ionosphere	17	23	18, 20	16	18
Iris	6	10	10	8	7
Wine	7	14	10, 11	13	9
Glass	29	35	33	30	35
Liver	40	55	48	48	46
Zoo	7	9	8	8	9
Pima	87	102	98	87	86
Tae	24	34	24	25	23
Wood	41	50	46	54	49

The main contributions of this work are: (i) a quantitative evaluation of the effect of random presentation orders which none of the previous Rules family papers have studied; (ii) a method to select good seed examples based on a density measure which employs a fuzzy membership function which considers a rectangular distribution in the same way as the training examples are covered by the rules; (iii) a presentation order method which combines clustering of the training examples with the density measure described previously; (iv) the evaluation of the proposed method not only on benchmark datasets but also on a real application.

The presentation order method was tested using the RULES 3 Plus, RULES 5, and the RULES 5 with pruning algorithms from the Rules family in 11 datasets. The results showed that the classification accuracy was superior than the average classification accuracy

Table 6
Classification performance in the RULES 5 (pruning) with random presentation v.s. RULES 5 (pruning) with PO method.

Data sets	Worst case %	Best case %	Average %	# Clusters	PO_{MMD} %	k	$PO_{k-means}$ %
Breast cancer	91.80	96.09	94.13	6	96.48	4	94.53
Haberman	60.58	74.45	71.33	5	73.72	3	73.72
Ionosphere	87.34	94.94	92.03	9	93.04	5	94.30
Iris	94.29	95.71	95.33	6	95.71	5	95.71
Wine	86.25	97.50	92.54	11	96.25	9	98.75
Glass	34.38	56.25	45.10	8	48.96	12	55.21
Liver	64.52	72.26	67.48	5	72.26	7	70.32
Zoo	88.89	88.89	88.89	7	88.89	7	88.89
Pima	74.57	79.48	76.76	5	78.32	4	79.19
Tae	40.30	47.76	42.84	4	46.27	3	46.27
Wood	49.52	62.86	58.10	13	60.00	16	60.00

Table 7
Number of rules formed by RULES 5 (pruning) with random presentation v.s. RULES 5 (pruning) with PO method.

Data sets	min-rules	max-rules	mode	PO_{MMD}-rules	$PO_{k-means}$-rules
Breast cancer	3	14	7	9	10
Haberman	24	37	32	24	28
Ionosphere	8	18	9	17	11
Iris	3	6	5	4	4
Wine	6	10	7	10	7
Glass	24	32	28	30	30
Liver	38	47	43	38	40
Zoo	7	8	8	8	8
Pima	52	61	55	54	61
Tae	22	24	24	23	22
Wood	38	43	43	41	39

obtained when 30 random presentation orders of the training patterns were considered. In many cases, when the proposed method was used, the resulting classification accuracy equalled the best classification accuracy found within the 30 random presentation orders and in some cases it outperformed the best case.

None of the most recent research related to the Rules family have addressed the presentation order of the training examples. The proposed method works as a preprocessing stage therefore it is independent of any of the Rules family algorithm, and therefore can be easily integrated to any of them. The effect of using this preprocessing stage has direct consequences for real expert systems applications in the sense of reducing the variability in the number of rules and generalization power, thus obtaining better and more reliable classification rules.

From a managerial point of view, the capacity to reduce the variability of the performance of an expert systems, implies less errors, which in return could imply less costs, due to the reduction of false positives, in applications such as defect detections in a production line for example (Ruz et al., 2005). Through the simulations carried out in this paper, it was found that the presentation order of the training patterns can generate a variation in the classification accuracy of about 7%. The capacity to control or reduce this level of variation could have direct economical implications.

Some aspects to improve, in future research, which can effect the performance of the PO is how to find the optimal number of clusters used by the clustering stage of the PO as well as exploring other data selection methods once the data is ranked in each cluster. Recall that for this problem, the optimal number of clusters is not necessarily the most plausible natural grouping or theoretically correct number of clusters through the use of a cluster validity index, the idea is to find the number of clusters that yield a good selection of seed examples through the proposed density measure.

Also, the application of this method in other classification systems which are sensitive to the presentation order of the training patterns like the fuzzy min-max neural network or fuzzy ARTMAP could also be considered in the future.

Acknowledgements

The author would like to thank Samuel Bigot at the School of Engineering, Cardiff University, for his comments and Rules family software. Also this work was supported partially by the Research Center Millennium Nucleus Models of Crisis (NS130017).

References

Almana, A. M., & Aksoy, M. (2014). An overview of inductive learning algorithms. *International Journal of Computer Applications, 88*, 20–28.
Baccarini, L. M. R., Rocha e Silva, V. V., Rodrigues de Menezes, B., & Caminhas, W. M. (2011). SVM practical industrial application for mechanical faults diagnostic. *Expert Systems with Applications, 38*, 6980–6984.
Baek, J., Lee, H., Lee, B., Lee, H., & Kim, E. (2014). An efficient genetic selection of the presentation order in simplified fuzzy ARTMAP patterns. *Applied Soft Computing, 22*, 101–107.
Breiman, L. (2001). Random forests. *Machine Learning, 45*, 5–32.
Carpenter, G. A., Grossberg, S., Markuzon, N., Reynolds, J. H., & Rosen, D. B. (1992). Fuzzy ARTMAP: a neural network architecture for incremental supervised learning of analog multidimensional maps. *IEEE Transactions Neural Networks, 3*, 698–713.
Cortes, C., & Vapnik, V. (1995). Support-vector network. *Machine Learning, 20*, 1–25.
ElGibreen, H., & Aksoy, M. S. (2015). *Intelligent information and database systems: 7th Asian conference, ACIIDS 2015, Bali, Indonesia, March 23–25, 2015, proceedings, Part II* (pp. 116–127). Cham: Springer International Publishing.
ElGibreen, H., & Aksoy, M. S. (2014a). Incremental transfer RULES with incomplete data. In *Ieee* (pp. 257–264).
ElGibreen, H., & Aksoy, M. S. (2014b). RULES-IT: incremental transfer learning with RULES family. *Frontiers of Computer Science, 8*, 537–562.
ElGibreen, H., & Aksoy, M. S. (2015a). Inductive learning for continuous classes and the effect of RULES family. *International Journal of Information and Education Technology, 5*, 564–570.
Gabrys, B., & Bargiela, A. (2000). General fuzzy min-max neural network for clustering and classification. *IEEE Transactions Neural Networks, 11*, 769–783.
Lichman, M. (2013). *UCI machine learning repository [http://archive.ics.uci.edu/ml]*. Irvine, CA: University of California, School of Information and Computer Science.
Loo, C. K., Liew, W. S., Seera, M., & Lim, E. (2015). Probabilistic ensemble Fuzzy ARTMAP optimization using hierarchical parallel genetic algorithms. *Neural Computing and Applications, 26*, 263–276.

Michalski, R. S. (1990). A theory and methodology of inductive learning. In J. W. Shavlik, & T. G. Dietterich (Eds.), *Readings in Machine Learning* (pp. 70–95). San Mateo, California: Morgan Kaufmann.

Oong, T. H., & Isa, N. A. M. (2014). Feature-based ordering algorithm for data presentation of fuzzy artmap ensembles. *IEEE Transactions Neural Networks and Learning Systems, 25*, 812–819.

Ortega, P. A., Figueroa, C. J., & Ruz, G. A. (2006). A medical claim fraud/abuse detection system based on data mining: A case study in Chile. In *The 2006 international conference on data mining (DMIN'06), Las Vegas, Nevada, USA* (pp. 224–231).

Othman, O. M., & Bryant, C. H. (2015). Pruning classification rules with instance reduction methods. *International Journal of Machine Learning and Computing, 5*, 187–191.

Packianather, M. S., & Drake, P. R. (2000). Neural networks for classifying images of wood veneer. Part 2. *International Journal of Advanced Manufacturing Technology, 16*, 424–433.

Palaniappan, R., & Eswaran, C. (2009). Using genetic algorithm to select the presentation order of training patterns that improves simplified fuzzy ARTMAP classification performance. *Applied Soft Computing, 9*, 100–106.

Pham, D. T., & Aksoy, M. S. (1993). An algorithm for automatic rule induction. *Artificial Intelligence in Engineering, 8*, 277–282.

Pham, D. T., & Aksoy, M. S. (1995a). A new algorithm for inductive learning. *Journal of Systems Engenering, 5*, 115–122.

Pham, D. T., & Aksoy, M. S. (1995b). RULES: A simple rule extraction system. *Expert Systems with Applications, 8*, 59–65.

Pham, D. T., Bigot, S., & Dimov, S. S. (2003). RULES-5: A rule induction algorithm for classification problems involving continuous attributes. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 217*, 1273–1286.

Pham, D. T., Bigot, S., & Dimov, S. S. (2004). A rule merging technique for handling noise in inductive learning. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 218*, 1255–1268.

Pham, D. T., & Dimov, S. S. (1997). An efficient algorithm for automatic knowledge acquisition. *Pattern Recognition, 30*, 1137–1143.

Pham, D. T., Dimov, S. S., & Salem, Z. (2000). Technique for selecting examples in inductive learning. In *ESIT 2000 European symposium on intelligent techniques, Erudit Aechen, Germany* (pp. 119–127).

Pham, D. T., & Salem, Z. (2004). Improved inductive learning using training data reorganisation. In *International conference on information and communication technologies: From theory to applications* (pp. 441–442).

Pourpanah, F., Lim, C. P., & Saleh, J. M. (2016). A hybrid model of fuzzy ARTMAP and genetic algorithm for data classification and rule extraction. *Expert Systems with Applications, 49*, 74–85.

Quinland, J. R. (1983). Learning efficient classification procedures and their applications to chess end games. In R. S. Michalski, J. G. Carbonell, & T. M. Mitchell (Eds.), *Machine learning an artificial intelligence approach* (pp. 463–482). Los Altos, California: Morgan Kaufmann.

Rizzi, A., Panella, M., & Mascioli, F. M. F. (2002). Adaptive resolution min-max classifiers. *IEEE Transactions Neural Networks, 13*, 402–414.

Ruz, G. A., Estévez, P. A., & Perez, C. A. (2005). Neurofuzzy color image segmentation method for wood surface defect detection. *Forest Products Journal, 55*, 52–58.

Ruz, G. A., Estévez, P. A., & Ramírez, P. A. (2009). Automated visual inspection system for wood defect classification using computational intelligence techniques. *International Journal of Systems Science, 40*, 163–172.

Ruz, G. A., & Pham, D. T. (2009). Building Bayesian network classifiers through a Bayesian complexity monitoring system. *Proceedings of the Institution of Mechanical Engineers, Part C: Journal of Mechanical Engineering Science, 223*, 743–755.

Ruz, G. A., Varas, S., & Villena, M. (2013). Policy making for broadband adoption and usage in chile through machine learning. *Expert Systems with Applications, 40*, 6728–6734.

Seera, M., Lim, C. P., Loo, C. K., & Singh, H. (2015). A modified fuzzy min-max neural network for data clustering and its application to power quality monitoring. *Applied Soft Computing, 28*, 19–29.

Shinde, S., & Kulkarni, U. (2016). Extracting classification rules from modified fuzzy min-max neural network for data with mixed attributes. *Applied Soft Computing, 40*, 364–378.

Simpson, P. K. (1992). Fuzzy min-max neural networks-Part 1: classification. *IEEE Transactions Neural Networks, 3*, 776–786.

Simpson, P. K. (1993). Fuzzy min-max neural networks-Part 2: clustering. *IEEE Transactions Fuzzy Systems, 1*, 32–45.

Tou, J. T., & Gonzalez, R. C. (1974). *Pattern Recognition Principles*. Addison-Wesley.

Valle, M. A., Varas, S., & Ruz, G. A. (2012). Job performance prediction in a call center using a naive Bayes classifier. *Expert Systems with Applications, 39*, 9939–9945.

Xie, Y., Li, X., Ngai, E. W. T., & Ying, W. (2009). Customer churn prediction using improved balanced random forests. *Expert Systems with Applications, 36*, 5445–5449.