Expert Systems with Applications 38 (2011) 12483–12490

Contents lists available at ScienceDirect

Expert Systems with Applications

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

Determination of relative agrarian technical efficiency by a dynamic over-sampling procedure guided by minimum sensitivity

Francisco Fernández-Navarro [a,*], César Hervás-Martínez [a], C. García-Alonso [b], M. Torres-Jimenez [b]

[a] Department of Computer Science and Numerical Analysis, University of Córdoba, Campus de Rabanales, Albert Einstein Building, 3rd Floor, 14071 Córdoba, Spain
[b] Department of Management and Quantitative Methods, ETEA, Escritor Castilla Aguayo 4, 14004 Córdoba, Spain

ARTICLE INFO

Keywords:
Neural networks
Multi-classification
Sensitivity
Accuracy
DEA-Montecarlo
Hybrid algorithm
Imbalanced datasets
Oversampling method
SMOTE
APS

ABSTRACT

In this paper, a dynamic over-sampling procedure is proposed to improve the classification of imbalanced datasets with more than two classes. This procedure is incorporated into a Hybrid algorithm (HA) that optimizes Multi Layer Perceptron Neural Networks (MLPs). To handle class imbalance, the training dataset is resampled in two stages. In the first stage, an over-sampling procedure is applied to the minority class to partially balance the size of the classes. In the second, the HA is run and the dataset is oversampled in different generations of the evolution, generating new patterns in the minimum sensitivity class (the class with the worst accuracy for the best MLP of the population). To evaluate the efficiency of our technique, we pose a complex problem, the classification of 1617 real farms into three classes (efficient, intermediate and inefficient) according to the Relative Technical Efficiency (RTE) obtained by the Monte Carlo Data Envelopment Analysis (MC-DEA). The multi-classification model, named Dynamic Smote Hybrid Multi Layer Perceptron (DSHMLP) is compared to other standard classification methods with an over-sampling procedure in the preprocessing stage and to the threshold-moving method where the output threshold is moved toward inexpensive classes. The results show that our proposal is able to improve minimum sensitivity in the generalization set (35.00%) and obtains a high accuracy level (72.63%).

1. Introduction

Classification problems based on imbalanced training datasets often occur in applications where there are rarely events of interest. That is, the size of interesting minority groups is usually in a rather small proportion in the training dataset (Chawla, Japlowicz, & Kotcz, 2006; Zhao & Huang, 2007). Imbalanced training datasets often results in low classification accuracies for minority classes (He & Garcia, 2009; Sun, Wong, & Kamel, 2009; Torres, Hervás, & García, 2009).

Many techniques are proposed to solve this kind of classification problem through either data (Kubat & Matwin, 1997) or algorithmic levels (Pazzani et al., 1994). In this paper, a dynamic over-sampling procedure (hybrid approach between data and algorithmic solutions) is proposed to improve the classification of imbalanced datasets that have more than two classes. The base over-sampling procedure is the Synthetic Minority Over-sampling Technique (SMOTE) (Chawla, Bowyer, Hall, & Kegelmeyer, 2002). This procedure has been applied in several research fields, for example in predictive microbiology (Fernández-Navarro et al., 2010; Fernández-Navarro, Hervás-Martínez, Cruz, Gutierrez, & Valero, 2011).

This procedure is incorporated into a Hybrid algorithm (HA) (Moscato & Cotta, 2003) that optimizes Multi Layer Perceptron Neural Networks (MLPs). The HA combines an Evolutionary algorithm (EA) (Back, 1996), a clustering process, and a Local Search (LS) procedure. The main objective of this research is, due to the unbalanced class structure (Fernández, Del Jesus, & Herrera, 2009; Sun et al., 2009), to check dynamic oversampling methods, where the class that increases its size is the one that has minimum sensitivity (*MS*) during the evolutive process. The base algorithm was proposed in Fernández-Navarro, Hervás-Martínez, and Gutíerrez (2011).

In recent years, several research projects related to DEA models have been developed, in the area of data mining, of which we highlight the papers by Toloo, Sohrabi, and Nalchigar (2009) and Yeh, Chi, and Hsu (2009). In the research works of Wu (2009) and Tsai, Lin, Cheng, and Lin (2009), the combination of neural networks and DEA models have already been applied successfully.

The performance of the proposed methodology was evaluated in a real problem which consists of classifying 1617 farms into three classes (efficient, intermediate and inefficient) according to Relative Technical Efficiency (RTE) obtained by use of the Monte

* Corresponding author. Tel.: +34 957 21 83 49; fax: +34 957 21 83 60.
E-mail address: i22fenaf@uco.es (F. Fernández-Navarro).

0957-4174/$ - see front matter © 2011 Elsevier Ltd. All rights reserved.
doi:10.1016/j.eswa.2011.04.031

Carlo Data Envelopment Analysis (MC-DEA) model on the 65 Agrarian Productive Strategies (APS) or typologies identified in the original database. The classification problem is very complex due to unbalanced class structure and the way in which this has determined the class each farms belongs. (see Section 3.1.1).

This paper is organized as follows: Section 2 describes the base classifier, the learning algorithm and over-sampling approaches; Section 3 explains the experiments carried out and a brief analysis of the database; Section 4 reports on the results obtained with the proposed methods and the results with methodologies used for comparative purposes and, finally, Section 5 summarizes the conclusions of our work.

2. Classification method

2.1. Base classifier

In this paper, we consider standard feed forward MLP with one input layer with independent variables or features, one hidden layer with sigmoidal hidden nodes and one output layer.

Let a coded "1-of-J" outcome variable y, (that is the outcome has the form $\mathbf{y} = (y^{(1)}, y^{(2)}, \ldots, y^{(J)})$, where $y^{(j)} = 1$ if the pattern belongs to the class j, and $y^{(j)} = 0$, otherwise); and a vector $\mathbf{x} = (1, x_1, x_2, \ldots, x_K)$ of input variables, where K is the number of input (we assume that the vector of inputs includes the constant term to accommodate the intercept or bias).

Then, the output layer is interpreted from a point of view of probability which considers the softmax activation function. The activation function of the lth node in the hidden layer is given by:

$$g_l(\mathbf{x}, \boldsymbol{\theta}_l) = \frac{\exp f_l(\mathbf{x}, \boldsymbol{\theta}_l)}{\sum_{l=1}^{J} \exp f_l(\mathbf{x}, \boldsymbol{\theta}_l)}, \quad l = 1, 2, \ldots, J \tag{1}$$

where $g_l(\mathbf{x}, \boldsymbol{\theta}_l)$ is the probability a pattern $\mathbf{x}$ has of belonging to class l, $\boldsymbol{\theta}_l = \left(\beta_0^l, \ldots, \beta_M^l, \mathbf{w_1}, \ldots, \mathbf{w_M}\right)$ is the vector of weights of the output node, M is the number of hidden nodes, $\mathbf{w_j} = \left\{w_0^j, \ldots, w_K^j\right\}$, for $j = 1, \ldots, M$, is the vector of inputs weights of the hidden node j, and $f_l(\mathbf{x}, \boldsymbol{\theta}_l)$ is the output of the output node for pattern $\mathbf{x}$ given by:

$$f_l(\mathbf{x}, \boldsymbol{\theta}_l) = \beta_0^l + \sum_{j=1}^{M} \beta_j^l \sigma\left(w_0^j + \sum_{i=1}^{K} w_i^j x_i\right), \quad \text{for } l = 1, \ldots, J \tag{2}$$

where $\sigma(\cdot)$ is the sigmoidal activation function.

The classification rule $C(\mathbf{x})$ of the MLP model is $C(\mathbf{x}) = \arg\max\{g_l(\mathbf{x}, \boldsymbol{\theta}_l)\}$, this classification rule coinciding with the optimal Bayes' rule.

The best MLP is determined by means of a Hybrid algorithm (HA) that optimizes the error function given by the negative log-likelihood for N observations associated with the MLP model:

$$L^*(\boldsymbol{\theta}) = \frac{1}{N} \sum_{n=1}^{N} \left[-\sum_{l=1}^{J-1} y_n^{(l)} f_l(\mathbf{x}_n, \boldsymbol{\theta}_l) + \log \sum_{l=1}^{J-1} \exp f_l(\mathbf{x}_n, \boldsymbol{\theta}_l) \right] \tag{3}$$

where $y_n^{(l)}$ is equal to 1 if the pattern $\mathbf{x_n}$ belongs to the lth class and equal to 0 otherwise.

2.2. Performance measures: correct classification rate and minimum sensitivity

Minimum sensitivity (MS) and the correct classification rate or accuracy (C) measures associated with a given classifier g are considered to be the performance measures in this work.

Firstly, we have to define the MS and C measurements which are derived from the contingency or confusion matrix M.

$$M = \left\{ n_{ij};\ \sum_{i,j=1}^{J} n_{ij} = N \right\} \tag{4}$$

where J is the number of classes, N is the number of training or testing patterns and n_{ij} represents the number of times the patterns are predicted by classifier g to be in class j when they really belong to class i. The diagonal corresponds to correctly classified patterns and the off-diagonal to mistakes in the classification task.

Let us denote the number of patterns associated with class i by $f_i = \sum_{j=1}^{J} n_{ij}$, $i = 1, \ldots, J$. Let $S_i = n_{ii}/f_i$ be the number of patterns correctly predicted to be in class i with respect to the total number of patterns in class i (sensitivity for class i). Therefore, the sensitivity for class i estimates the probability of correctly predicting a class i example.

From the above quantities the minimum sensitivity (MS) of a classifier g is the minimum value of the sensitivities for each class:

$$MS = \min\{S_i; i = 1, \ldots, J\} \tag{5}$$

The correct classification rate or accuracy (C) is defined as:

$$C = (1/N) \sum_{j=1}^{J} n_{jj} \tag{6}$$

that is, the rate of all correct predictions.

Minimum sensitivity and accuracy measures express two features associated with a classifier: global performance C and the accuracy for the worst classified class S. These measures have been simultaneously taken into account in previous studies (Martínez-Estudillo, Gutiárrez, Hervás-Martínez, & Fernández, 2008), achieving good performance for the classification of imbalanced data. In this paper, the application of dynamic over-sampling techniques improves the sensitivity of the classifier population, without drastically decreasing global accuracy.

2.3. Base evolutionary algorithm

An evolutionary algorithm is applied to estimate the structure and learn the weights of standard MLP neural networks models. The basic framework of the evolutionary algorithm is the following: the search begins with an initial population of neural networks and, in each iteration, the population is updated using a population-update algorithm which evolves both its structure and weights. The population is subject to the operations of replication and mutation. Crossover is not used due to its potential disadvantages in evolving artificial networks (Angeline, Sauders, & Pollack, 1994; Fernández-Navarro, Hervás-Martínez, Gutierrez, & Carboreno, in press; Yao & Liu, 1997).

The algorithm evolves architectures and connection weights simultaneously, each individual being a fully specified MLP. Neural networks are represented using an object-oriented approach and the algorithm deals directly with the MLP phenotype. Each connection is specified by a binary value indicating if the connection exists and a real value representing its weight. As the crossover is not considered, this object-oriented representation does not assume a fixed order between different hidden nodes. The general structure of the EA has been included in Fig. 1, where N and p_m are parameters of the algorithm.

We considered $L^*(\boldsymbol{\theta})$ defined in (3) as the error function of an individual g in the population. The fitness measure needed to evaluate the individuals is a strictly decreasing transformation of the error function $L^*(\boldsymbol{\theta})$ given by

$$A(g) = \frac{1}{1 + L^*(\boldsymbol{\theta})}; \quad 0 < A(g) \leqslant 1 \tag{7}$$

The severity of both structural and parametric mutations depends on the temperature $T(g)$ of the neural network model, defined by:

$$T(g) = 1 - A(g), \quad 0 \leqslant T(g) \leqslant 1 \tag{8}$$

where $A(g)$ is the fitness of the individual or model g.

1: Base Evolutionary Algorithm:
2: Generate a random population of size N
3: **repeat**
4: Calculate the fitness of every individual in the population
5: Rank the individuals with respect to their fitness
6: The best individual is copied into the new population
7: The best 10% of individuals in the population are replicated and substitute the worst 10% of individuals
8: Apply parametric mutation to the best $(p_m)\%$ of individuals
9: Apply structural mutation to the remaining $(100 - p_m)\%$ of individuals
10: **until** the stopping criterion is fulfilled

Fig. 1. Base evolutionary algorithm (EA) framework.

Given the vector of parameters representing the MLP, $\theta = (\theta_1, \ldots, \theta_Q)$, parametric mutation (Fig. 1, step 8) is accomplished for each weight $w \in \theta$ adding Gaussian noise:

$$w(t+1) = w(t) + \xi(t) \tag{9}$$

where $\xi(t)$ represents a one dimensional normally distributed random variable, $N(0, \alpha \cdot T(g))$. The α value is updated throughout the evolutionary process, applying the simplest heuristic 1/5 success rule of Rechenberg (1973). The weights are sequentially mutated, hidden node after hidden node, and a standard simulated annealing process is applied to accept or reject the modifications in each node.

On the other hand, structural mutation (Fig. 1, step 9) implies a modification in the neural network structure and allows explorations of different regions in the search space while helping to keep up the diversity of the population. There are four different structural mutations: node deletion, connection deletion, node addition and connection addition. These four mutations are applied sequentially to each network.

For each mutation, there is a minimum value $\Delta_{\min}$ and a maximum value $\Delta_{\max}$, and the number of elements (nodes and connections) involved in the mutation is calculated as

$$\Delta_{\min} + uT(g)(\Delta_{\max} - \Delta_{\min}) \tag{10}$$

where u is a random uniform variable in the interval $[0,1]$. All the above mutations are made sequentially in the given order, with probability $T(g)$, in the same generation on the same network. If probability does not select any mutation, one of the mutations is chosen at random and applied to the network. Finally, a maximum number of hidden nodes m is used to control the final complexity of the MLPs.

The stop criterion is reached when the following condition is fulfilled: for 20 generations there is not improvement in the average performance of the best 10% of the population or when 500 generations are completed.

For further details about parametric and structural mutations and other characteristics of the algorithm see the papers of Gutiérrez, Hervás-Martínez, Carbonero, and Fernández (2009) and Martínez-Estudillo, Hervás-Martínez, Gutiárrez, and Martínez-Estudillo (2008).

2.4. The Hybrid Multi Layer Perceptron algorithm (HMLP)

The Hybrid Multi Layer Perceptron (HMLP) consists of applying the previously described base evolutionary algorithm but including a local search to some specifically selected individuals. This Hybrid algorithm (HA) includes an optimization clustering process applied every 50 generations of the evolutionary algorithm. In this clustering process, each neural network model or individual is represented by the set of the accuracies of the neural network model for each class of the problem (called Sensitivity Clustering in Fig. 2). The clustering algorithm is able to obtain groups of individuals that show similar behavior for different classes. After that, we apply the *iRprop* + algorithm (Igel & Hüsken, 2003) to the individual closest to the centroid obtained in each cluster and the optimized individual with the best minimum sensitivity value in the training set is returned as the final solution (Hybrid Evolutionary algorithm with the Sensitivity Clustering solution, HEASC solution in Fig. 2). This solution is stored every 50 generations. The final solution is the individual with the best minimum training sensitivity value among the local optima found during the evolutionary process. The methodology proposed is described in Fig. 2

Another feature of our approach is that the optimized individuals are not included in the new population. Once the optimization algorithm is applied, we think that any further modification, by some mutation, of the individual will be counterproductive, because the HA does not include a crossover operator and the optimized genotype will not be transferred. So, these individuals are stored in a separate population till the end of the EA unlike Memetic algorithms (MAs) where the optimized individuals are returned to the population (Whitley, Gordon, & Mathias, 1994).

This combination of a clustering process and a local optimization method for EAs was previously proposed in Martínez-Estudillo, Hervás-Martínez, Martínez-Estudillo, and García-Pedrajas (2006), and obtained good results in regression problems. In this paper, the method has been adapted for classification problems.

2.5. A dynamic over-sampling approach: Dynamic Smote Hybrid Multi Layer Perceptron algorithm (DSHMLP)

This section describes the Dynamic Smote Hybrid Multi Layer Perceptron (DSHMLP) algorithm. In this approach, the dataset is modified in two stages. Firstly, the dataset is changed before the algorithm performs (taking into account the number of patterns per class) and secondly, the dataset is increased by adding the number of patterns in the minimum sensitivity class in different generations of the HA. The DSHMLP algorithm is detailed in Fig. 3

The DSHMLP method include a pre-processing stage (see Step 1 Fig. 3) where the number of minority class patterns is added. The aim is to decrease the problem imbalance rate (He & Garcia, 2009; Sun, Kamel, Wong, & Wang, 2007) by selecting the minority class to apply the resampling procedure. Synthetic examples are obtained by the Synthetic Minority Over Sampling Technique (SMOTE) algorithm (Chawla et al., 2002) applied to minority class patterns. The procedure is performed subject to the following condition:

$$p^* \leqslant \frac{1}{2 \cdot J} \tag{11}$$

where J is the number of classes and p^* is the minimum of prior estimated probabilities (i.e. $p^* = \min\{(f_i/N),\ 1 \leqslant i \leqslant J\}$, where f_i is the number of patterns of the ith class and N is the total number of patterns). This condition was established since the preprocessing SMOTE should be applied only to the most imbalanced datasets (the size of the minority class is less than half of the size that this class should be in an ideal balanced case). The problem of classifying the real farms considered in this work fulfills that condition, therefore, the minority class is doubled in the preprocessing stage.

After that, the HA runs, and every 50 generations from the initial generation 25, the HA is stopped and the proposed over-sampling procedure is applied. The over-sampling procedure is defined as follows: first, the DSHMLP method selects the best MLP from the population and determines the class with minimum sensitivity (see Step 6 Fig. 3). If two or more classes are classified with the same minimum sensitivity, the minority class is selected.

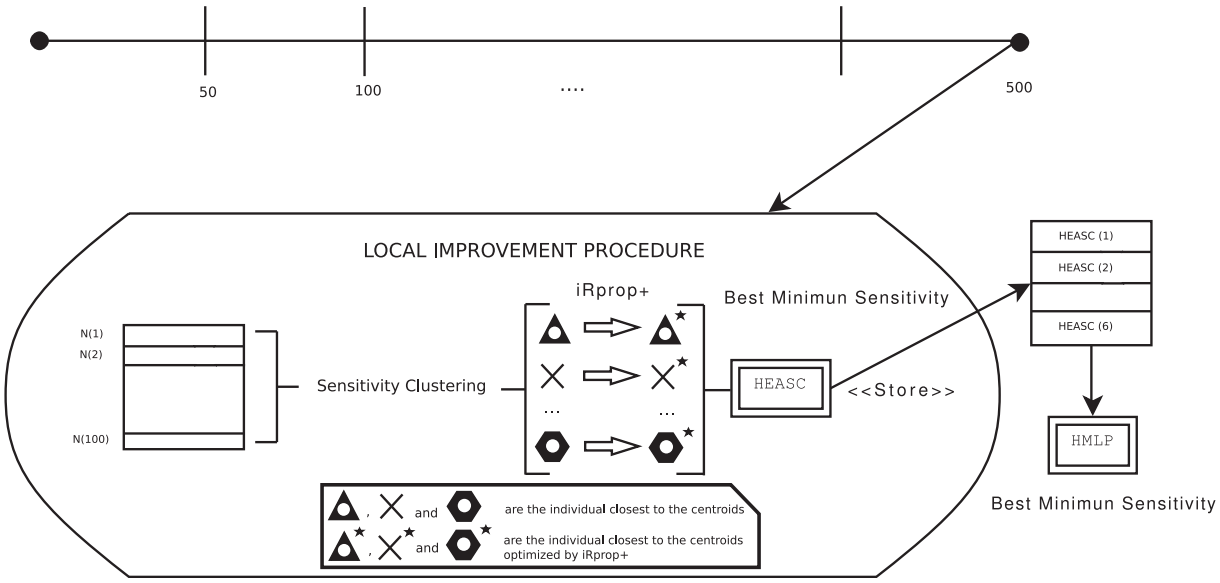


Fig. 2. Hybrid Multi Layer Perceptron methodology.

Dynamic Smote Hybrid Multi Layer Perceptron Algorithm:

1: Apply SMOTE algorithm to the minority class
2: Generate a random population of size N
3: **repeat**
4: Perform a base HA iteration
5: **if** is Over Sampling Epoch **then**
6: Determine the class with the minimum sensitivity value
7: **if** Number of patterns of the minimum sensitivity class > 5 **then**
8: $Neighbours \leftarrow 5$
9: **else**
10: $Neighbours \leftarrow$ Minimum Sensitivity Class Patterns $- 1$
11: **end if**
12: Apply SMOTE Algorithm to the minimum sensitivity class
13: Add synthetic patterns to training set
14: Evaluate individuals with the new training set
15: Sort individuals
16: Store the best individual
17: **end if**
18: **until** the stopping criterion is fulfilled

Fig. 3. Dynamic Smote Hybrid Multi Layer Perceptron algorithm (DSHMLP) framework.

The selected class is over-sampled by taking each pattern from the minimum sensitivity class and introducing synthetic examples along the line segments joining any/all of the k minority class nearest neighbors. Our implementation currently uses five nearest neighbors as the maximum value of the k parameter in the SMOTE algorithm (see Step 7–11 Fig. 3), as suggested by Chawla et al. (2002). The over-sampling method adds the number of patterns that the class had in the original dataset without considering synthetic patterns to generate new samples. Once synthetic patterns have been generated, these are inserted into the training set, resulting in the need to re-evaluate and sort the population according to fitness (see Step 13–16 Fig. 3). This procedure is performed whenever the following condition is fulfilled:

$$(C_t + \nabla C \geqslant C_{t-1}) \quad \text{and} \quad (MS_t - \Delta MS \geqslant MS_{t-1}) \tag{12}$$

where C_t, C_{t-1}, MS_t and MS_{t-1} are the values of C and MS of the best MLP in the over-sampling steps t and $t-1$ and ∇C and ΔMS are the decrement and increment values of C and MS, required carry out the next step of dynamic over-sampling. To test the over-sampling procedure, the configuration parameter values considered are, a ΔMS value of 5 and a ∇C of 2, since for high values of C, the MS measure can be in conflict with C.

3. Experiments

3.1. Database analysis

3.1.1. Determination of Relative Technical Efficiency (RTE)

In this study, 65 different types of Agrarian Productive Strategies (APS) were identified – in a sample of 1617 surveyed agrarian enterprises in the south of Spain- based on both the Gross Value Added (GVA) of the main productive activity and the size of the farm (very small, small, medium sized, big and very big) (Fig. 4, Activity 1). In all the APS, costs and revenues were determined as random variables and fitted to standard statistical distributions (uniform, triangular and trapezoidal). In order to evaluate RTE, Monte Carlo Data Envelopment Analysis (MC-DEA) (Hu, Lai, & Huang, 2009; Liang, Li, & Li, 2009) was used selecting the Banker–Charnes–Cooper (BCC) input-oriented model. The BCC-DEA model was selected because, in this framework, there is no evidence of a constant return to scale environment. In each simulation, a new BCC-DEA model is generated by the Monte-Carlo engine. Input and output values were also interpreted (Lin, Lee, & Chiu, 2009; Zerafat Angiz, Emrouznejad, Mustafa, & Rashidi Komijan, 2009) if they were considered non-standard or undesired (positive inputs, negative outputs, etc.). RTE scores generated a statistical distribution of the efficiency of each productive strategy analysed (Fig. 4, Activity 2).

For each APS analysed (65), 2500 simulations were carried out (Fig. 4, Activity 3). Using the frequency results of the 65 APS in each simulation, a k-means algorithm was used and obtain the best inter and intra-group variance results to identify different effyciency-based classes of APS (Fig. 4, Activity 4). From this analysis, 3

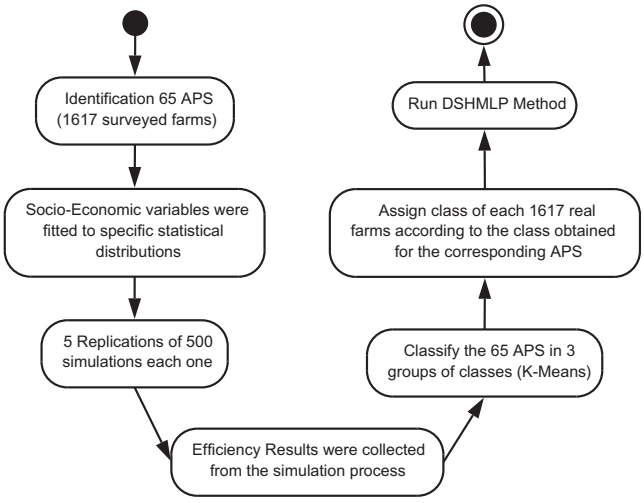


Fig. 4. Structure of the analysis.

different groups or classes were identified: efficient, intermediate and inefficient agrarian productive strategies.

The real farms surveyed inherited the corresponding class – efficient, intermediate and inefficient – obtained for their APS (Fig. 4, Activity 6). Therefore, the classes that were found carrying out the BCC-DEA model in a selected group of APS were assigned to all the farms included in the corresponding agrarian productive strategy.

3.1.2. Database description

The complete socio-economic structure of 1617 agrarian farms comprised the database analysed. This database included information about: farmer characteristics, mechanization, the size of the farm and, finally, the costs and revenues of all productive activities. Micro-economic information was grouped into the most consistent set of variables that described the activity and production of the farms surveyed. The descriptive variables of agrarian enterprises were the following: Crop revenues (Rev) and subsidies (Sub), land rental, fallow land revenues, etc. (ORev), diversification revenues, service made to others and other revenues (DRev), number of crops (# Crops), seed and plant costs (Seed), fertilizer, pesticide hand labor and energy costs (Fert, Pest and HLab), other cost of crops (OCrops), energy costs of the enterprise (Ener), service costs of the enterprise (Serv), Financial, fallow land maintenance and land rental costs (Fin), social security costs and taxes (Tax), maintenance costs excluding hand labor (Main), hand labor costs in maintenance (HLma) and revenues obtained outside agrarian activity (ERev). The main statistical characteristics of all input variables can bee seen in Table 1.

The first four variables (Rev, Sub, ORev and DRev) were considered to be outputs and the rest, 13 in total, as inputs. All of these variables except #Crop were homogenized per area (hectares, ha) to eliminate the influence of farm size. The farm size was included in the definition of the APS that were analysed initially (see Section 3.1.1).

In order to make the BCC-DEA analysis easier, all the original data was standardized within a [0,50] range because this strategy allowed the model to avoid extreme numerical values when their statistical distributions were identified (see Table 1). In addition to this data, all the 1617 farms analysed were described by an efficiency-based class obtained from the k-means method ($k = 3$) that analysed the frequency scores obtained from MC-BCC-DEA. So, each farm was grouped into one of the following categories or classes: efficient, intermediate and inefficient.

The first cluster obtained groups the intermediate farms where the probability of being efficient is within [0.548,0.766]-average 0.693, being, on the other hand, the probability of being weak-efficient within [0.166,0.416] (see Table 2). Efficient agrarian enterprises are in cluster 2, in this group the probability of being efficient increases a lot and is located within [0.739,0.97]-average 0.843. Finally, non efficient productive strategies are located in cluster 3, the probability of being efficient is low and within [0.417,0.698]-average 0.589 (see Table 2).

Once the k-means analysis was carried out, a discriminant analysis on each APS was used to analyze the validity of the classification obtained (3 efficiency-based groups). The 65 APS dataset was randomly divided 10 times in two samples following the guidelines of Prechelt (1994): training (60%–70%) and generalization (40%–30%) sets. Results obtained showed that all the classes were very well recognized with a percentage higher than 95.20% (generalization) in the worst sample design. Other designs with 4 and 5 classes achieved worse results using the same statistical procedure. Take into account that the classification in 3 classes (efficient, intermediate and inefficient APS) is the best, all the original farms (1617) inherit the class of the corresponding. In consequence, all the 1617 farms are described by their socio-economic variables and belong to a specific efficiency-based class. The problem that arises is the identification of the class (classification output) taking the socio-economic variables as the input variables (classification inputs: number of crops, costs and

Table 1
Statistics (1617 cases) of each of the input variables.

Variable	Mean	Median	Mode	Std.Dev.	V. Coeff. (%)	Skewness	Kurtosis	Min.	Max.
Rev	0.40	0.07	0.01	1.83	459	20.02	489.27	0	50
Sub	1.63	0.69	0.00	3.19	196	6.40	66.12	0	50
ORev	0.07	0.00	0.00	1.30	1817	35.40	1345.78	0	50
DRev	0.28	0.00	0.00	2.33	844	14.22	235.90	0	50
#Crop	1.63	1.00	1.00	0.97	60	1.84	3.65	1	7
Seed	0.49	0.01	0.00	2.78	568	11.32	153.25	0	50
Fert	2.15	0.56	0.00	4.64	216	4.42	25.62	0	50
Pest	1.19	0.13	0.00	3.68	310	7.13	69.94	0	50
HLab	1.29	0.24	0.30	3.43	266	7.05	69.19	0	50
OCost	0.30	0.02	0.00	1.94	641	17.37	363.52	0	50
Ener	0.25	0.04	0.00	1.61	632	21.82	601.70	0	50
Serv	0.51	0.05	0.00	1.91	376	13.61	295.56	0	50
Fin	0.61	0.00	0.00	2.57	422	9.65	133.24	0	50
Tax	0.67	0.16	0.00	2.36	353	14.35	260.32	0	50
Main	0.90	0.05	0.00	3.43	383	7.16	64.39	0	50
HLma	1.17	0.09	0.00	3.97	340	6.85	58.77	0	50
ERev	0.22	0.00	0.00	1.88	839	19.03	416.95	0	50

Table 2
Basic statistics on probabilities of being weakly efficient and efficient (frequency analysis carried out on efficiency scores obtained using the MC-DEA model for 65 agrarian productive strategies).

	Cluster 1		Cluster 2		Cluster 3	
	Weak Eff.	Efficient	Weak Eff.	Efficient	Weak Eff.	Efficient
Min.	0.17	0.55	0.02	0.74	0.03	0.42
Max.	0.42	0.77	0.20	0.97	0.06	0.70
Mean	0.26	0.69	0.11	0.84	0.13	0.59
Median	0.24	0.71	0.10	0.84	0.13	0.60
SD	0.06	0.07	0.05	0.06	0.06	0.09

SD: Standard Deviation.

revenues, Table 1) in a non-balanced database with 1617 observations – agrarian enterprises.

3.2. Experimental design

As it was mentioned in Section 3.1.2, the proposed classification methodologies are used to identify the efficiency-based class (efficient, intermediate and inefficient) of 1617 farms using as classification inputs their socio-economic variables (see Table 1). The dataset has been included on a public website.[1]

The experimental design was conducted using a 10-fold cross validation stratified by APS, with 3 repetitions per each fold. The performance of each method has been evaluated using the correct classification rate (C_G) and the Minimun Sensitivity (MS_G) value for the generalization set, i.e. the accuracy for the class that had the worst classification.

The parameter values used in the hybrid techniques proposed were the following: we have done a simple linear rescaling of the input variables in the interval $[-2,2]$, X_i^* being the transformed variables. The connection between hidden and output layer are initialized in the $[-5,5]$ interval. The maximum and minimum number of sigmoidal units in the hidden layer is in the interval [10,20].

The size of the population is $N = 100$. For the structural mutation, the number of nodes that can be added or removed is within the [1,2] interval, and the number of connections to add or delete in the hidden and the output layer during structural mutations is within the [1,7] interval.

The DSHMLP method is compared to different algorithms:

- The HMLP method (detailed in Section 2.4).
- Specific methods for imbalanced data proposed in Zhou and Liu (2006):
  - The OverSampling (OS) algorithm. This method duplicates higher-cost training examples until the appearances of different training examples are proportional to their costs.
  - The SmoteOverSampling (SOS) algorithm. Implementation of the SMOTE algorithm in the preprocessing stage to balance in part the datasets. Then, the neural network is trained with the modified dataset.
  - The ThresholdMovNN (TMNN) algorithm. This method moves the output threshold toward inexpensive classes such that examples with higher costs become harder to be misclassified.

These methods have been selected due to their similarities to the model we have proposed. The first two techniques modify the distribution of the training dataset so that the costs of the examples are conveyed explicitly in the appearance of the examples. These methods use Multilayer Perceptron Neural Networks (MLPs) as the base classifier, and the model is trained by the RProp algorithm.

The HMLP algorithm was implemented in JAVA. For the DSHMLP methods, the HMLP algorithm was modified slightly, applying over-sampling procedures. We also used CSNN (Zhou & Liu, 2006) software package[2] to obtain the results in the OS, SmoteOS and TMNN methods.

Table 3
Comparison with other statistical and artificial intelligence methods.

Methodology	C_G (%)	MS_G (%)
OS	63.09 ± 5.85	33.27 ± 15.28
TMNN	65.22 ± 6.28	28.42 ± 15.03
SOS	63.95 ± 5.40	*34.65 ± 12.29*
HMLP	**74.67 ± 5.54**	18.19 ± 12.64
DSHMLP	*72.63 ± 2.86*	**35.00 ± 14.14**

The best result is in bold face and the second best result in italics.

4. Results

A comparison has been made of the DSHMLP method with the well known classification techniques given in Section 3.2. Table 3 shows the results obtained with the different techniques tested. A descriptive analysis of the results leads to the following remarks: the DSHMLP method obtained the best result in terms of MS_G and the second best result in C_G comparing over all techniques.

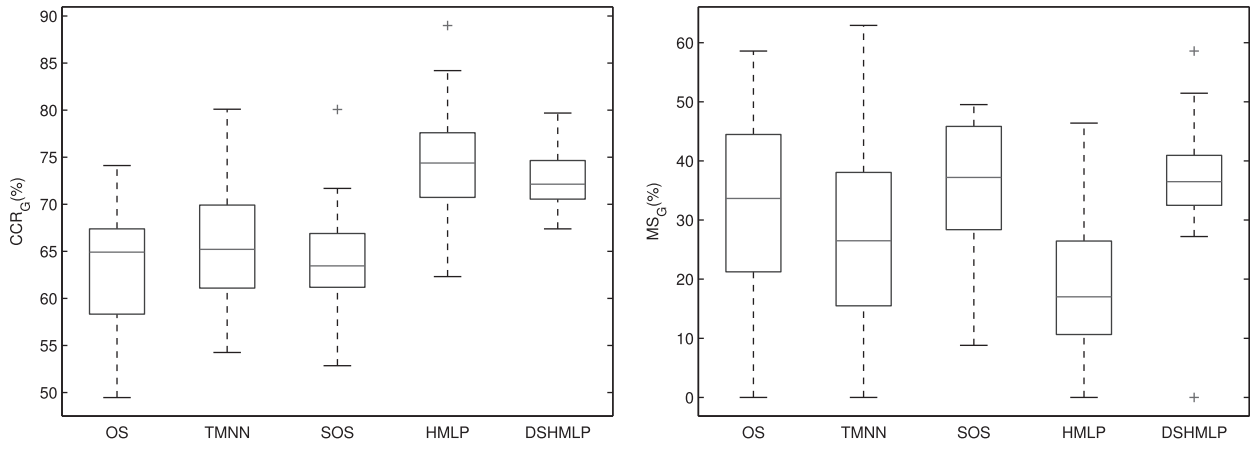
Fig. 5(a) and (b) show the boxplots obtained with the results of the different algorithms in C_G and MS_G. Boxplots depict groups of algorithms results through the smallest observation, lower quartile, median, upper quartile and largest observation. As we can see in Fig. 5(a) and (b), in C_G, the DSHMLP method generated the lowest degree of dispersion of results obtained, as shown by standard deviation values (2.86 for C_G, see Table 3).

To ascertain the statistical significance of the differences between the means (in C_G and MS_G for each stochastic methodology: OS, TMNN, SOS, HMLP and DSHMLP), the non-parametric Kolmogorov–Smirnov test (K–S test) with $\alpha = 0.05$, was used to evaluate if the C_G and MS_G values followed normal distribution. As can be seen from the results in Table 4, normal distribution can be assumed because the critical levels, p-values, were higher than 0.05 in all cases. In order to determine the best methodology (in the sense of its influence on the accuracy and on the minimum sensitivity in the generalization set, C_G and MS_G), an ANOVA statistical method test was carried out. The results of the ANOVA analysis for C_G and MS_G values show that the effect of the methodology was statistically significant at a signification level of 5% (see first row of Table 5). Once this test guaranteed that there were significant differences between the results found by different methods, we performed a multiple comparison test on the C_G and MS_G values in order to establish an order among the different methods (see Table 5). First, we carried out a Levene test (Miller, 1996) to evaluate the equality of variances. Then, a Tukey test (Miller, 1981) was performed, because the variances were equal (either for C_G or MS_G), in order to rank the different methods. Our aim was to find the methodology that performed (in C_G and/or MS_G) significantly better than the other methodologies.

The two best results in C_G were achieved by the DSHMLP and HMLP methods. The procedure recommended by this paper is the DSHMLP method, because using a t-test in average C_G, did not produce significant differences compared to the HMLP method (for $\alpha = 0.05$), while in average MS_G, using another t-test, the DSHMLP

[1] http://www.uco.es/grupos/ayrna/index.php?lang=en ("Datasets" section).

[2] http://lamda.nju.edu.cn/datacode/CSNN.htm.



(a) $C_G(\%)$ results (b) $MS_G(\%)$ results

Fig. 5. Box plots: Results of the OS, TMNN, SOS, HMLP and DSHMLP methods.

Table 4
p-values of the Kolmogorov–Smirnov test applied for the generalization the normality of the distributions of the generalization correct classification rate and minimum sensitivity (C_G (%) and MS_G (%), respectively) of the models obtained comparing different methods.

Test variable	Kolmogorov–Smirnov test				
	OS	TMNN	SOS	HMLP	DSHMLP
C_G	0.504	1.000	0.892	0.995	0.910
MS_G	0.801	0.988	0.273	0.963	0.065

Table 5
p-values of the Snedecor's F ANOVA I test, ordered mean for the statistical multiple comparison Tukey test and t-test when considering the generalization correct classification rate and minimum sensitivity (C_G (%) and MS_G (%), respectively) of the models obtained comparing different methods.

	Test variable	
	C_G	MS_G
F (*p*-values)	0.000[a]	0.000[a]
Ranking of averages	$\mu_{HMLP} \geqslant \mu_{DSHMLP} > \mu_{TMNN}$ $\mu_{TMNN} \geqslant \mu_{SOS} \geqslant \mu_{OS}$	$\mu_{DSHMLP} \geqslant \mu_{SOS} \geqslant \mu_{OS}$ $\mu_{OS} \geqslant \mu_{TMNN} > \mu_{HMLP}$ $\mu_{DSHMLP} > \mu_{HMLP}$[a]

$\mu_A \geqslant \mu_B$: *A* yields better results than *B*, but the differences are not significant.
$\mu_A > \mu_B$: *A* yields better results than *B* with significant differences.
The binary relation $\geqslant$ is not transitive.
[a] Significant differences were found ($\alpha = 0.05$).

method obtained significant differences with respect to the HMLP method (for $\alpha = 0.05$, see third row of Table 5).

5. Conclusions

This paper combines three powerful techniques used in machine learning research: resampling procedures, evolutionary algorithms and neural networks. The approach carries out an adequate combination of the three elements to resolve the problem of classifying real farms. The Relative Technical Efficiency (RTE) of each farm has been determined by the Monte Carlo Data Envelopment Analysis (MC-DEA) model. It is important to note that the classification problem considered is within the scope of imbalanced multi-classification problems.

In general, the results obtained show that the approaches proposed, which are based on MLPs trained with HAs are robust enough to tackle the multi-classification of RTE in real farms, and obtain better results than the majority of existing alternative methods.

There are two future research directions suggested by this study: (i) a multi-objective approach considering both *MS* and *C* functions could be carried out; and (ii), since the (*MS*,*C*) measures are independent of the evolutionary algorithm and of the base classifier used, other types of base classifiers and evolutionary algorithms could be considered.

Acknowledgement

This work has been partially subsidized by the TIN 2008-06681-C06-03 project of the Spanish Inter-Ministerial Commission of Science and Technology (MICYT), FEDER funds and the P08-TIC-3745 project of the "Junta de Andalucía" (Spain). The research of Francisco Fernández-Navarro has been funded by the "Junta de Andalucia" Predoctoral Program, grant reference P08-TIC-3745.

References

Angeline, P. J., Sauders, G. M., & Pollack, J. B. (1994). An evolutionary algorithm that constructs recurrent neural networks. *IEEE Transactions on Neural Networks, 5*, 54–65.

Back, T. (1996). *Evolutionary algorithms in theory and practice*. Oxford.

Chawla, N. V., Bowyer, K. W., Hall, L. O., & Kegelmeyer, W. P. (2002). SMOTE: Synthetic minority over-sampling technique. *Journal of Artificial Intelligence Research, 16*, 321–357.

Chawla, N. V., Japlowicz, N., & Kotcz, A. (2006). Editorial: Special issue on learning from imbalanced data sets. *Aigkdd Explorations, 6*(1), 1–6.

Fernández, A., Del Jesus, M. J., & Herrera, F. (2009). On the influence of an adaptative inference system in fuzzy rule based classification systems for imbalanced data-sets. *Expert Systems with Applications* (36), 9805–9812.

Fernández-Navarro, F., Hervás-Martínez, C., Gutierrez, P. A., & Carboreno, M. (in press). Evolutionary q-Gaussian radial basis functions neural networks for multi-classification. Neural Networks. <http://dx.doi.org/10.1016/j.neunet.2011.03.014>.

Fernández-Navarro, F., Hervás-Martínez, C., Cruz, M., Gutierrez, P. A., & Valero, A. (2011). Evolutionary *q*-Gaussian radial basis function neural network to determine the microbial growth/no growth interface of *Staphylococcus aureus*. *Applied Soft Computing, 11*(3), 3012–3020.

Fernández-Navarro, F., Hervás-Martínez, C., & Gutíerrez, P. (2011). A dynamic over-sampling procedure based on sensitivity for multi-class problems. *Pattern Recognition, 44*(8), 1821–1833.

Fernández-Navarro, F., Valero, A., Hervás-Martínez, C., Gutíerrez, P., García-Gimeno, R., & Zurera-Cosano, G. (2010). Development of a multi-classification neural network model to determine the microbial growth/no growth interface. *International Journal of Food Microbiology, 141*, 203–212.

Gutiérrez, P. A., Hervás-Martínez, C., Carbonero, M., & Fernández, J. C. (2009). Combined projection and kernel basis functions for classification in evolutionary neural networks. *Neurocomputing, 72*(13–15), 2731–2742.

He, H., & Garcia, E. A. (2009). Learning from imbalanced data. *IEEE Transactions on Knowledge and Data Engineering, 21*(9), 1263–1284.

Hu, W., Lai, M., & Huang, H. (2009). Rating the relative efficiency of financial holding companies in an emerging economy: A multiple DEA approach. *Expert Systems with Applications, 36*(3), 5592–5599.

Igel, C., & Hüsken, M. (2003). Empirical evaluation of the improved Rprop learning algorithms. *Neurocomputing, 50*(6), 105–123.

Kubat, M., & Matwin, S. (1997). Addressing the curse of imbalanced training sets: One-sided selection. In *Proceedings of the 14th international conference on machine learning* (pp. 179–186).

Liang, L., Li, Y., & Li, S. (2009). Increasing the discriminatory power of DEA in the presence of the undesirable outputs and large dimensionality of data sets with PCA. *Expert Systems with Applications, 36*(3), 5895–5899.

Lin, T. T., Lee, C., & Chiu, T. (2009). Application of DEA in analyzing a bank's operating performance. *Expert Systems with Applications, 36*(5), 8883–8891.

Martínez-Estudillo, F. J., Gutiárrez, P. A., Hervás-Martínez, C., & Fernández, J. C. (2008). Evolutionary learning by a sensitivity-accuracy approach for multi-class problems. In *Proceedings of the 2008 IEEE congress on evolutionary computation (CEC'08)* (pp. 1581–1588). Hong Kong, China: IEEE Press.

Martínez-Estudillo, F. J., Hervás-Martínez, C., Gutiárrez, P. A., & Martínez-Estudillo, A. C. (2008). Evolutionary product-unit neural networks classifiers. *Neurocomputing, 72*(1–2), 548–561.

Martínez-Estudillo, A. C., Hervás-Martínez, C., Martínez-Estudillo, F. J., & García-Pedrajas, N. (2006). Hybridization of evolutionary algorithms and local search by means of a clustering method. *IEEE Transactions on Systems, Man and Cybernetics, Part B: Cybernetics, 36*(3), 534–545.

Miller, R. G. (1981). *Simultaneous statistical inference* (2nd ed.). New York, USA: Wiley.

Miller, R. (1996). *Beyond ANOVA, basics of app. statistics*. London: Chapman & Hall.

Moscato, P., & Cotta, C. (2003). A gentle introduction to memetic algorithms. In *Handbook of metaheuristics. International series in operations research and management science* (Vol. 57, pp. 105–144). New York: Springer.

Pazzani, M., Merz, C., Murphy, P., Ali, K., Hume, T., & Brunk, C. (1994). Reducing misclassification costs: Knowledge intensive approaches to learning from noisy data. In *Proceedings of the 11th international conference on machine learning (ICML-1994)*.

Prechelt, L. (1994). PROBEN1: A set of neural network benchmark problems and benchmarking rules. Technical Report 21/94, Fakultät für Informatik, Universität Karlsruhe.

Rechenberg, I. (1973). *Evolutionsstrategie: Optimierung technischer Systeme nach Prinzipien der biologischen Evolution*. Stuttgart: Frommann-Holzboog.

Sun, Y., Kamel, M. S., Wong, A. K. C., & Wang, Y. (2007). Cost-sensitive boosting for classification of imbalanced data. *Pattern Recognition, 40*(12), 3358–3378.

Sun, Y., Wong, A. K. C., & Kamel, M. S. (2009). Classification of imbalanced data: A review. *International Journal of Pattern Recognition and Artificial Intelligence, 23*(4), 687–719.

Toloo, M., Sohrabi, B., & Nalchigar, S. (2009). A new method for ranking discovered rules from data mining by DEA. *Expert Systems with Applications, 36*(4), 8503–8508.

Torres, M., Hervás, C., & García, C. (2009). Multinomial logistic regression and product unit neural network models: Application of a new hybrid methodology for solving a classification problem in the livestock sector. *Expert Systems with Applications, 36*(10), 12225–12235.

Tsai, M., Lin, S., Cheng, C., & Lin, Y. (2009). The consumer loan default predicting model – An application of DEA-DA and neural network. *Expert Systems with Applications, 36*(9), 11682–11690.

Whitley, D. L., Gordon, V. S., & Mathias, K. E. (1994). Lamarckian evolution, the Baldwin effect and function optimization. In Y. Davidor, H. P. Schwefel, & R. Männer (Eds.), *Parallel Problem Solving from Nature – PPSN III* (pp. 6–15). Berlin: Springer.

Wu, D. (2009). Supplier selection: A hybrid model using DEA, decision tree and neural network. *Expert Systems with Applications, 36*(5), 9105–9112.

Yao, X., & Liu, Y. (1997). A new evolutionary system for evolving artificial neural networks. *IEEE Transactions on Neural Networks, 8*(3), 694–713.

Yeh, C., Chi, D., & Hsu, M. (2009). A hybrid approach of DEA, rough set and support vector machines for business failure prediction. *Expert Systems with Applications, 37*(2), 1535–1541.

Zerafat Angiz, L. M., Emrouznejad, A., Mustafa, A., & Rashidi Komijan, A. (2009). Selecting the most preferable alternatives in a group decision making problem using DEA. *Expert Systems with Applications, 36*(5), 9599–9602.

Zhao, Z. Q., & Huang, D. S. (2007). A mended hybrid learning algorithm for radial basis function neural networks to improve generalization capability. *Applied Mathematical Modelling, 31*, 1271–1281.

Zhou, Z.-H., & Liu, X.-Y. (2006). Training cost-sensitive neural networks with methods addressing the class imbalance problem. *IEEE Transactions on Knowledge and Data Engineering, 18*(1), 63–77.