CO^2RBFN-CS: First Approach Introducing Cost-Sensitivity in the Cooperative-Competitive RBFN Design

María Dolores Pérez-Godoy[1(✉)], Antonio Jesús Rivera[1], Francisco Charte[2], and Maria Jose del Jesus[1]

[1] Department of Computer Science, University of Jaén, Jaén, Spain
{lperez,arivera,mjjesus}@ujaen.es
[2] Department of Computer Science and Artificial Inteligence, University of Granada, Granada, Spain
francisco@fcharte.com

Abstract. The interest in dealing with imbalanced datasets has grown in the last years, since they represent many real world scenarios. Different methods that address imbalance problems can be classified into three categories: data sampling, algorithmic modification and cost-sensitive learning. The fundamentals of the last methodology is to assign higher costs to wrong classification classes with the aim of reducing higher cost errors.

In this paper, CO^2RBFN-CS, a cooperative-competitive Radial Basis Function Network algorithm that implements cost-sensitivity is presented. Specifically, a cost parameter is introduced in the training stage of the algorithm. This parameter modifies the learning rate of the weights taking into account the right (or wrong) classification of the instance, the type of class (majority or minority) and the imbalance ratio of the data set.

Keywords: RBFNs · Imbalanced data sets · Cost sensitive

1 Introduction

Nowadays, there exist many real applications represented by data sets where the frequency (number of instances) of certain classes substantially exceeds the frequency of the remaining classes. Furthermore, in imbalanced data sets [7], as they are typically known, the importance resides in the fact that a minority class usually represents the concept of interest, for example the intruder in an intrusion detection system; whereas the other class represents the counterpart of that concept (standard users).

Standard classifiers often show weaknesses when addressing the imbalance problem, having a bias towards the majority classes. This is due to the mechanisms inside these classifiers that benefit the right classification of the majority classes to achieve a better accuracy metric. The methods proposed for dealing with the imbalance problem can be categorized into three groups [19]: data

I. Rojas et al. (Eds.): IWANN 2015, Part I, LNCS 9094, pp. 361–373, 2015.
DOI: 10.1007/978-3-319-19258-1_31

sampling, algorithmic modification and cost-sensitive learning. The last approach considers higher costs for the misclassification of examples of the majority classes trying to minimize higher cost errors.

Radial Basis Function Networks (RBFNs) are one of the most important Artificial Neural Network (ANN) paradigms in the machine learning field. An RBFN is a feed-forward ANN with a single layer of hidden units, called radial basis functions (RBFs) [5]. The overall efficiency of RBFNs has been proved in many areas [6] such as as classification [25,26], regression [2,8] or time series forecasting [10,22].

Authors have developed an algorithm for the cooperative-competitive design of Radial Basis Functions Networks, CO^2RBFN [25], that has been successfully used in classification tasks. The base version of CO^2RBFN has also obtained outstanding results in imbalanced problems [23]. Cost-sensitivity is one of the main methodologies addressing the imbalance problem. In this paper CO^2RBFN-CS, a first cost-sensitive version of CO^2RBFN, is presented. CO^2RBFN-CS is based on introducing a cost in the training stage of CO^2RBFN, modifying the learning rate of the weights. This costs depends on the right (or wrong) classification of the instance, the type of class (majority or minority) and the imbalance ratio of the data set. The experimentation has been performed in two stages, firstly CO^2RBFN-CS is compared with its base version, CO^2RBFN, and secondly, CO^2RBFN-CS is compared with existing cost-sensitive methods referenced in the bibliography.

The text is organized as follows. In Section 2, the imbalanced data sets environment is described. Section 3 introduces the cost sensitive learning. CO^2RBFN, is described in Section 4 and the new method proposal, CO^2RBFN-CS, is explained in Section 5. Finally the analysis of the experiments and the conclusions are shown in Sections 6 and 7, respectively.

2 Imbalanced Data Sets

In classification tasks, we are dealing with imbalanced data sets when the data do not have an equitable distribution among the different classes of the problem. Particularly, with a data set of only two classes, the imbalance problem occurs when one class is represented by a large number of examples, while the other is represented by only a few [7].

Classification in an imbalanced data sets environment is a difficult and important task. It is difficult, due to standard classifier algorithms having a bias towards the majority (negative) class and usually the minority (positive) class represents the concept of interest. On the other hand, many real applications present an imbalanced data sets scenario.

In the specialized literature, imbalanced data sets are managed as a whole or are characterized according to their degree of imbalance using the imbalance ratio (IR), which is defined as the ratio of the number of instances of the majority class and the minority class.

Imbalanced data sets can be categorized taken into account the IR level [11]: data sets with a low imbalance when the instances of the positive class are between 10 and 40% of the total instances (IR lower than 9) and data sets with a high imbalance where there are no more than 10% of positive instances in the whole data set compared to the negative ones (IR higher than 9).

In order to deal with this problem, some approaches have been proposed. These approaches can be categorized into three major groups [19]:

- Data sampling: In which the training instances are modified in such a way to produce a more or less balanced class distribution that allow classifiers to perform in a similar manner to standard classification.
- Algorithmic modification: This procedure is oriented towards the adaptation of base learning methods to be more attuned to class imbalance issues.
- Cost-sensitive learning: This type of solutions incorporate approaches at the data level, at the algorithmic level, or at both levels combined, considering higher costs for the misclassification of examples of the positive class with respect to the negative class, and therefore, trying to minimize higher cost errors.

Into classification task, accuracy is the most used empirical measure, but it is not adequate to the evaluation in imbalanced domains due to the fact that it does not distinguish between the number of correct labels of different classes, which in the framework of imbalanced problems may lead to erroneous conclusions. One of the most used metrics in imbalanced data sets is the geometric mean (GM) of the true rates [3], defined as:

$$GM = \sqrt{\frac{TP}{TP + FN} \frac{TN}{FP + TN}} \tag{1}$$

where TP, TN, FP and FN stand for True Positives, True Negatives, False Positives and False Negatives respectively. This metric attempts to maximize the accuracy of each one of the two classes with a good balance.

3 Cost-Sensitive Learning

Generally speaking, cost-sensitive learning states different costs of misclassification with respect to the existing classes in a problem. With this aim, a cost matrix $C(i, j)$ defines the penalties of classifying examples of one class i as a different one j. Typically, these misclassification costs can be heuristically established or extracted from domain experts. In binary imbalance problems $C(+, -)$ is the cost of misclassifying a positive (class) instance as the negative and $C(-, +)$ is the opposite cost. As mentioned, in this kind of problem, it is more interesting to obtain a correct classification of the positive instance and therefore $C(+, -) > C(-, +)$. According to [19], cost-sensitive learning algorithms can be divided into three categories:

- Direct methods: These algorithms introduce misclassification costs into the learning algorithms. In decision tree algorithms the cost information is used

to choose the best attribute to split the data [18] or to determine whether a subtree should be pruned [4]. In [30] an approach based on genetic algorithms that incorporates misclassification costs in the fitness function is presented. [16] modifies the learning weight algorithm for training RBFNs in order to introduce cost sensitivity. In [28] and [31] a cost-sensitive learning for SVM is implemented, the main idea being to assign a larger penalty value to false negatives than false positives.
– Meta-learning: In this category an additional stage is introduced where the training data are pre-processed or the output is post-processed, remaining the original learning algorithm unmodified. Furthermore, cost-sensitive algorithms can be classified into thresholding and sampling strategies. In thresholding strategies instances are assigned to the class with minimum expected costs as in [9]. In [36] a threshold-moving method is defined with the aim of moving the output threshold toward inexpensive classes such that examples with higher costs become harder to be misclassified. On the other hand, the sampling methodology is based on undersampling/oversampling [35] or assigning instance weights [29]. In [29] an instance-weighting method is defined in order to design cost-sensitive trees.

4 CO^{2}RBFN: An Evolutionary Cooperative-Competitive Hybrid Algorithm for RBFN Design

Radial Basis Function Networks (RBFNs) are one of the most important Artificial Neural Network (ANN) paradigms in the machine design field. An RBFN is a feed-forward ANN with a single layer of hidden units, called radial basis functions (RBFs) [5,21].

From a structural point of view, an RBFN is a feed-forward neural network with three layers: an input layer with n nodes, a hidden layer with m neurons or RBFs, and an output layer (Figure 1).

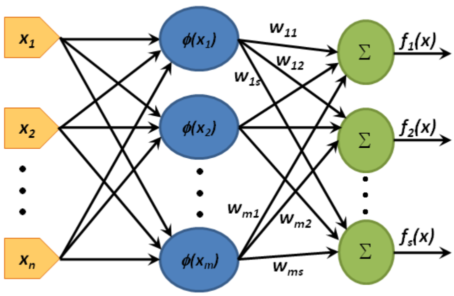

Fig. 1. RBFN Topology

The m neurons of the hidden layer are activated by a radially-symmetric basis function, $\phi_i : \mathbb{R}^n \rightarrow \mathbb{R}$, which can be defined in several ways [27], the Gaussian function being the most widely used (Equation 2):

$$\phi_i(\boldsymbol{x}) = \phi_i(e^{-(\|\boldsymbol{x}-\boldsymbol{c_i}\|/d_i)^2}) \tag{2}$$

where $\boldsymbol{c_i} \in R^n$ is the center of basis function ϕ_i, $d_i \in R$ is the width (radius), and $\|\|$ is typically the Euclidean norm on R^n. This expression is the one used in this paper as the Radial Basis Function (RBF). The output node implements the following function, where weights w_{ij} show the contribution of an RBF to the output node (Equation 3):

$$f_j(\boldsymbol{x}) = \sum_{i=1}^{m} w_{ij}\phi_i(\boldsymbol{x}) \tag{3}$$

The objective of any RBFN design process [6] is to determine centers, widths and the linear output weights connecting the RBFs to the output neuron layer.

An important paradigm for RBFN design is Evolutionary Computation [14]. There are different proposals in this area with different scheme representations: Pittsburgh [15], where each individual is a whole RBFN, and cooperative-competitive [32], where an individual represents a single RBF.

CO^{2}RBFN [25] is an evolutionary cooperative-competitive hybrid algorithm for the design of RBFNs. The network represents the entire population and each individual is represented by a neuron or RBF. The fitness of each individual is known as credit assignment and it is calculated by using three factors: the RBF contribution to the network output, the error in the basis function radius, and the degree of overlapping among RBFs.

The three parameters used for credit assignment are given as input of a Fuzzy Rule-Based System that determines which of the four evolutive operator must be applied to the individual.

The main steps of CO^{2}RBFN, explained in the following subsections, are shown in the pseudocode, in Algorithm 1. For a wider explanation of the algorithm see reference [25].

Algorithm 1. Main steps of CO^{2}RBFN

```
1. Initialize RBFN
2. Train RBFN
3. Evaluate RBFs
4. Apply operators to RBFs
5. Substitute the eliminated RBFs
6. Select the best RBFs
7. If the stop condition is not verified go to step 2
```

In the RBFN initialization step, to define the initial network, a specified number m of neurons (i.e. the size of population) is randomly allocated to the different patterns of the training set. To do so, each RBF center, $\boldsymbol{c}_i$, is randomly assigned to a pattern of the training set. The RBF widths, d_i, will be set to half of the average distance between the centers. Finally, the RBF weights, w_{ij}, are set to zero.

In the RBFN training step, LMS training algorithm is used.

For the RBF evaluation, a credit assignment mechanism is required in order to evaluate the role of each RBF ϕ_i in the cooperative-competitive environment. For an RBF, three parameters:

- The contribution of the RBF, a_i, is determined by considering the weight and the number of patterns of the training set inside its width.
- The error measure, e_i, for each RBF, is obtained by counting the wrongly classified patterns inside its radius.
- The overlapping of the RBF with respect to the other RBFs.

In CO^{2}RBFN four operators have been defined in order to be applied to the RBFs:

- Operator *Remove*: eliminates an RBF.
- Operator *Random Mutation*: randomly modifies the coordinates of the center and the width of an RBF.
- Operator *Biased Mutation*: modifies the width and the coordinates of the center using local information of the RBF environment.
- Operator *Null*: in this case all the parameters of the RBF are maintained.

The operators are applied to the whole population of RBFs. The probability of choosing an operator is determined by means of a Mandani-type fuzzy rule based system [20].

The inputs of this system are parameters a_i, e_i and o_i used for defining the credit assignment of the RBF ϕ_i. These inputs are considered linguistic variables va_i, ve_i and vo_i. The outputs, p_{remove}, p_{rm}, p_{bm} and p_{null}, represent the probability of applying Remove, Random Mutation, Biased Mutation and Null operators, respectively.

The rule base system aims to evolve RBFs with a good behavior (high contribution, low error and low overlapping) and to eliminate RBFs with a bad behavior (low contribution, high error and high overlapping).

In the step of introduction of new RBFs, the eliminated RBFs are substituted by new RBFs. The new RBF is located in the center of the area with maximum error or in a randomly chosen pattern with a probability of 0.5 respectively.

The width of the new RBF will be set to the average of the RBFs in the population plus half of the minimum distance to the nearest RBF. Its weights are set to zero.

The replacement scheme determines which new RBFs (obtained before the mutation) will be included in the new population. To do so, the role of the mutated RBF in the net was compared with the original one to determine the RBF with the best behavior in order to include it in the population.

5 CO^2RBFN-CS: CO^2RBFN with Cost-Sensitivity

As mentioned in previous sections, approaches that deal with imbalanced data sets can be categorized into three categories: Data sampling, Algorithmic modification and Cost-sensitive. This last approach can incorporate modifications at the data level, at the algorithmic level, or at both levels combined, in order to introduce higher costs for the misclassification classes.

A first base to design CO^2RBFN-CS, the adaptation of CO^2RBFN for addressing imbalance problems, is that the Positive class is poorly represented (with only a few instances) into an imbalanced data set. CO^2RBFN learns the classes of the data set using the Least Mean Square (LMS, [33]) and, in some way, each class is learned from its instances. Nevertheless, this can be a difficult task for the Positive class which can have few instances in an imbalanced data set.

Although the classical LMS algorithm, as it was seen in [24], obtains good performance in imbalanced data sets scenarios, the objective of this paper is to modify it in order to achieve better results. LMS is a local weights training algorithm that uses the gradient descent technique. This technique exploits the local information that can be obtained from the behavior of each RBFs. Equation 4 shows the update of the weights.

$$\overline{w}_{k+1} = \overline{w}_k + \alpha \frac{e_k \phi_i(\overline{x}_k)}{|\phi_i(\overline{x}_k)|^2} \tag{4}$$

where k is the number of iterations, $\overline{w}_{k+1}$ is the next value of the weight vector, $\overline{w}_k$ is the present value of the weight vector and $\overline{x}_k$ is the value of the actual input pattern vector.

The present linear error, e_k, is defined as the difference between the desired output and the output network before adaptation. The α value is the *speed of learning*, which measures the size of the adjustment to be made. The choice of α controls stability and speed of convergence.

The challenge in the new proposal is to redefine the LMS algorithm incorporating cost-sensitivity. Few methods have been proposed in this line, the most similar is [16] where cost-sensitivity is introduced in the weights training algorithm, and although it is a gradient descent based algorithm, it does not use the classic LMS algorithm. Moreover [16] is designed to solve an specific problem.

Focusing on our proposal, as can be seen in equations 5, a new variable *cost* is introduced in the LMS algorithm. The equations 6, 7 are used to explain this cost, instead of a cost matrix, because it depends on three aspects: the IR of the data set, the classified class (Positive or Negative) and the current success on the classification of the instance. Obviously *cost* depends on the classified class and in this way its value is mainly modified when a pattern of the minority class is learned. The IR of the dataset is taken into account because when this ratio is high, that means there are few positive patterns, it is necessary to increase the learning of rate of these patterns in order to improve the accuracy. Finally, if an instance is being wrongly classified its learning rate will be increased.

In summary, the *cost* is higher if the IR is higher than 9, the classified class is the minority one and the instance is being wrongly classified by the model.

On the other hand the *cost* is lower if the IR is lower than 9, the classified class is the majority one and the instance is being correctly classified by the model.

$$\overline{w}_{k+1} = \overline{w}_k + cost * \alpha \frac{e_k \phi_i(\overline{x}_k)}{|\phi_i(\overline{x}_k)|^2} \tag{5}$$

$$cost = \begin{cases} csuccess & \text{well classified minority class pattern} \\ cerror & \text{wrong classified minority class pattern} \\ 1 & \text{majority class pattern} \end{cases} \tag{6}$$

$$\begin{aligned} IR < 9 &\begin{cases} csuccess = 1 \\ cerror = 2 \end{cases} \\ IR >= 9 &\begin{cases} csuccess = 2 \\ cerror = 4 \end{cases} \end{aligned} \tag{7}$$

6 Experimentation and Results

The methodology to test CO^{2}RBFN-CS has consisted in comparing it with its base version CO^{2}RBFN and after that, with other more mature cost-sensitive methods. With this objective, different data sets have been chosen from the KEEL data set repository [1]. Table 1 summarizes the data employed in this study and shows, for each data set, the number of examples (#Ex.), number of attributes (#Atts.), class name of each class (minority and majority), class attribute distribution and IR. This table is ordered by the IR, from low to highly imbalanced data sets. Half of the data sets have an IR lower than 9 and the remaining ones have an IR higher than 9.

Table 1. Description for imbalanced data sets

Data sets	#Ex.	#Atts.	Class(min., maj.)	%Class(min., maj.)	IR
glass0	214	9	(build-win-float-proc, remainder)	(32.71, 67.29)	2.06
haberman	306	3	(Die, Survive)	(27.42, 73.58)	2.68
glass0123vs456	214	9	(non-window glass, remainder)	(23.83, 76.17)	3.19
vehicle0	846	18	(Van, remainder)	(23.64, 76.36)	3.23
ecoli2	336	7	(pp, remainder)	(15.48, 84.52)	5.46
pageblocks0	5472	10	(remainder, text)	(10.23, 89.77)	8.77
wowel0	988	13	(hid, remainder)	(9.01, 90.99)	10.10
shuttle0vs4	1829	9	(Rad Flow, Bypass)	(6.72, 93.28)	13.87
yeast2vs8	482	8	(pox, cyt)	(4.15, 95.85)	23.10
yeast1289vs7	947	8	(vac, nuc, cyt, pox, erl)	(3.17, 96.83)	30.56
yeast5	1484	8	(me1, remainder)	(2.96, 97.04)	32.78
yeast6	1484	8	(exc, remainder)	(2.49, 97.51)	39.15

With these data sets, a typical experimental framework has been established with 5-fold cross validation and five repetitions for obtaining the results.

Firstly the new proposal, CO^2RBFN-CS, and the base version, CO^2RBFN, are compared. The same configuration parameters are set up for the two algorithms: 200 iterations are established for the main loop and the number of individuals or RBFs is set to five.

In Table 2 the average correct classification rates, using GM measures, obtained by both versions (the original CO^2RBFN and the CO^2RBFN-CS) are shown.

Table 2. Average GM test results

Data sets	CO^2RBFN-CS	CO^2RBFN
glass0	78.314	75.693
haberman	63.206	61.209
glass0123vs456	93.363	92.269
vehicle0	90.776	89.116
ecoli2	92.072	92.024
pageblocks0	87.832	86.073
vowel0	91.717	87.026
shuttlec0vsc4	99.665	99.671
yeast2vs8	76.291	71.883
yeast1289vs7	68.788	55.190
yeast5	94.803	94.121
yeast6	87.840	83.269

Table 3. Wilcoxon test. R^+ corresponds to new proposal, CO^2RBFN-CS, and R^- to CO^2RBFN

R^+	R^-	p-value
77.0	1.0	0.002526

If these first results are analyzed (Table 2), it can be observed that CO^2RBFN-CS outperforms CO^2RBFN in 11 of the 12 tested data sets.

In order to detect significant differences, the Wilcoxon signed-ranks test [34] is applied to compare the results of each version. Table 3 shows the result of the test, in this table R+ corresponds to CO^2RBFN-CS and and R- to CO^2RBFN. The p-value obtained is very low, indicating that the null hypothesis of equality of means is rejected with a high confidence level. Therefore, CO^2RBFN-CS outperforms CO^2RBFN with significant difference.

Focusing the analysis on the IR of the data sets, we can observe that when the IR is higher than 9 the difference between the methods grows. As example, for yeast6 CO^2RBFN-CS beats CO^2RBFN by 4 points and for yeast1289vs7 by 13 points. These results indicate that the cost-sensitive methodology proposed in this paper is even more suitable when the IR grows.

Now, CO^2RBFN-CS is compared with other more mature cost-sensitive methods, specifically with the cost-sensitive methods implemented in Keel, these are:

Table 4. Average GM test results

Data sets	CO^2RBFN-CS	SVM-CS-RBF	C4.5-CS	NN-CS
glass0	78.314	78.799	78.450	59.969
haberman	63.206	52.040	50.040	61.541
glass0123vs456	93.363	93.624	87.459	91.191
vehicle0	90.776	40.095	92.850	65.587
ecoli2	92.072	90.764	72.263	67.314
pageblocks0	87.832	60.895	97.957	70.522
vowel0	91.717	100.000	88.702	66.653
shuttlec0vsc4	99.665	95.003	99.971	84.736
yeast2vs8	76.291	75.269	84.698	65.190
yeast1289vs7	68.788	69.324	62.057	43.873
yeast5	94.803	96.931	93.098	62.166
yeast6	87.840	87.406	85.337	55.909

Table 5. Friedman ranking test

Algorithm	Ranking
CO^2RBFN-CS	1.833
SVM-CS-RBF	2.167
C4.5-CS	2.417
NN-CS	3.583

Table 6. Hochberg post hoc test

i	algorithm	$z = (R_0 - R_i)/SE$	p	Hochberg
3	NN-CS	3.320392	0.000899	0.016667
2	C4.5-CS	1.106797	0.268382	0.025
1	SVM-CS-RBF	0.632456	0.527089	0.05

– SVM-CS-RBF [28,31]: a cost-sensitive learning for SVM, with RBF Kernel, that assigns a larger penalty value to false negatives than false positives.
– C4.5-CS [29]: an instance-weighting method to design cost-sensitive trees.
– NN-CS [36]: a threshold-moving method that aims to move the output threshold toward inexpensive classes such that examples with higher costs become harder to be misclassified.

Further information can be found in section 3 and in the referenced bibliography. The parameters established for these methods are the default ones configured in Keel. The results obtained for the different methods are shown in Table 4.

With the objective of carrying out an adequate multiple comparison, as recommended in [13], the statistical analysis was performed in two steps. Firstly, the Friedman test [12] is used to rank the methods, and to establish if any statistical differences exist. In the ranking of the methods obtained by this test, Table 5, CO^{2}RBFN-CS achieves the best (lowest) value. The p-value computed by Friedman test is 0.0058, which indicates significant differences among the methods. Secondly, in order to elucidate these significant differences between CO^{2}RBFN-CS and the remaining methods, the Hochberg test [17] has been applied, obtaining the results showed in Table 6. Taking CO^{2}RBFN-CS as control algorithm, because is the best algorithm according to Friedman test, it can be highlighted that CO^{2}RBFN-CS obtains significant differences with respect to a more mature method such as NN-CS, and a notable p-value with respect to C4.5-CS. In any case it is remarkable that this first approach introducing cost-sensitivity into a cooperative-competitive design of RBFNs, CO^{2}RBFN-CS, has achieved similar or even better results than other more tested methods.

7 Conclusions

Currently, the data sets that represent an important number of real applications are imbalanced. These imbalance problems are addressed with different methods which can be categorized into three groups. data sampling, algorithmic modification and cost-sensitive learning. Cost-sensitive learning can hybridize both data sampling and algorithmic modification in order to incorporate higher costs for the misclassification of examples, trying to minimize higher cost errors.

In this paper CO^{2}RBFN-CS, a first approach in the cooperative-competitive design of RBFNs with cost-sensitivity, is presented. This proposal is based on introducing a cost parameter into the training weights stage of the algorithm in order to modifying its learning rate. This costs depends on the right (or wrong) classification of the instance, the type of class (majority or minority) and the imbalance ratio of the data set.

The results show that CO^{2}RBFN-CS outperforms CO^{2}RBFN in 11 of the 12 tested data sets and after applying Wilcoxon's test, significant differences have been achieved. Also, it can be observed that CO^{2}RBFN-CS even works better when the IR of the data set is higher.

When CO^{2}RBFN-CS is compared with more mature cost-sensitive methods in the bibliography, it can be highlighted that CO^{2}RBFN-CS achieves the first position in the ranking of the Friedman and, even, outperforms the NN-CS method with significant differences. In summary, this first approach in the cost-sensitive cooperative-competitive design of RBFNs, CO^{2}RBFN-CS, has obtained promising results, similar or even higher than other more mature methods. As future research lines the proposed equation will be modified adjusting the value of the parameters or introducing new ones related with the neighborhood of positive patterns.

Acknowledgments. F. Charte is supported by the Spanish Ministry of Education under the F.P.U. National Program (Ref. AP2010-0068). This paper is partially supported by the Spanish Ministry of Science and Technology under the Project TIN 2012-33856, FEDER founds.

References

1. Alcalá-Fdez, J., Fernández, A., Luengo, J., Derrac, J., García, S., Sánchez, L., Herrera, F.: Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework. J. of Mult.-Valued Logic & Soft Computing 17, 255–287 (2011)
2. Alexandridis, A., Chondrodima, E., Sarimveis, H.: Radial basis function network training using a nonsymmetric partition of the input space and particle swarm optimization. IEEE Transactions on Neural Networks and Learning Systems **24**(2), 219–230 (2013)
3. Barandela, R., Sánchez, J.S., García, V., Rangel, E.: Strategies for learning in class imbalance problems. Pattern Recognition **36**(3), 849–851 (2003)
4. Bradford, J.P., Kunz, C., Kohavi, R., Brunk, C., Brodley, C.E.: Pruning decision trees with misclassification costs. In: Proceedings of the the 10th European Conference on Machine Learning (ECML 1998), pp. 131–136 (1998)
5. Broomhead, D., Lowe, D.: Multivariable functional interpolation and adaptive networks. Complex Systems **2**, 321–355 (1988)
6. Buchtala, O., Klimek, M., Sick, B.: Evolutionary optimization of radial basis function classifiers for data mining applications. IEEE Transactions on System, Man, and Cybernetics, B **35**(5), 928–947 (2005)
7. Chawla, N.V., Japkowicz, N., Kolcz, A.: Special issue on learning from imbalanced data sets. SIGKDD Explorations Newsletter **6**(1), 1–6 (2004)
8. Chen, H., Kong, L., Leng, W.: Numerical solution of pdes via integrated radial basis function networks with adaptive training algorithm. Applied Soft Computing **11**, 856–860 (2011)
9. Domingos, P.: Metacost: a general method for making classifiers cost sensitive. In: Proceedings of the 5th International Conference on Knowledge Discovery and Data Mining, pp. 155–164 (1999)
10. Du, H., Zhang, N.: Time series prediction using evolving radial basis function networks with new enconding scheme. Neurocomputing **71**, 1388–1400 (2008)
11. Fernández, A., del Jesus, M.J., Herrera, F.: Hierarchical fuzzy rule based classification system with genetic rule selection for imbalanced data-set. International Journal of Approximate Reasoning **50**, 561–577 (2009)
12. Friedman, M.: The use of ranks to avoid the assumption of normality implicit in the analysis of variance. Journal of the American Statistical Association **32**, 675–701 (1937)
13. García, S., Fernández, A., Luengo, J., Herrera, F.: Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: Experimental analysis of power. Information Sciences **180**, 2044–2064 (2010)
14. Goldberg, D.: Genetic Algorithms in Search, Optimization, and Machine Learning. Addison-Wesley, Reading (1989)
15. Harpham, C., Dawson, C.W., Brown, M.R.: A review of genetic algorithms applied to training radial basis function networks. Neural Computing and Applications **13**, 193–201 (2004)
16. He, Z.M.: Cost-sensitive steganalysis with stochastic sensitvity and cost sensitive training error. Proceedings of the International Conference on Machine Learning and Cybernetics **1**, 349–354 (2012)
17. Hochberg, Y.: A sharper bonferroni procedure for multiple tests of significance. Biometrika **75**(4), 800–802 (1988)

18. C. X. Ling, Q. Yang, J. Wang, and S. Zhang. Decision trees with minimal costs. In: Proceedings of the 21st International Conference on Machine Learning (ICML 2004), vol. 69, pp. 544–551 (2004)
19. López, V., Fernández, A., García, S., Palade, V., Herrera, F.: An insight into classiffcation with imbalanced data: Empirical results and current trends on using data intrinsic characteristics. Information Sciences **250**, 113–141 (2013)
20. Mandani, E., Assilian, S.: An experiment in linguistic synthesis with a fuzzy logic controller. International Journal of Man-Machine Studies **7**(1), 1–13 (1975)
21. Moody, J., Darken, C.J.: Fast learning in networks of locally-tuned processing units. Neural Computation **1**, 281–294 (1989)
22. Niu, H.L., Wang, J.: Financial time series prediction by a random data-time effective RBF neural network. Soft Computing **18**(3), 497–508 (2014)
23. Pérez-Godoy, M.D., Fernández, A., Rivera, A.J., del Jesus, M.J.: Analysis of an evolutionary RBFN design algorithm, CO^{2}RBFN, for imbalanced data sets. Pattern Recognition Letters **31**(15), 2375–2388 (2010)
24. Pérez-Godoy, M.D., Rivera, A.J., Carmona, C.J., del Jesus, M.J.: Training algorithms for radial basis function networks to tackle learning processes with imbalanced data-sets. Applied Soft Computing **25**, 26–39 (2014)
25. Pérez-Godoy, M.D., Rivera, A.J., del Jesus, M.J., Berlanga, F.J.: CO^{2}RBFN: An evolutionary cooperative-competitive RBFN design algorithm for classification problems. Soft Computing **14**(9), 953–971 (2010)
26. Qasem, S.N., Shamsuddin, S.M.: Memetic elitist pareto differential evolution algorithm based radial basis function networks for classification problems. Applied Soft Computing **11**(8), 5565–5581 (2011)
27. Rojas, I., Valenzuela, O., Prieto, A.: Statistical analysis of the main parameters in the definition of radial basis function networks. LNCS **1240**, 882–891 (1997)
28. Tang, Y., Zhang, Y.-Q., Chawla, N.V., Krasser, S.: Svms modeling for highly imbalanced classification. IEEE Transactions on Systems Man and Cybernetics. PART B. Cybernetics **39**(1), 281–288 (2009)
29. Ting, K.M.: An instance-weighting method to induce cost-sensitive trees. IEEE Transactions on Knowledge and Data Engineering **14**(3), 659–665 (2002)
30. Turney, P.D.: Cost-sensitive classification: empirical evaluation of a hybrid genetic decision tree induction algorithm. Journal of Artificial Intelligence Research **2**, 369–409 (1995)
31. Veropoulos, K., Cristianini, N., Campbell, C.: Controlling the sensitivity of support vector machines. In: Proceedings of the International Joint Conference on Artificial Intelligence (IJCAI 1999), pp. 55–60 (1999)
32. Whitehead, B., Choate, T.: Cooperative-competitive genetic evolution of radial basis function centers and widths for time series prediction. IEEE Transactions on Neural Networks **7**(4), 869–880 (1996)
33. Widrow, B., Lehr, M.A.: 30 years of adaptive neural networks: perceptron, madaline and backpropagation. Proceedings of the IEEE **78**(9), 1415–1442 (1990)
34. Wilcoxon, F.: Individual comparisons by ranking methods. Biometrics **1**, 80–83 (1945)
35. Zadrozny, B., Langford, J., Abe, N.: Costsensitive learning by costproportionate-example weighting. In: Proceedings of the 3rd IEEE International Conference on Data Mining (ICDM 2003), pp. 435–442 (2003)
36. Zhou, Z.H., Liu, X.Y.: Training cost-sensitive neural networks with methods addressing the class imbalance problem. IEEE Transactions on Knowledge Data Engineering **18**(1), 63–77 (2006)