Global Sensitivity Estimates for Neural Network Classifiers

Francisco Fernández-Navarro, *Member, IEEE*, Mariano Carbonero-Ruz, David Becerra Alonso, and Mercedes Torres-Jiménez

***Abstract*— Artificial neural networks (ANNs) have traditionally been seen as black-box models, because, although they are able to find "hidden" relations between inputs and outputs with a high approximation capacity, their structure seldom provides any insights on the structure of the functions being approximated. Several research papers have tried to debunk the black-box nature of ANNs, since it limits the potential use of ANNs in many research areas. This paper is framed in this context and proposes a methodology to determine the individual and collective effects of the input variables on the outputs for classification problems based on the ANOVA-functional decomposition. The method is applied after the training phase of the ANN and allows researchers to rank the input variables according to their importance in the variance of the ANN output. The computation of the sensitivity indices for product unit neural networks is straightforward as those indices can be calculated analytically by evaluating the integrals in the ANOVA decomposition. Unfortunately, the sensitivity indices associated with ANNs based on sigmoidal basis functions or radial basis functions cannot be calculated analytically. In this paper, the indices for those kinds of ANNs are proposed to be estimated by the (quasi-) Monte Carlo method.**

***Index Terms*— Feedforward neural networks, functional decomposition, global sensitivity analysis (GSA), multiclassification, product unit neural networks (PUNNs).**

I. Introduction

THERE have been many methods, within the supervised learning realm, that have shown a high degree of prediction and accuracy. Artificial neural networks (ANNs) have been among the many successful proposals in this field. Despite this, many researchers still refuse to use them, due to their "black-box" nature [56]. Aiming to readdress this issue, many ANN rule extraction approaches have been developed to transform the numerical weights in an ANN into symbolic rules [1]–[3].

The issue of whether ANNs are true black boxes has previously been covered in the literature [4]–[6], [56], but the question remains open. Alongside this line of research, comparative studies between ANNs and more "transparent" methods have been made [7]–[9]. The conclusions yield acceptance on the fact that, black box or not, ANNs are efficient on a wide range of topics [10]–[12]. While the interpretation of ANN weights remains inconclusive, work has turned to what ANNs do, as opposed to how ANNs turn out to be weightwise [13], [14].

Sensitivity analysis (SA) is one of the methods used to understand the internal doings of ANNs [15], [16]. It acknowledges the black-box nature of ANNs by studying the direct effects that inputs have on classification or regression outputs. On the basis of working with a reliable ANN, performing SA can rank attributes in terms of their relevance. But the reverse process has also been proposed: using SA to choose better ANNs [17], or to prune a good ANN to simplify or improve it [18], [19]. Another application of SA is incremental learning [20].

SA has been traditionally addressed in the field of ANNs using the local sensitivity analysis (LSA) techniques [21], [22]. LSA involves taking the partial derivative of the ANN output with respect to their input variables ($\{x_1, x_2, \ldots, x_K\}$, where K is the number of input variables). It is important to note that in LSA, it is still assumed that the ANN is efficiently trained. The most common methods within the framework of LSA are perturbation methods [23]. The main contribution of these methodologies is the idea of perturbing inputs by a small amount around some fixed point aiming to compute the effect of those perturbations on the ANN outputs. Perturbation methods do not make any assumptions on the local minima from the Jacobian. Some applications show how, depending on the data, it suffices with 1-D attribute-by-attribute perturbation [21], [24], [25]; however, this approach does not fit every kind of data (especially when sensitivity depends on linear or nonlinear combinations of multiple input variables). Perturbing pairs of attributes also proved useful in some cases [22]. The main limitations of LSA are as follows.

1) They just analyze the behavior of the ANN in the immediate region around the optimum determined during the training stage. Furthermore, LSA methods are valid only for the small regions of uncertainty [26].
2) LSA techniques only consider changes to one or just a few parameters at a time, with all other parameters fixed. This approach is not very suitable to real-world problems. The study of the interactions between parameters is often crucial to understand the system at hand.

Manuscript received January 24, 2016; revised April 27, 2016; accepted August 1, 2016. Date of publication August 19, 2016; date of current version October 16, 2017. This work was supported in part by the Spanish Ministry of Economy and Competitiveness, FEDER Funds under Project TIN2014-54583-C2-1-R and in part by Junta de Andalucia, Spain, under Project P2011-TIC-7508.

The authors are with the Department of Quantitative Methods, Universidad Loyola Andalucia, 41014 Córdoba, Spain (e-mail: i22fenaf@uco.es, fafernandez@uloyola.es; mariano@uloyola.es; dbecerra@uloyola.es; mtorres@uloyola.es).

This paper has supplementary downloadable material available at http://ieeexplore.ieee.org., provided by the author.

Color versions of one or more of the figures in this paper are available online at http://ieeexplore.ieee.org.

Digital Object Identifier 10.1109/TNNLS.2016.2598657

When using global sensitivity analysis (GSA) techniques [27], all inputs are varied simultaneously over their entire input space, typically using a sampling-based approach [28]. The effects on the output of individual inputs and interactions between inputs are considered. The first method implemented under the umbrella of GSA techniques and ANNs was the Gram–Schmidt orthogonalization [29], [30]. This method ranks the input variables of a model, linear with respect to its parameters, using the correlation ratio between the input and the output. This approach is computationally expensive and may need a large amount of memory space [31]. From a different perspective, a new approach based on the variance decomposition of the ANN output and the extended Fourier amplitude sensitivity test (EFAST) [32] has been recently proposed, aiming to overcome the limitations of the above-mentioned GSA methods [31]. The methodology does not rely on any assumptions made on the Jacobian or the Hessian. It is independent of the model and is applied after the training stage. Despite all these good properties, this methodology inherits all the problems of the FAST analysis, and therefore has the following limitations.

1) FAST and its extended version, EFAST, are numerical propagation techniques and, therefore, they do not provide an exact value for their sensitivity indices. In certain industrial applications, this lack of accuracy might not be acceptable.
2) Saltelli *et al*. [32] did further improvements on the FAST method. They worked on the possibility of computing total sensitivity indices for a given input parameter (the fraction of the variance from it and all its interactions). This extended methodology, called EFAST, was the one adopted by Fock [31]. However, FAST and EFAST remain unable to compute sensitivity indices for interactions. As it was previously said, these indices are crucial to real-world problems.

Furthermore, Fock [31] computes the sensitivity indices in classification problems without considering the interaction on the edges between pairs of classes. Motivated by these facts, we propose an alternative way of measuring the sensitivity indices in classification problems based on the Sobol decomposition [33] (a GSA technique based on the ANOVA decomposition) that overcomes all the above-mentioned limitations. The main commonalities among our approach and Fock's [31] are that both the methods are applied after the training stage, and both rank the inputs according to their importance in the variance of the model output. The computation of the sensitivity indices for product unit neural networks (PUNNs) [34] is straightforward, since its functional form allows performing an SA after a fast analytical approach. Unfortunately, the sensitivity indices associated with ANNs, based on sigmoidal basis functions or a radial basis function, cannot be calculated analytically. In this paper, the indices for those kind of ANNs are proposed to be estimated by the (quasi-) Monte Carlo method [27].

Section II presents the variance-based method used. The output function we will use for classification is detailed in Section III-A. A set of prerequisites need to be met for the integrability of our solution. These are specified in Section III-B. Global variance is defined and its computation is explained in Section III-C. Sections III-D–III-F detail the decompositions required to this global variance. After defining and explaining the ANNs that we will use (Sections IV and V), a series of experiments will be outlined in Section VI. This paper finishes with the conclusions and the supporting material.

II. Variance-Based Methods for Global Sensitivity Analysis: The Sobol Decomposition

In the field of GSA, variance-based methods determine the importance of each input variable considering the variance provided by the parameter to the total variance of the output [33], [35]. Let us suppose that a mathematical model is described by a function $Y = f(\mathbf{x})$, where Y is the univariate model output and $\mathbf{x} = (x_1, \ldots, x_K)$ is a pattern inside a K-dimensional space. Furthermore, it will be assumed that the inputs are independently and uniformly distributed within the unit hypercube, i.e., $x_i \in [0, 1]$ for $i = 1, 2, \ldots, K$[1] and that $f(\mathbf{x})$ is an integrable function. If these assumptions are met, $f(\mathbf{x})$ could be decomposed as

$$f(\mathbf{x}) = f_0 + \sum_{I} f_I(\mathbf{x}_I) = f_0 + \sum_{i=1}^{K} f_i(x_i) + \sum_{i<j}^{K} f_{ij}(x_i, x_j) + \cdots + f_{12\ldots K} \quad (1)$$

where the total number of summands in the equation is 2^K, I is a nonempty subset of the set $(\{1, 2, \ldots, K\})$ $(I \subset \{1, 2, \ldots, K\})$, f_0 is a constant, x_I is the subvector of $\mathbf{x}$ with subindices defined in I, f_i is a function of x_i, f_{ij} a function of x_i and x_j and so on. Thus, f_i could be interpreted as the effect on Y by varying x_i alone, while f_{ij} represents the effect on Y by varying x_i and x_j simultaneously in addition to the effect of their individual variations. Analogously, f_I is the effect of varying the elements included in x_I simultaneously and added to the effect of the individual variations of the components included in the set I.

These functions could be recurrently obtained as follows:

$$f_0 = \int f(\mathbf{x}) d\mathbf{x} \quad (2)$$

$$f_I(x_I) = \int f(\mathbf{x}) dx_{(I)} - \sum_{J \subset I} f_J(x_J) - f_0 \quad (3)$$

where (I) is the complementary of I, and therefore, $dx_{(I)} = \prod_{k \notin I} dx_k$. These functions have the following three important properties.

Property 1: The integrals of the summands $f_I(x_I)$ with respect to any of their "own" variables are zero, that is

$$\int f_I(\mathbf{x}_I) dx_i = 0, \quad \text{if } i \in I. \quad (4)$$

[1]Please note that this does not affect the generality of the method, because any input space can be transformed into this unit hypercube.

Taking into account that x_i is uniformly distributed between 0 and 1, this property could be reinterpreted as

$$E_i(f_I) = 0, \quad \text{if } i \in I \tag{5}$$

where E_i is the expected value with respect to the variable x_i.

Proof: Property 1 is proved by mathematical induction using the cardinal of I, $|I|$. In particular, for $|I| = 1$ and $I = \{x_i\}$

$$\begin{aligned}\int f_i(x_i)dx_i &= \int \left(\int f(\mathbf{x})dx_{(i)} - f_0 \right) dx_i \\ &= \int f(\mathbf{x})d\mathbf{x} - f_0 = 0\end{aligned}$$

where $x_{(i)}$ is the complementary of x_i. We will assume that the formula is true for $|I| < m$. Assuming this, we prove that the formula is true for $|I| = m$

$$\begin{aligned}\int f_I(\mathbf{x}_I)dx_i &= \int \left(\int f(\mathbf{x})dx_{(I)}) \right) dx_i \\ &\quad - \sum_{J \subset I} \int f_J(\mathbf{x}_J)dx_i - f_0 \\ &= \int f(\mathbf{x})dx_{(A)} - \sum_{J \subseteq A} f_J(\mathbf{x}_J) - f_0 \\ &= \int f(\mathbf{x})dx_{(A)} - \sum_{J \subset A} f_J(\mathbf{x}_J) - f_A(\mathbf{x}_A) - f_0 \\ &= 0\end{aligned}$$

where $A = I - \{i\}$ and its complementary $(A) = (I) \cup \{i\}$.

Property 2: The integral of the product of functions with different indices is zero

$$\int f_I(\mathbf{x}_I) f_J(\mathbf{x}_J)d\mathbf{x} = 0. \tag{6}$$

Proof: The fact that $I \neq J$ involves that there is a subindex i, such that $i \in I - J$ and that, therefore, the previously mentioned integral could be rewritten as

$$\int f_I(\mathbf{x}_I) f_J(\mathbf{x}_J)d\mathbf{x} = \int f_J(\mathbf{x}_J) \left(\int f_I(\mathbf{x}_I)dx_i \right) dx_{(i)} = 0.$$

Considering that f_I and f_J are uncorrelated, this property could be statistically expressed as

$$\Sigma_{I,J} = \text{COV}(f_I, f_J) = 0. \tag{7}$$

Property 3: The variance of function f is the sum of variances that corresponds to the indices I, since they are uncorrelated

$$V(f) = \sum_I V(f_I) \tag{8}$$

$$V(f) = \int (f(\mathbf{x}) - f_0)^2 d\mathbf{x} = \int f^2(\mathbf{x})d\mathbf{x} - f_0^2 \tag{9}$$

$$V(f_I) = \int f_I^2(\mathbf{x}_I)d\mathbf{x}_I. \tag{10}$$

It is also important to mention that (1) could be written according to the number of variables included in the terms of the decomposition

$$f(\mathbf{x}) = f_0 + \sum_{i=1}^{K} \sum_{|I|=i} f_I(\mathbf{x}_I). \tag{11}$$

Hence, the decomposition of order r is defined as

$$f(\mathbf{x}) = f_0 + \sum_{i=1}^{r} \sum_{|I|=i} f_I(\mathbf{x}_I) + R(\mathbf{x}) \tag{12}$$

where $R(\mathbf{x})$ encompasses the residual part of the decomposition of order r and

$$R(\mathbf{x}) = \sum_{i=r+1}^{K} \sum_{|I|=i} f_I(\mathbf{x}_I). \tag{13}$$

Under this formulation of functional decomposition of order r, the variance is defined as

$$V(f) = \sum_{i=1}^{r} V_i(f) + V_R(f) \tag{14}$$

$$V_i(f) = \sum_{|I|=i} V(f_I). \tag{15}$$

Finally, Sobol [35] defined global sensitivity indices as the ratio given by

$$S_{f_I(\mathbf{x}_I)} = \frac{V(f_I)}{V(f)} \tag{16}$$

where all $S_{f_I(\mathbf{x}_I)} \geq 0$ and $\sum_{i=1}^{K} \sum_{|I|=i} S_{f_I(\mathbf{x}_I)} = 1$.

The calculation of all the sensitivity indices requires the evaluation of 2^K integrals. For this reason, Sobol [35] introduced sensitivity indices for the subsets of variables (as it has been described in this paper) and total sensitivity indices. The total sensitivity indices measure the contribution to the output variance of each of the variables of the model, including all variances caused by its interactions, of any order, with any other input variables. It is defined as

$$S_{T_{f_I(\mathbf{x}_I)}} = 1 - \frac{V(f_{(I)})}{V(f)}. \tag{17}$$

III. Global Sensitivity Analysis on Neural Network Classifiers

The goal of this section is to mathematically formulate the sensitivity of the classification function with respect to the input variables using a global approach based on the Sobol decomposition. In Section III-A, the neural network model adopted and the classification function proposed are described. In Section III-B, the requirements of the methodology are listed. The computation of the global variance is detailed in Section III-C. The mathematical expressions to obtain the functional decompositions of orders 1, 2, and r are, first, analyzed in Sections III-D–III-F, respectively.

A. Classification Function Proposed for SFLNs

For the decomposition, we have considered a single hidden layer feedforward neural networks (SLFNs) with Q output nodes and L *basis functions*. The transfer function of all output nodes is the identity function, and their activation functions are defined as

$$f_q(\mathbf{x}) = \sum_{l=1}^{L} \beta_{ql} B_l(\mathbf{x}, \mathbf{w}_l) \tag{18}$$

where $q \in \{1, 2, \ldots, Q\}$, $\mathbf{x} = (x_1, \ldots, x_K) \in \mathbb{R}^K$ is the input vector of the model, $f_q(\mathbf{x})$ is the qth output of the neural network, β_{ql} is the weight of the connection between the lth *basis functions* and the qth output node, $\boldsymbol{\beta}_q = (\beta_{q1}, \beta_{q2}, \ldots, \beta_{qL}) \in \mathbb{R}^L$, $\boldsymbol{\beta} = \{\boldsymbol{\beta}_q\}_{q=1}^{Q} \in \mathbb{R}^Q \times \mathbb{R}^L$, $B_l(\mathbf{x}, \mathbf{w}_l)$ is the output of the lth *basis function* in the hidden layer, $\mathbf{w}_l = (w_{l1}, w_{l2}, \ldots, w_{lK}) \in \mathbb{R}^K$ is the vector of connections between the lth *basis function* and the input variables, and $\mathbf{W} = \{\mathbf{w}_l\}_{l=1}^{L} \in \mathbb{R}^L \times \mathbb{R}^K$.

The class predicted by the neural network corresponds to the node in the output layer with the largest output value. Hence, the optimum classification rule is

$$\mathbf{x} \to \mathcal{C}_{q'} \quad \text{if } \max\{f_1(\mathbf{x}), \ldots, f_Q(\mathbf{x})\} = f_{q'}(\mathbf{x}). \tag{19}$$

As proposed by Sobol [35], the sensitivity of a multivariate system with respect to the input variables $(x_1, x_2, \ldots, x_K)$ could be determined by the individual analysis of the sensitivities of the functions that compose the multivariate system with respect to the inputs. However, this approach is not directly applicable to classification problems, because the final classification decision is not taken by individually analyzing the functions composing the system. Instead, the system is analyzed as a whole (assigning the class label of a certain pattern to the label associated with the function with the greatest output value). For example, if we modify the value of x_k, a possible scenario could be that all the outputs of the functions change after the modification of the variable, but the function with the greatest output before the modification remains the dominant function. In this case, the functions of the system are sensitive to x_k unlike the classification function.

Motivated by the above-mentioned fact, the differences among the classification functions are proposed as a framework to determine the sensitivity of the overall classification function with respect to the input variables. If x_k is modified, the output functions will most likely also change. However, these changes do not modify the final classification: the differences among the functions involved in the system are what really change the classification decision. This leads us to define the following triangular matrix of differences:

$$\Delta(p, q) = f_p(\mathbf{x}) - f_q(\mathbf{x}) \tag{20}$$

$$= \sum_{l=1}^{L} (\beta_{pl} - \beta_{ql}) B_l(\mathbf{x}, \mathbf{w}_l) \tag{21}$$

where $p < q$ and $\binom{Q-1}{2}$ is the total number of functions to be analyzed. Furthermore, under this formulation, a variable could be relevant to a pair of classes and irrelevant to a different pair of classes. This happens unlike to the work of Fock [31], where the relevance of each variable is studied without considering its impact on each pair of classes. In the experimentation will be shown how this formulation could be considered as a subset of our formulation. The importance of this new formulation is shown in Fig. 1. As can be seen in Fig. 1, the variable x_1 is the most important one to discriminate among classes $\mathcal{C}_1$ and $\mathcal{C}_2$ and among $\mathcal{C}_1$ and $\mathcal{C}_3$; however, the classifier should focus more on x_2 to discriminate a pattern among $\mathcal{C}_2$ and $\mathcal{C}_3$. Although the proposed methodology has been specifically developed for classification problems, it could also be applied to function approximation and time series prediction problems. In those cases, the analysis should be performed directly on the outputs of the ANN, as it was done in [31].

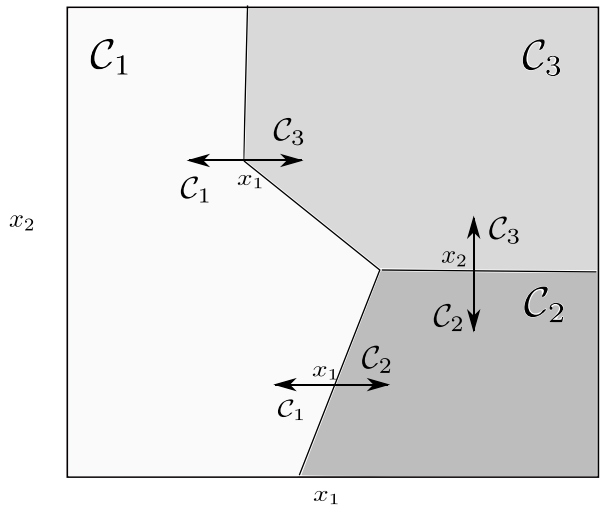

Fig. 1. Illustrative example to justify the construction of the $\Delta(p, q)$ function.

B. Requirements

The two conditions to be imposed on the classification function $\Delta(p, q)$ are follows.

1) The input variables should be defined within the interval [0, 1]. This incurs no loss of generality, because any input space can be transformed into this unit hypercube using the following formula:

$$x_{nk}^{\star} = \frac{x_{nk} - m_k}{M_k - m_k} \tag{22}$$

where $n = 1, 2, \ldots, N$ is the nth pattern in the training set, N is the total number of training patterns, $x_{nk}^{\star}$ is the scaled value of the nth pattern in its kth dimension, x_{nk} is the original value of the nth pattern in its kth dimension, and M_k and m_k are the maximum and minimum values in the kth dimension, respectively. The main problem of this approach is that the *basis functions* will be constrained to a specific interval depending on the interval defined by matrix $\mathbf{W}$. Furthermore, some *basis functions* require input values on a different scale. For that reason, we will assume the existence of an internal normalization function, $n(\mathbf{x}^{\star})$, which will be included on each *basis function* in the model. Thus, the $\Delta(p, q)$ function is now defined as

$$\Delta(p, q) = \sum_{l=1}^{L} (\beta_{pl} - \beta_{ql}) B_l(n(\mathbf{x}^{\star}), \mathbf{w}_l). \tag{23}$$

2) The classification function must be square-integrable. In other words, the integral of $\Delta^2(p, q)$ over the entire space must be finite. This condition could also be satisfied by choosing a square-integrable transfer function over all the input spaces for the *basis function* of the SLFN.

C. Computation of the Global Variance

Following the guidelines described in Section II, the global variance is equal to the variance of the function to be decomposed:

$$V(\Delta(p,q)) = V\left(\sum_{l=1}^{L}(\beta_{pl}-\beta_{ql})B_l(n(\mathbf{x}^\star),\mathbf{w}_l)\right). \quad (24)$$

The mathematical function to be analyzed is a weighted linear combination of L random variables, and therefore, its variance could be computed as

$$V(\Delta(p,q)) = (\boldsymbol{\beta}_p-\boldsymbol{\beta}_q)^T\boldsymbol{\Sigma}(\mathbf{B})(\boldsymbol{\beta}_p-\boldsymbol{\beta}_q) \quad (25)$$

where $\boldsymbol{\Sigma}(\mathbf{B}) \in \mathbb{R}^L \times \mathbb{R}^L$ is the covariant matrix of the *basis functions*, defined as

$$\boldsymbol{\Sigma}(\mathbf{B})_{lh} = \int B_l(n(\mathbf{x}^\star),\mathbf{w}_l)B_h(n(\mathbf{x}^\star),\mathbf{w}_h)d\mathbf{x} - \int B_l(n(\mathbf{x}^\star),\mathbf{w}_l)d\mathbf{x}\int B_h(n(\mathbf{x}^\star),\mathbf{w}_h)d\mathbf{x}. \quad (26)$$

As can be seen, the complexity of the computation of the variance depends exclusively on the complexity of integrating the *basis function* selected.

D. Mathematical Formulation of the First-Order Decomposition

The general equation associated with the first-order decomposition is

$$\Delta(p,q) = \Delta(p,q)_0 + \sum_{i=1}^{K}\Delta(p,q)_{x_i} + \Delta(p,q)_R \quad (27)$$

where $\Delta(p,q)_0$ is defined as

$$\Delta(p,q)_0 = \int \Delta(p,q)d\mathbf{x} = \sum_{l=1}^{L}(\beta_{pl}-\beta_{ql})\int B_l(n(\mathbf{x}^\star),\mathbf{w}_l)d\mathbf{x} \quad (28)$$

$\Delta(p,q)_{x_i}$ is defined as

$$\Delta(p,q)_{x_i} = \int \Delta(p,q)dx_{(i)} - \Delta(p,q)_0 = \sum_{l=1}^{L}(\beta_{pl}-\beta_{ql})B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l) - \Delta(p,q)_0 \quad (29)$$

where $B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l)$ is defined as

$$B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l) = \int B_l(n(\mathbf{x}^\star),\mathbf{w}_l)dx_{(i)}. \quad (30)$$

Finally, $\Delta(p,q)_R$ is the residual term and includes all the interaction among the different input variables. It is defined as

$$\Delta(p,q)_R = \Delta(p,q) - \Delta(p,q)_0 - \sum_{i=1}^{K}\Delta(p,q)_{x_i}. \quad (31)$$

The partial variances associated with each variable x_i, $V(\Delta(p,q)_{x_i})$, are defined as

$$V(\Delta(p,q)_{x_i}) = (\boldsymbol{\beta}_p-\boldsymbol{\beta}_q)^T\boldsymbol{\Sigma}(\mathbf{B}^i)(\boldsymbol{\beta}_p-\boldsymbol{\beta}_q) \quad (32)$$

where matrix $\boldsymbol{\Sigma}(\mathbf{B}^i) \in \mathbb{R}^L \times \mathbb{R}^L$ is defined as

$$\boldsymbol{\Sigma}(\mathbf{B}^i)_{lh} = \int B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l)B_h^i(n(\mathbf{x}^\star),\mathbf{w}_h)dx_i - \int B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l)dx_i\int B_h^i(n(\mathbf{x}^\star),\mathbf{w}_h)dx_i. \quad (33)$$

The variance associated with the interaction among the input variables, $V(\Delta(p,q)_R)$, could be obtained as

$$V(\Delta(p,q)_R) = V(\Delta(p,q)) - \sum_{i=1}^{K}V(\Delta(p,q)_{x_i}). \quad (34)$$

Hence, the total variance could be decomposed as

$$V(\Delta(p,q)) = V_1(\Delta(p,q)) + V(\Delta(p,q)_R) \quad (35)$$

$$V_1(\Delta(p,q)) = \sum_{i=i}^{K}V(\Delta(p,q)_{x_i}). \quad (36)$$

Finally, the first-order sensitivity indices and the sensitivity of the interactions term are computed as

$$S_{x_i}(p,q) = \frac{V(\Delta(p,q)_{x_i})}{V(\Delta(p,q))} \quad (37)$$

$$S_R(p,q) = \frac{V(\Delta(p,q)_R)}{V(\Delta(p,q))}. \quad (38)$$

E. Mathematical Formulation of the Second-Order Decomposition

The second-order functions of the Sobol decomposition are defined as

$$\Delta(p,q)_{x_ix_j} = \sum_{l=1}^{L}(\beta_{pl}-\beta_{ql})B_l^{ij}(n(\mathbf{x}^\star),\mathbf{w}_l) - \Delta(p,q)_0 \quad (39)$$

where $B_l^{ij}(n(\mathbf{x}^\star),\mathbf{w}_l)$ is defined as

$$B_l^{ij}(n(\mathbf{x}^\star),\mathbf{w}_l) = \int B_l(n(\mathbf{x}^\star),\mathbf{w}_l)dx_{(ij)} - B_l^i(n(\mathbf{x}^\star),\mathbf{w}_l) - B_l^j(n(\mathbf{x}^\star),\mathbf{w}_l). \quad (40)$$

The variance associated with the second-order terms is

$$V(\Delta(p,q)_{x_ix_j}) = (\boldsymbol{\beta}_p-\boldsymbol{\beta}_q)^T\boldsymbol{\Sigma}(\mathbf{B}^{ij})(\boldsymbol{\beta}_p-\boldsymbol{\beta}_q) \quad (41)$$

where the matrix $\boldsymbol{\Sigma}(\mathbf{B}^{ij}) \in \mathbb{R}^L \times \mathbb{R}^L$ is defined as

$$\boldsymbol{\Sigma}(\mathbf{B}^{ij})_{lh} = \int B_l^{ij}(n(\mathbf{x}^\star),\mathbf{w}_l)B_h^{ij}(n(\mathbf{x}^\star),\mathbf{w}_h)dx_idx_j - \left(\int B_l^{ij}(n(\mathbf{x}^\star),\mathbf{w}_l)dx_idx_j \times \int B_h^{ij}(n(\mathbf{x}^\star),\mathbf{w}_h)dx_idx_j\right). \quad (42)$$

Hence, the total variance could be decomposed as

$$V(\Delta(p,q)) = V_1(\Delta(p,q)) + V_2(\Delta(p,q)) + V(\Delta(p,q)_R) \quad (43)$$

$$V_1(\Delta(p,q)) = \sum_{i=i}^{K}V(\Delta(p,q)_{x_i}) \quad (44)$$

$$V_2(\Delta(p,q)) = \sum_{i<j}V(\Delta(p,q)_{x_ix_j}) \quad (45)$$

and finally, the variance associated with the residual term, $V(\Delta(p,q)_R)$, could be obtained as

$$V(\Delta(p,q)_R) = V(\Delta(p,q)) - V_1(\Delta(p,q)) - V_2(\Delta(p,q)). \tag{46}$$

F. Mathematical Formulation of the Decomposition of Order r

In this section, the Sobol decomposition is generalized to the order r. To that aim, the set $I_r = \{i_1, \ldots, i_r\}$ will be used. In this case, the order r functions of the Sobol decomposition are defined as

$$\Delta(p,q)_{I_r} = \sum_{l=1}^{L} (\beta_{pl} - \beta_{ql}) B_l^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_l) - \Delta(p,q)_0 \tag{47}$$

where $B_l^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_l)$ is defined as

$$B_l^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_l) = \int B_l(n(\mathbf{x}^\star), \mathbf{w}_l) dx_{(I_r)} - \sum_{J \subset I_r} B_l^J(n(\mathbf{x}^\star), \mathbf{w}_l). \tag{48}$$

The variance associated is computed as

$$V(\Delta(p,q)_{I_r}) = (\boldsymbol{\beta}_p - \boldsymbol{\beta}_q)^T \boldsymbol{\Sigma}(\mathbf{B}^{I_r})(\boldsymbol{\beta}_p - \boldsymbol{\beta}_q) \tag{49}$$

where the matrix $\boldsymbol{\Sigma}(\mathbf{B}^{I_r}) \in \mathbb{R}^L \times \mathbb{R}^L$ is defined as

$$\boldsymbol{\Sigma}(\mathbf{B}^{I_r})_{lh} = \int B_l^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_l) B_h^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_h) dx_i dx_{I_r} - \left(\int B_l^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_l) dx_{I_r} \int B_h^{I_r}(n(\mathbf{x}^\star), \mathbf{w}_h) dx_{I_r} \right). \tag{50}$$

Hence, the total variance could be decomposed as

$$V(\Delta(p,q)) = \sum_{i=1}^{r} V_i(\Delta(p,q)) + V(\Delta(p,q)_R) \tag{51}$$

$$V_i(\Delta(p,q)) = \sum_{|I|=i} V(\Delta(p,q)_I) \tag{52}$$

and finally, the variance associated with the residual term, $V(\Delta(p,q)_R)$, could be obtained as

$$V(\Delta(p,q)_R) = V(\Delta(p,q)) - \sum_{i=1}^{r} V_i(\Delta(p,q)). \tag{53}$$

IV. ANNs Models With Analytical Solution: Product Unit Basis Function

The main limitation of the previously described methodology is the requirement of the Σ matrices to be analytically computable. In order to meet this constraint, the iterated integral of the *basis function* selected should exist. Unfortunately, commonly used transition functions adopted by the machine learning community (such as the sigmoidal, radial basis, or the arctan) do not satisfy this constraint. These facts motivate the use of the PU *basis function*, since its iterated integral exists in all the cases, independently of the number of input variables.

PUs are an alternative to sigmoidal transfer functions, and are based on multiplicative neurons instead of additive ones. PU-based neural networks (PUNNs) have several advantages, including increased information capacity and the ability to express strong interactions between input variables [34]. Despite these advantages, PUNNs have a major handicap: they have more local minima and a greater probability of finding themselves trapped in them [36]. Another problem with PUNNs is the very steep output functions it can return. Small changes in the exponents can cause large changes in the total error surface. For that reason, their parameters have traditionally been estimated by using global optimization algorithms [36], [37]. The main advantages and limitations of PUNNs were already addressed in [36], where a pruning algorithm for PUNNs was developed, based on the variance nullity algorithm proposed in [19]. The mathematical expression of the lth PU *basis function* is

$$B_l(n(\mathbf{x}^\star), \mathbf{w}_l) = \prod_{k=1}^{K} \left(n\left(x_k^\star\right)\right)^{w_{lk}} \tag{54}$$

where w_{lk} is the weight of the connection between the input variable l and the hidden neuron j.

In PUNNs, the input data are scaled to positive values to avoid complex numbers as the outputs of the basis function. Two ranges have been traditionally used: $[1, 2]$ [37] and $[0.1, 0.9]$ [38]. In both the cases, the lower bound is chosen to avoid input values near zero that could produce very large outputs for negative exponents. The upper bound is traditionally chosen to avoid dramatic changes in the outputs of the network when there are weights with large values (especially in the exponents).

One problem associated with this scaling is that the training input patterns, $\mathbf{X}$, are defined in $[0, 1]^N \times [0, 1]^K$. However, the *basis functions* are expected to take input values with different data ranges. This problem could be solved in a straightforward way by defining the internal normalization function $n(\mathbf{x}^\star)$ as

$$n(\mathbf{x}^\star) = \left(a + \left((b-a) \times x_k^\star\right)\right) \tag{55}$$

where a and b are the lowest and greatest values of the normalization range. Hence, the *basis functions* are defined as

$$B_l(n(\mathbf{x}^\star), \mathbf{w}_l) = \prod_{k=1}^{K} \left(a + \left((b-a) \times x_k^\star\right)\right)^{w_{lk}}. \tag{56}$$

V. ANNs Models With Numerical Solution: Sigmoidal or Radial Basis Function

The applicability of the Sobol-based GSA is related to the possibility of computing the multidimensional integrals reported in Section III. Unfortunately, the sensitivity indices associated with ANNs based on sigmoidal basis functions [also called multilayer perceptron (MLP)] or radial basis functions [radial basis function neural networks (RBFNN)] [39], [40] cannot be calculated analytically by evaluating the integrals in the decomposition. In those cases, the sensitivity indices could be estimated using the (quasi-) Monte Carlo method [27].

Below, the procedure followed in this paper (which was previously proposed in [27]) for computing the full set of the

first-order and total-effect indices for an MLP or an RBFNN model of K input variables is described.

1) An $(N, 2K)$ matrix of "quasi-random" numbers is generated. Two matrices of data (A and B) are defined, each one with half of the base sample N. As suggested in [27], the substitution of random sequences for low-discrepancy sequences (also known as "quasi-random" sequences) leads to an improvement in the efficiency of the sensitivity estimators. Low-discrepancy sequences are the numbers that are more evenly distributed, within a given volume, than pseudorandom numbers. This property helps Monte Carlo simulations in achieving faster convergence and better accuracy than simulations using conventional pseudorandom numbers. Thus, the Sobol' sets (low-discrepancy sequences) [33] are used then as the standard sampling technique in this paper

$$A = \begin{pmatrix} x_1^1 & x_2^1 & \dots & x_i^1 & \dots & x_K^1 \\ x_1^2 & x_2^2 & \dots & x_i^2 & \dots & x_K^2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ x_1^{N-1} & x_2^{N-1} & \dots & x_i^{N-1} & \dots & x_K^{N-1} \\ x_1^N & x_2^N & \dots & x_i^N & \dots & x_K^N \end{pmatrix}$$

$$B = \begin{pmatrix} x_{K+1}^1 & x_{K+2}^1 & \dots & x_{K+i}^1 & \dots & x_{2K}^1 \\ x_{K+1}^2 & x_{K+2}^2 & \dots & x_{K+i}^2 & \dots & x_{2K}^2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ x_{K+1}^{N-1} & x_{K+2}^{N-1} & \dots & x_{K+i}^{N-1} & \dots & x_{2K}^{N-1} \\ x_{K+1}^N & x_{K+2}^N & \dots & x_{K+i}^N & \dots & x_{2K}^N \end{pmatrix}.$$

2) A matrix C_i is defined. It is made of all columns of B except the ith column, which is taken from A

$$C = \begin{pmatrix} x_{K+1}^1 & x_{K+2}^1 & \dots & x_i^1 & \dots & x_{2K}^1 \\ x_{K+1}^2 & x_{K+2}^2 & \dots & x_i^2 & \dots & x_{2K}^2 \\ \dots & \dots & \dots & \dots & \dots & \dots \\ x_{K+1}^{N-1} & x_{K+2}^{N-1} & \dots & x_i^{N-1} & \dots & x_{2K}^{N-1} \\ x_{K+1}^N & x_{K+2}^N & \dots & x_i^N & \dots & x_{2K}^N \end{pmatrix}.$$

3) The model output for the sample matrices A, B, and C_i is computed. Three vectors of model outputs, with dimension $N \times 1$ per each pair of classes p and q, are obtained

$$y_A = f_p(A) - f_q(A) \tag{57}$$

$$y_B = f_p(B) - f_q(B) \tag{58}$$

$$y_{C_i} = f_p(C_i) - f_q(C_i). \tag{59}$$

The first-order sensitivity indices are then defined as

$$S_{x_i}(p, q) = \frac{y_A \cdot y_{C_i} - f_0^2}{y_A \cdot y_A - f_0^2} \tag{60}$$

where

$$f_0 = \left(\frac{1}{N} \sum_{n=1}^{N} y_A^n \right) \tag{61}$$

is the mean and $(\cdot)$ represents the scalar product of the vectors. Finally, the total-effect indices are estimated as

$$S_{T_{x_i}}(p, q) = \frac{y_B \cdot y_{C_i} - f_0^2}{y_A \cdot y_A - f_0^2}. \tag{62}$$

TABLE I

CHARACTERISTICS OF THE 15 DATA SETS USED FOR THE EXPERIMENTS: NUMBER OF INSTANCES (#PAT.), TOTAL NUMBER OF INPUTS (#ATTR.), NUMBER OF CLASSES (#CLASSES), AND PER-CLASS DISTRIBUTION OF THE INSTANCES

Dataset	#Pat.	#Attr.	#Classes	Class distribution
hepatitis (HE)	155	19	2	$(32, 123)$
breast-cancer (BC)	286	15	2	$(201, 85)$
haberman (HA)	306	3	2	$(225, 81)$
liver (LI)	345	6	2	$(145, 200)$
diabetes (DI)	768	8	2	$(500, 268)$
card (CA)	690	51	2	$(307, 383)$
contact-lenses (CL)	24	6	3	$(15, 5, 4)$
pasture (PA)	36	24	3	$(12, 12, 12)$
squash-stored (SS)	52	51	3	$(23, 21, 8)$
squash-unstored (SU)	52	52	3	$(24, 24, 4)$
tae (TA)	151	54	3	$(49, 50, 52)$
newthyroid (NE)	215	5	3	$(30, 150, 35)$
balance-scale (BS)	625	4	3	$(288, 49, 288)$
lymph (LY)	148	37	4	$(2, 81, 61, 4)$
vehicle (VE)	946	18	4	(199, 212, 218, 218)

VI. EXPERIMENTS

This section presents the design of the experimental study followed in this paper (Section VI-A), the results obtained for the data sets considered (Section VI-B), and an application of the proposed methodologies to a real-world classification problem (Section VI-C). It is important to clarify that the goal of this section is not to test the classification accuracy of PUNNs or MLPs but to illustrate the utility of the SA method proposed. For that reason, their SA indices will be discussed, thus omitting the classification accuracies of each model.

The source code developed along with the examples shown in this paper has been included in the Supplementary Material Web site, which is hosted on www.uco.es/ayrna/GSA-ANN.

A. Experimental Design

The experiments proposed will now be specified. These include the data sets considered, the ANN models analyzed, and the way their parameters have been estimated. The metric used to evaluate the degree of agreement between different rankings of variables for the different SA methods is also discussed.

1) Data Sets Selected: Table I shows the characteristics of the 15 data sets, including the number of patterns, attributes and classes, and also the class distribution (number of patterns per class). The publicly available classification data sets were obtained from benchmark repositories (UCI [41] and `mldata.org` [42]). The selected data sets include six binary problems and nine multiclass problems and present different numbers of instances, features, and classes (see Table I).

In all the experiments, we have adopted a tenfold cross-validation model, with ten repetitions per each fold, i.e., we have split the data set randomly into ten folds, each one containing the 10% of the patterns of the data set. The stability of each sensitivity methods is checked analyzing the 100 models obtained per data set (tenfold with ten repetitions per fold). The 100 training sessions allowed us to draw the standard error, which gives an indication of the stability of the method.

Finally, it is also important to mention that all nominal attributes were transformed into as many binary attributes as the number of categories. In addition, all the data sets were property standardized, considering only the training set to obtain the mean and standard deviation for each variable.

2) Neural Networks Models: Two types of neural network models were analyzed in the experimental part: PUNNs and MLPs. The ANN models were obtained using the neural net evolutionary programming (NNEP) software package [43], [44], which is an extension of the JCLEC framework (http://jclec.sourceforge.net/) [45]. This software package was selected as it allows us to train both the types of ANNs using the same algorithmic procedure.

The parameter values used for training the models were as follows: to start processing data, each of the input variables was scaled within the interval $[0, 1]$, and after that, the basis functions were internally normalized to the following ranges: $[1, 2]$ for PUNNs and $[-2, 2]$ for MLPs. Weights were assigned using a uniform distribution defined in the interval $[-5, 5]$ for connections between the input layer and the hidden layer. Values range within $[-10, 10]$ for connections between the hidden layer and the output layer. The maximum and minimum number of basis functions in the hidden layer is in the interval $[10, 20]$. The size of the population was set to 100. For the structural mutation, the number of nodes that can be added or removed was within the [1, 2] interval, and the number of connections to add or delete in the hidden and the output layer during structural mutations was within the [1, 7] interval. The stopping criterion 500 generations. Those parameter values were chosen after carefully reviewing the literature about the NNEP framework. Hence, the parameters values suggested by the authors for each neural network model [37], [43], [44] have been used.

3) Sensitivity Analysis Methods: The methodologies proposed, the Sobol method for PUNNs (S-PUNN) and the Sobol method for MLPs (S-MLP), have been evaluated comparing their results to the results of the most promising approaches used in LSA and GSA in the ANNs literature, more concretely as follows.

1) *Local Sensitivity Methods:*
   a) *Weights Product (WP Method) [13]:* The sensitivity indices are obtained after computing the product of the raw input-hidden and hidden-output connection weights between each input neuron and output neuron and sums the products across all hidden neurons.
   b) *Garson's Algorithm [Weights Garson (WG) method] [6]:* This method partitions the hidden-output connection weights into components associated with each input neuron using absolute values of connection weights. Concretely, the sensitivity of the ith input variable with respect to the qth output is defined as

$$S_{x_i} = \frac{\sum_{l=1}^{L} \left(\left(|w_{li}| / \sum_{k=1}^{K} |w_{lk}| \right) |\beta_{ql}| \right)}{\sum_{r=1}^{K} \left(\sum_{l=1}^{L} \left(\left(|w_{lr}| / \sum_{k=1}^{K} |w_{lk}| \right) |\beta_{ql}| \right) \right)}. \tag{63}$$

   c) *Partial Derivatives Method (PaD Method) [17]:* In this method, the relative contribution of each variable on a specific output is determined by computing the sum of the squares of the partial derivatives obtained per input variable.

2) *Global Sensitivity Methods:*
   a) *EFAST [31]:* This method is so far the first and only global sensitivity method, which has been applied to ANNs. The relevance of each variable is determined by analyzing the Fourier decomposition of the variance of each output node.

TABLE II
DIFFERENT SA METHODS CONSIDERED IN THE EXPERIMENTS

Abbr.	Short description
	Local Sensitivity Methods
WP-MLP	Weights Products method for MLPs. [13]
WG-MLP	Weights Garson method for MLPs. [47]
PaD-MLP	Partial Derivatives method for MLPs. [48]
	Global Sensitivity Methods
EFAST-MLP	Extended Fourier Amplitude Sensitivity Test for MLPs [31]
S-PUNN	Sobol method using PUNNs.
S-MLP	Sobol method using MLPs.

The number of samples considered for the methods based on Monte Carlo simulations was 4000. For the state-of-the-art methods, the base classification model considered was the MLP ANN. Note that those methods were originally tested considering, specifically, the architecture and characteristics of this type of network. For the sake of clarity, Table II includes the whole list of SA methods grouped by families with the abbreviations used during the experimental part and a short description.

4) Metric Considered: The Kendall Tau-b, τ_b, was used to evaluate the similarity of the different orderings of input variables when ranked by each SA method [46]. The Kendall Tau-b coefficient is defined as

$$\tau_b = \frac{n_c - n_d}{\sqrt{(n_0 - n_1)(n_0 - n_2)}} \tag{64}$$

where n_c is the number of concordant pairs, n_d the number of discordant pairs, $n_0 = K(K-1)/2$, $n_1 = \sum_i t_i(t_i - 1)/2$, t_i is the number of tied values in the ith group of ties for the first sensitivity algorithm, $n_2 = \sum_j u_j(u_j - 1)/2$ and u_j is the number of tied values in the jth group of ties for the second sensitivity algorithm.

In particular, we have evaluated the degree of agreement of each one of the rankings provided by the different SA methods with respect to the S-MLP method (considered as the reference method). As previously mentioned, all the SA methods have been evaluated with the MLP model except the S-PUNN method. Hence, we aimed not only to evaluate the degree of concordance in the ranking of the first-order input variables of the state-of-the-art methods when compared with the S-MLP but also to analyze the extent to which PUNNs and MLPs networks do their classifications considering the same set of important variables.

To ascertain the statistical significance of the differences between the rankings compared, we have tested the null

TABLE III
NUMBER OF TIMES EACH ONE OF THE COMPARISON METHODS OBTAINED A τ_b VALUE SIGNIFICANTLY DIFFERENT FROM ZERO WHEN COMPARING THEIR RANKINGS WITH THE RANKINGS PROVIDED BY THE S-MLP METHOD

	WP-MLP	WG-MLP	PaD-MLP	EFAST-MLP	S-PUNN
			$\Delta(\mathcal{C}_1, \mathcal{C}_2)$		
HE	72	3	12	89	87
BC	95	29	14	97	91
HA	33	45	29	100	77
LI	93	90	96	98	56
DI	91	34	41	99	94
CA	81	40	55	90	33
			$\Delta(\mathcal{C}_1, \mathcal{C}_3)$		
CL	21	3	68	91	93
PA	70	67	83	100	88
SS	50	70	81	94	44
SU	79	60	62	90	32
TA	31	4	5	83	77
NE	23	19	2	96	84
BS	3	21	4	99	53
			$\Delta(\mathcal{C}_2, \mathcal{C}_3)$		
CL	84	6	74	97	92
PA	81	88	40	93	90
SS	42	60	59	88	55
SU	39	69	70	91	39
TA	29	14	8	89	82
NE	8	27	7	93	90
BS	92	41	35	95	70
			$\Delta(\mathcal{C}_1, \mathcal{C}_4)$		
LY	77	12	31	90	93
VE	88	1	17	97	85
			$\Delta(\mathcal{C}_2, \mathcal{C}_4)$		
LY	83	14	28	94	95
VE	92	6	11	89	64
			$\Delta(\mathcal{C}_3, \mathcal{C}_4)$		
LY	83	23	33	91	92
VE	82	9	8	86	80

$\Delta(\mathcal{C}_i, \mathcal{C}_j) = \Delta(i, j)$
Reference: S-MLP

hypothesis of no correlation ($\tau_b = 0$) against the alternative that there is a nonzero correlation for the 100 models considered in each data set, reporting the number of times that the p-value associated with the τ_b was smaller than 0.05 (Section VI-B).

B. Results

Table III shows the number of times each one of the comparison methods obtained a τ_b value significantly different from zero when compared their rankings with those provided by the S-MLP method. It is interesting to note that the rankings of the EFAST-MLP method and the method proposed were very similar in most cases. Both the methods belong to the same sensitivity algorithms family: GSA methods. Furthermore, the method proposed considered the benchmark when compared with the state-of-the-art methods in the SA literature [47], [48]. The rankings of the S-MLP are also similar to the WP-MLP method, which got the best performance in a synthetic experiment, where different LSA methods for quantifying the sensitivity values of the inputs of an ANN were tested [49].

Below, it is shown that the number of samples used in the experiments is sufficient to provide convergence of the results. The convergence study is performed on the Newthyroid data set and the pairs of classes $\mathcal{C}_1$ and $\mathcal{C}_2$ and their corresponding first-order indices $S_{x_1}, S_{x_2}, S_{x_3}, S_{x_4}, S_{x_5}$. Fig. 2 shows the convergence of these Sobol indices using two sets of samples. Convergence of the indices is observed beyond 4000 samples.

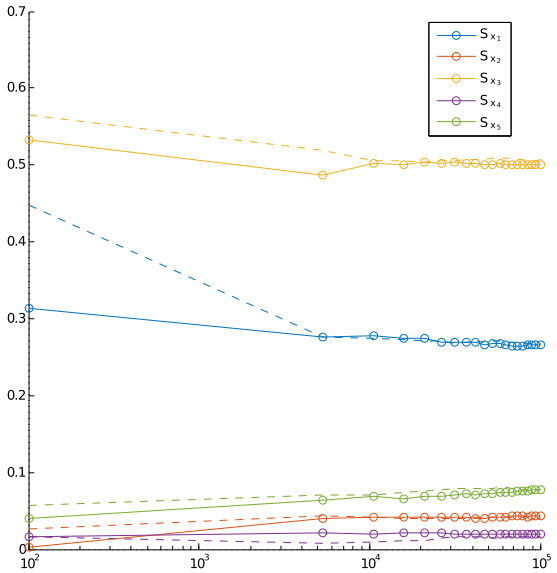

Fig. 2. Convergence of the first-order sensitivity indices for $\Delta(\mathcal{C}_1, \mathcal{C}_2)$ (Newthyroid) with respect to the number of samples using two sets. Solid line: first set. Dashed line: second set.

After analyzing the main commonalities between the different methods with respect to the way the different methodologies rank the variables and studying the convergence of the method proposed, we will highlight the additional sensitivity information that only the proposed method provides. The advantages of the method proposed will be highlighted using Newthyroid once more as the case study.

1) Analysis of the $\Delta(p, q)$ Function: As previously mentioned, Fock [31] studied the relevance of each input variable without considering its impact on each pair of classes but analyzing its impact on each function composing the classification system: $f_1, f_2, \ldots, f_Q$. It is well known that in the ANN literature, traditionally, the last output of the classification system is set to zero ($f_Q = 0$) aiming to reduce the number of parameters to estimate. For example, in Fock [31], the impact of each class in the Iris data set (three-class classification problem) is studied through the analysis of the sensitivity indices in f_1 and f_2 as $f_3 = 0$. This is equivalent to analyze the impact on each input variable for each class and the class of reference ($\mathcal{C}_Q$), because $\Delta(f_{\mathcal{C}_i}, f_{\mathcal{C}_Q}) = f_{\mathcal{C}_i}$ for $i < Q$. Therefore, our proposal could be seen as a generalization of this approach as we also analyze the functions $\Delta(f_{\mathcal{C}_i}, f_{\mathcal{C}_j})$ for $i < j$.

Below, we will show how Δ function could be seen as a generalization of the multivariate approach followed in [31] and [50] (and all the remaining approaches adopted in LSA) in the case of classification problems. For example, Table IV shows how EFAST-MLP and S-MLP provide very similar rankings of variables for functions $\Delta(f_{\mathcal{C}_1}, f_{\mathcal{C}_3})$, $f_{\mathcal{C}_1}$, $\Delta(f_{\mathcal{C}_2}, f_{\mathcal{C}_3})$, and $f_{\mathcal{C}_2}$ as $f_3 = 0$. Our approach allows the exploration of not only the sensitivities of each class with respect to its frontier with class Q ($\mathcal{C}_3$ in the case of the Newthyroid data set) but also to analyze the sensitivities of the input variables between the remaining pairs of classes.

TABLE IV
INDIVIDUAL FIRST-ORDER SENSITIVITY AND TOTAL SENSITIVITY INDICES FOR THE NEWTHYROID DATA SET FOR THE FUNCTION $\Delta(\mathcal{C}_1, \mathcal{C}_2)$

Newthyroid Dataset $\Delta(\mathcal{C}_1, \mathcal{C}_2)$			
First order sensitivities		Total sensitivities	
S_{x_1}	0.265	$S_{T_{x_1}}$	0.296
S_{x_2}	0.043	$S_{T_{x_2}}$	0.064
S_{x_3}	0.501	$S_{T_{x_3}}$	0.517
S_{x_4}	0.016	$S_{T_{x_4}}$	0.079
S_{x_5}	0.077	$S_{T_{x_5}}$	0.152

In the particular case of the Newthyroid data set, the additional information gained with our approach would be the sensitivity information between classes $\mathcal{C}_1$ and $\mathcal{C}_2$ (Table IV first part). The variance of the output function that allows the discrimination between classes $\mathcal{C}_1$ and $\mathcal{C}_2$ is mostly explained by variable x_3. These findings add, for example, to those claimed in [51].

2) Computing Interactions Among Input Variables: The possibility of computing sensitivity indices for interactions is yet another value of the method proposed. Again, the Newthyroid data set for function $\Delta(\mathcal{C}_1, \mathcal{C}_2)$ is employed as the case study. Table IV shows the first-order sensitivity indices for the function previously mentioned. The total amount of variance explained by these first-order sensitivity indices is 90.20%. In fact, more than the 50% of the total variance of the output was explained by variable x_3. The variance explained by the second-order terms was only 8.51% (see 65). The greatest interaction of variables is found between x_5 and x_4 (4.2% of the total variance is explained by this interaction between variables). The method proposed is especially interesting when the key sources of variability have a rather low main effect. In this scenario, it is necessary to evaluate the effect of the interactions among variables. Hence, the method proposed is the only one in the ANN literature so far to provide a numeric value to characterize the system in terms of variance to the interaction terms

$$S_{ij} = \begin{pmatrix} - & - & - & - & - \\ 0.001 & - & - & - & - \\ 3.9E-4 & 0.001 & - & - & - \\ 0.001 & 0.008 & 0.007 & - & - \\ 0.017 & 0.006 & 0.002 & 0.042 & - \end{pmatrix}. \quad (65)$$

This added value can be useful, for instance, in environmental management [52]. The nature of integrated modeling in environmental management is such that inputs are used most of the time in different processes, with their effects often being larger than anticipated due to interactions with other parameters. This gives rise to the need for SA methods to be able to estimate the impact of the interactions in the variance of the output, preferably with the ability to evaluate all parameter interactions.

Sobol's method can also provide insights regarding the total effect of each input variable. As previously mentioned, the total effect of an input includes both the main effect and interaction effects of any dimensionality. Table IV (second part) shows the value associated with the total effect of each input variable. Note that none of the LSA methods are able to return such indices. Finally, Fig. 3 shows also the convergence of the total-effect sensitivity indices.

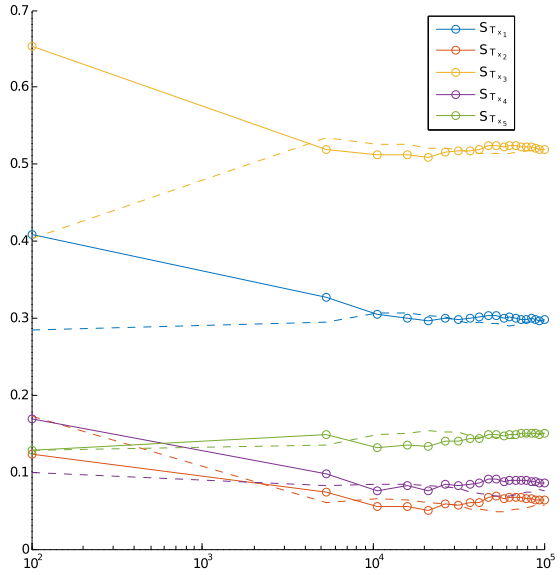

Fig. 3. Convergence of the total sensitivity indices for $\Delta(\mathcal{C}_1, \mathcal{C}_2)$ (Newthyroid) with respect to the number of samples using two sets. Solid line: first set. Dashed line: second set.

TABLE V
COMPUTATIONAL TIME RESULTS FOR ESTIMATING THE FIRST-ORDER SENSITIVITIES OF MODELS WITH DIFFERENT INPUT VARIABLES (RANGING FROM 5 TO 30) OF THE DIFFERENT METHODS

	WP-MLP	WG-MLP	PaD-MLP	EFAST-MLP	S-MLP	S-PUNN
5	0.10	0.10	0.10	0.10	2.10	0.20
10	0.10	0.10	0.10	1.60	4.30	0.80
15	0.10	0.10	0.10	5.30	6.90	3.30
20	0.10	0.10	0.10	11.20	26.10	5.50
25	0.20	0.40	0.20	74.20	126.30	24.40
30	0.30	0.50	0.20	222.10	551.20	33.70

Finally, the computational time required for the different methods in the computation of the first-order sensitivity indices is also analyzed. To evaluate this property of the different methods and consider that they are all independent of the training algorithm, we have decided to randomly create six different MLP and PUNN models for $L = 5$ and $K \in \{5, 10, 15, 20, 25, 30\}$ and assess the computational burden of each sensitivity method in the different ANN models. Table V shows the computational burden of the sensitivity on different methods. In general, the proposed algorithms are more complex than the state-of-the-art sensitivity methods (especially when compared with local methods). However, the additional assets (previously described) provided by the methods proposed here can outtake this high computational cost.

C. Application to a Real-World Classification Problem

The goal of this section is twofold: to analyze the error produced in numerical methods and to illustrate the utility of the method in a real-world problem. For that reason, a research paper where the mathematical function associated with the best PUNN is provided by the authors has been selected. Sensitivity indices of the input variables and the interaction effect for

TABLE VI
INDIVIDUAL FIRST-ORDER SENSITIVITIES AND MSE FOR THE SHEEP CLASSIFICATION DATA SET FOR THE ANALYTICAL AND NUMERICAL SOLUTIONS

	$\Delta(\mathcal{C}_1, \mathcal{C}_2)$								
	First Lactation			Second Lactation			Third Lactation		
	A-GSA	SM-GSA	FAST-GSA	A-GSA	SM-GSA	FAST-GSA	A-GSA	SM-GSA	FAST-GSA
		Mean_{SD}			Mean_{SD}			Mean_{SD}	
S_{x_1}	0.013	$0.000_{0.000}$	0.057	0.006	$0.000_{0.000}$	0.084	0.044	$0.041_{0.012}$	0.048
S_{x_2}	0.010	$4.85E-05_{0.000}$	0.065	0.105	$0.091_{0.021}$	0.087	0.020	$0.017_{0.007}$	0.027
S_{x_3}	1.02E-06	$0.000_{0.000}$	0.022	0.040	$0.137_{0.083}$	0.035	0.290	$0.285_{0.071}$	0.287
S_{x_4}	0.008	$7.4E-04_{0.000}$	0.060	0.158	$0.006_{0.003}$	0.114	0.357	$0.358_{0.080}$	0.353
S_{x_5}	2.59E-04	$0.000_{0.000}$	0.014	0.111	$0.114_{0.072}$	0.087	0.143	$0.140_{0.054}$	0.148
S_{x_6}	0.001	$0.000_{0.000}$	0.016	0.059	$0.048_{0.008}$	0.056	0.092	$0.091_{0.038}$	0.093
S_R	0.965	$0.999_{0.000}$	0.765	0.518	$0.603_{0.097}$	0.537	0.049	$0.066_{0.009}$	0.043
MSE	-	1.35E-04	0.007	-	0.006	0.002	-	4.9E-05	2.17E-05

the classification problem have been calculated. Finally, the results of the analytical method have been compared with the numerical approaches. Please note that this comparison is only possible using PUNNs as their sensitivity indices can be computed analytically or through numerical methods.

The objective of the problem analyzed is the classification of sheep in three different categories (good, $\mathcal{C}_1$, normal, $\mathcal{C}_2$, and bad, $\mathcal{C}_3$) with respect to their milk production in their first three lactations using for this covariable that only involve the first weeks of lactation: maximum production quantity, in liters ($x_1^\star$); maximum production date, in the number of days elapsed since birth ($x_2^\star$); quantity of milk produced, in liters, from birth to the fifth week after birth ($x_3^\star$); quantity of milk produced, in liters, between the fifth and seventh week after birth ($x_4^\star$); quantity of milk produced, in liters, between the seventh and ninth week after birth ($x_5^\star$); quantity of milk produced, in liters, between the 9th and 11th week after birth ($x_6^\star$). This enables the productive capacity of the animal to be identified more rapidly and leads to a faster selection process in determining the best producers. The variable used to establish the productive category of the sheep flock was the production, in liters, obtained during 150 lactation days, because this is the variable used for the genetic selection program for the Manchegan breed. The PUNN mathematical function was extracted from [38].

The analytical methodology, analytical GSA, A-GSA, are compared with the most promising numerical approaches used in GSA:

1) quasi-Monte Carlo GSA (QM-GSA) [50];
2) FAST GSA (FAST-GSA) [53], [54].

The results of the numerical approaches, QM-GSA and FAST-GSA, were obtained using the GSAT framework [55]. In both the cases, the function to be analyzed was exactly the same as the one associated with the analytical method: the $\Delta(p, q)$ function. The QM-GSA method was run 30 times, and its average and standard deviation associated with each experiment were reported as its final performance. The remaining methods considered are deterministic, and therefore, they were run just once in each experiment. The mean squared error (MSE) was used to evaluate the differences among the methodologies

$$\text{MSE} = \frac{1}{(K+1)} \sum_{i=1}^{K+1} (y_i - \hat{y}_i)^2 \tag{66}$$

where y_i is the sensitivity reported by the analytical method in the ith term, $\hat{y}_i$ is the sensitivity reported by the numerical methods (SM-GSA or FAST-GSA), and $K+1$ is the total number of terms included in the decomposition (as only the first-order sensitivities are included in this section).

In this paper, we will focus on the discrimination between the classes good $\mathcal{C}_1$ and normal $\mathcal{C}_2$, as shown in Table VI. In this case, the FAST method outperforms in their approximations to the Sobol on in two out of the three lactations. In particular, FAST obtains good approximations for the second and the third lactation classification problem. For the first lactation, all the methods agree that the variance is explained not by the individual effect of the input variable on the output but by the interaction among all the input variables. The interaction effect is still very high for the second lactation (more than the 51% of the variance of the model output is explained by this effect) but decrease drastically in the third lactation (only the 4.9% of the variance is explained by this interaction effect).

It is important to mention that as it has been claimed in the literature [37], PUNNs report a very promising classification performance especially in scenarios where there is a high interaction among the input variables. This fact is again confirmed in this paper if the variance explained by the interaction effect is analyzed: the PUNN model obtained the best classification performance in the first two lactations but was outperformed by the quadratic discriminant analysis in the third one. These results could be explained if the variance explained by the residual term is investigated. As it has been observed in the experimentation, PUNNs models tend to provide promising results when the value associated with the S_R is high.

The variable-selection process in [38] was carried out according to the following criteria: the maximum production quantity as well as cumulative production until the fifth lactation week were selected because of their strong linear correlation with the dependent variable (both linear correlation coefficients were significant to a 99% confidence level). However, our analysis shows how these two variables are not in the group of the most important variables composed of the variables x_3, x_4, x_5 (especially in the third lactation). Most likely, these linear correlations do not represent the real interaction/dependence among these variables and the dependent one due to the high nonlinear nature of the problem. The maximum production date was included, because it is,

according to the literature, one of the most influential variables in total milk production; although in the study under consideration, the linear correlation between this variable and total production was not significant. In this case, this fact is confirmed in the GSA as x_2 has a reduced impact on the variance of the output except in the second lactation. The remaining independent variables considered (production quantities obtained at different lactation weeks) were selected to reach the goal of estimating the sheep productive category as soon as possible, without waiting until the end of their lactations. As shown in Table VI, good classification models could be obtained if the data only until the ninth week is considered as in all cases the importance of x_6 is very reduced.

VII. Conclusion

This paper explores the possibilities of applying GSA techniques to determine the effects of the input variables on the outputs for classification problems. The technique proposed is based on the ANOVA-functional decomposition and is able to compute sensitivity indices for interactions unlike traditional GSA techniques proposed in the ANN literature. Furthermore, the methodology proposed analyzes the differences among the different classes instead of analyzing classes separately. Thus, under this formulation, a variable could be relevant to a pair of classes and irrelevant to a different pair of classes.

As it was explained in this paper, the traditional ANOVA decomposition proposed by Sobol imposes that the classification function must be square-integrable. To the best of our knowledge, the only well-known ANNs with an associated analytically integrable classification function are the PUNN. Thus, the estimation of the sensitivity indices for PUNNs was done by evaluating the integrals in the decomposition. On the other hand, the estimation of the sensitivity indices for the remaining types of ANNs (nonanalytically integrable functions) was made through numerical methods. More specifically, in this paper, the indices for those kinds of ANNs were computed using the (quasi-) Monte Carlo method.

The experiments presented show how the interaction of multiple attributes can be more relevant than the contribution of any attribute by itself. This confirms the need for a global exploration of the attribute space in terms of the ANN's sensibility. GSA is not just an option, but a necessary extension of LSA, if an adequate analysis of the relevance of attributes is to be addressed.

Future work should find other integrable-function ANNs (or other methods) to test the reliability and consistency of the analytical GSA. A sufficient consistency can also speak for the quality of the classifiers themselves, when the most common GSA outputs are contrasted with those not as common. The attributes ranked by these methods should also be compared with the rankings obtained from other wrappers and filters. Finally, GSA methods, such as this one, could help understand when intrinsic orthogonalities between attributes are taking place and when they do not. This can help with grouping attributes according to how they collectively correlate with classes. These groups, in turn, should further inform about the relevance and/or redundancy of attributes.

References

[1] M. Fukumi and N. Akamatsu, "A new rule extraction method from neural networks," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 6. 1999, pp. 4134–4138.

[2] R. Andrews, J. Diederich, and A. B. Tickle, "Survey and critique of techniques for extracting rules from trained artificial neural networks," *Knowl.-Based Syst.*, vol. 8, no. 6, pp. 373–389, 1995.

[3] J. Chorowski and J. M. Zurada, "Extracting rules from neural networks as decision diagrams," *IEEE Trans. Neural Netw.*, vol. 22, no. 12, pp. 2435–2446, Dec. 2011.

[4] P. D. Wasserman, *Advanced Methods in Neural Computing*. New York, NY, USA: Wiley, 1993.

[5] S. Sestito and T. Dillon, "Knowledge acquisition of conjunctive rules using multilayered neural networks," *Int. J. Intell. Syst.*, vol. 8, no. 7, pp. 779–805, 1993.

[6] D. G. Garson, "Interpreting neural-network connection weights," *AI Expert*, vol. 6, no. 4, pp. 46–51, 1991.

[7] S. Dreiseitl and L. Ohno-Machado, "Logistic regression and artificial neural network classification models: A methodology review," *J. Biomed. Informat.*, vol. 35, nos. 5–6, pp. 352–359, 2002.

[8] M.-S. Duh, A. M. Walker, and J. Z. Ayanian, "Epidemiologic interpretation of artificial neural networks," *Amer. J. Epidemiol.*, vol. 147, no. 12, pp. 1112–1122, 1998.

[9] D. J. Sargent, "Comparison of artificial neural networks with other statistical approaches," *Cancer*, vol. 91, no. S8, pp. 1636–1642, 2001.

[10] J. D. Buckley, "Predicting time-to-relapse in breast cancer using neural networks," Univ. Southern California Los Angeles, DTIC Document, Tech. Rep., 1997.

[11] P. K. Simpson, *Neural Networks Applications*. Piscataway, NJ, USA: IEEE Press, 1997.

[12] F. Fernández-Navarro, P. Campoy-Muñoz, M. de la Paz-Marín, C. Hervás-Martínez, and X. Yao, "Addressing the EU sovereign ratings using an ordinal regression approach," *IEEE Trans. Cybern.*, vol. 43, no. 6, pp. 2228–2240, Dec. 2013.

[13] J. D. Olden and D. A. Jackson, "Illuminating the 'black box': A randomization approach for understanding variable contributions in artificial neural networks," *Ecol. Model.*, vol. 154, nos. 1–2, pp. 135–150, 2002.

[14] B. Cheng and D. M. Titterington, "Neural networks: A review from a statistical perspective," *Statist. Sci.*, vol. 9, no. 1, pp. 2–30, 1994.

[15] S. Hashem, "Sensitivity analysis for feedforward artificial neural networks with differentiable activation functions," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 1. 1992, pp. 419–424.

[16] P. J. G. Lisboa, A. R. Mehridehnavi, and P. A. Martin, "The interpretation of supervised neural networks," in *Proc. Workshop Neural Netw. Appl. Tools*, 1993, pp. 11–17.

[17] Y. Dimopoulos, P. Bourret, and S. Lek, "Use of some sensitivity criteria for choosing networks with good generalization ability," *Neural Process. Lett.*, vol. 2, no. 6, pp. 1–4, 1995.

[18] A. P. Engelbrecht and I. Cloete, "A sensitivity analysis algorithm for pruning feedforward neural networks," in *Proc. IEEE Int. Conf. Neural Netw.*, vol. 2. Jun. 1996, pp. 1274–1277.

[19] A. P. Engelbrecht, "A new pruning heuristic based on variance analysis of sensitivity information," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1386–1399, Nov. 2001.

[20] A. P. Engelbrecht and I. Cloete, "Incremental learning using sensitivity analysis," in *Proc. Int. Joint Conf. Neural Netw. (IJCNN)*, vol. 2. Jul. 1999, pp. 1350–1355.

[21] R. H. Kewley, M. J. Embrechts, and C. Breneman, "Data strip mining for the virtual design of pharmaceuticals with neural networks," *IEEE Trans. Neural Netw.*, vol. 11, no. 3, pp. 668–679, May 2000.

[22] M. J. Embrechts, F. A. Arciniegas, M. Ozdemir, and R. H. Kewley, "Data mining for molecules with 2-D neural network sensitivity analysis," *Int. J. Smart Eng. Syst. Design*, vol. 5, no. 4, pp. 225–239, 2003.

[23] J. M. Zurada, A. Malinowski, and S. Usui, "Perturbation method for deleting redundant inputs of perceptron networks," *Neurocomputing*, vol. 14, no. 2, pp. 177–193, 1997.

[24] M. Stevenson, R. Winter, and B. Widrow, "Sensitivity of feedforward neural networks to weight errors," *IEEE Trans. Neural Netw.*, vol. 1, no. 1, pp. 71–80, Mar. 1990.

[25] X. Zeng and D. S. Yeung, "Sensitivity analysis of multilayer perceptron to input and weight perturbations," *IEEE Trans. Neural Netw.*, vol. 12, no. 6, pp. 1358–1366, Nov. 2001.

[26] M. Li, "Robust optimization and sensitivity analysis with multi-objective genetic algorithms: Single- and multi-disciplinary applications," Ph.D. dissertation, Dept. Mech. Eng., Univ. Maryland, College Park, College Park, MD, USA, Nov. 2007.

[27] A. Saltelli *et al.*, *Global Sensitivity Analysis: The Primer*. New York, NY, USA: Wiley, 2008.
[28] H. M. Wagner, "Global sensitivity analysis," *Oper. Res.*, vol. 43, no. 6, pp. 948–969, 1995.
[29] S. Chen, S. A. Billings, and W. Luo, "Orthogonal least squares methods and their application to non-linear system identification," *Int. J. Control*, vol. 50, no. 5, pp. 1873–1896, 1989.
[30] K. Z. Mao, "Fast orthogonal forward selection algorithm for feature subset selection," *IEEE Trans. Neural Netw.*, vol. 13, no. 5, pp. 1218–1224, Sep. 2001.
[31] E. Fock, "Global sensitivity analysis approach for input selection and system identification purposes—A new framework for feedforward neural networks," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 8, pp. 1484–1495, Aug. 2014.
[32] A. Saltelli, S. Tarantola, and K. P.-S. Chan, "A quantitative model-independent method for global sensitivity analysis of model output," *Technometrics*, vol. 41, no. 1, pp. 39–56, 1999.
[33] I. M. Sobol, "Global sensitivity indices for nonlinear mathematical models and their Monte Carlo estimates," *Math. Comput. Simul.*, vol. 55, nos. 1–3, pp. 271–280, 2001.
[34] R. Durbin and D. E. Rumelhart, "Product units: A computationally powerful and biologically plausible extension to backpropagation networks," *Neural Comput.*, vol. 1, no. 1, pp. 133–142, 1989.
[35] I. M. Sobol, "On sensitivity estimation for nonlinear mathematical models," *Matematicheskoe Modelirovanie*, vol. 2, no. 1, pp. 112–118, 1990.
[36] A. Ismail and A. P. Engelbrecht, "Pruning product unit neural networks," in *Proc. Int. Joint Conf. Neural Netw.*, vol. 1. 2002, pp. 257–262.
[37] F. J. Martínez-Estudilloa, C. Hervás-Martínez, P. A. Gutiérrez, and A. C. Martínez-Estudillo, "Evolutionary product-unit neural networks classifiers," *Neurocomputing*, vol. 72, nos. 1–3, pp. 548–561, 2008.
[38] M. Torres, C. Hervás, and C. García, "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 Syst. Appl.*, vol. 36, no. 10, pp. 12225–12235, 2009.
[39] G.-B. Huang, H. Zhou, X. Ding, and R. Zhang, "Extreme learning machine for regression and multiclass classification," *IEEE Trans. Syst., Man, Cybern. B, Cybern.*, vol. 42, no. 2, pp. 513–529, Apr. 2012.
[40] G. S. Babu and S. Suresh, "Meta-cognitive RBF network and its projection based learning algorithm for classification problems," *Appl. Soft Comput.*, vol. 13, no. 1, pp. 654–666, 2013.
[41] A. Asuncion and D. Newman. (2007). *UCI Machine Learning Repository*. [Online]. Available: http://www.ics.uci.edu/~mlearn/MLRepository.html
[42] PASCAL. (2011). *Pascal (Pattern Analysis, Statistical Modelling and Computational Learning) Machine Learning Benchmarks Repository*. [Online]. Available: http://mldata.org/
[43] P. A. Gutiérrez *et al.*, "Hybridizing logistic regression with product unit and RBF networks for accurate detection and prediction of banking crises," *Omega*, vol. 38, no. 5, pp. 333–344, Oct. 2010.
[44] F. Fernández-Navarro, C. Hervás-Martínez, C. García-Alonso, and M. Torres-Jimenez, "Determination of relative agrarian technical efficiency by a dynamic over-sampling procedure guided by minimum sensitivity," *Expert Syst. Appl.*, vol. 38, no. 10, pp. 12483–12490, 2011.
[45] S. Ventura, C. Romero, A. Zafra, J. A. Delgado, and C. Hervás, "JCLEC: A Java framework for evolutionary computation," *Soft Comput.*, vol. 12, no. 4, pp. 381–392, 2008.
[46] M. Kendall, *Rank Correlation Methods*. London, U.K.: Griffin, 1948.
[47] A. Marrel, B. Iooss, B. Laurent, and O. Roustant, "Calculations of Sobol indices for the Gaussian process metamodel," *Rel. Eng. Syst. Safety*, vol. 94, no. 3, pp. 742–751, 2009.
[48] A. Marrel, B. Iooss, S. Da Veiga, and M. Ribatet, "Global sensitivity analysis of stochastic computer models with joint metamodels," *Statist. Comput.*, vol. 22, no. 3, pp. 833–847, 2011.
[49] J. D. Olden, M. K. Joy, and R. G. Death, "An accurate comparison of methods for quantifying variable importance in artificial neural networks using simulated data," *Ecol. Model.*, vol. 178, nos. 3–4, pp. 389–397, 2004.
[50] I. M. Sobol, "On quasi-Monte Carlo integrations," *Math. Comput. Simul.*, vol. 47, nos. 2–5, pp. 103–112, 1998.
[51] F. Fernández-Navarro, A. Riccardi, and S. Carloni, "Ordinal regression by a generalized force-based model," *IEEE Trans. Cybern.*, vol. 45, no. 4, pp. 844–857, Apr. 2015.
[52] J. K. Ravalico, H. R. Maier, G. C. Dandy, J. P. Norton, and B. F. W. Croke, "A comparison of sensitivity analysis techniques for complex models for environment management," in *Proc. 16th Int. Congr. Modelling Simulation*, Melbourne, VIC, Australia, 2005, pp. 2533–2539.
[53] R. I. Cukier, C. M. Fortuin, K. E. Shuler, A. G. Petschek, and J. H. Schaibly, "Study of the sensitivity of coupled reaction systems to uncertainties in rate coefficients. I Theory," *J. Chem. Phys.*, vol. 59, no. 8, pp. 3873–3878, 1973.
[54] G. J. McRae, J. W. Tilden, and J. H. Seinfeld, "Global sensitivity analysis—A computational implementation of the Fourier amplitude sensitivity test (FAST)," *Comput. Chem. Eng.*, vol. 6, no. 1, pp. 15–25, 1982.
[55] F. Cannavó, "Sensitivity analysis for volcanic source modeling quality assessment and model selection," *Comput. Geosci.*, vol. 44, pp. 52–59, Jul. 2012.
[56] J. M. Benítez, J. L. Castro, and I. Requena, "Are artificial neural networks black boxes?" *IEEE Trans. Neural Netw.*, vol. 8, no. 5, pp. 1156–1164, 1997.

Francisco Fernández-Navarro (M'13) received the M.Sc. degree in computer science from the University of Córdoba, Córdoba, Spain, in 2008, and the M.Sc. degree in artificial intelligence and the Ph.D. degree in computer science and artificial intelligence from the University of Málaga, Málaga, Spain, in 2009 and 2011, respectively.

He was a Research Fellow in Computational Management with the European Space Agency, Noordwijk, The Netherlands. He is currently an Associate Professor at the Universidad Loyola Andalucia, Córdoba, Spain. His current research interests include neural networks, ordinal regression, imbalanced classification, and hybrid algorithms.

Mariano Carbonero-Ruz received the B.Sc. degree in mathematics from the University of Seville, Seville, Spain, in 1985, the B.Sc. degree in economics from the Universidad Nacional de Educación a Distancia, and the Ph.D. degree in mathematics from the University of Seville in 1995.

He has been a Lecturer with ETEA from 1987 to 2013, a private Business Administration faculty affiliated to the University of Córdoba, Córdoba, Spain, and since then at the Universidad Loyola Andalucia, Córdoba, as a Tenured Lecturer. He is a member of AYRNA, a Spanish research group, involved in learning and artificial neural networks. His current research interests include computational intelligence, both theoretical research as applications of methods to solve real problems in different areas of economics and education.

David Becerra Alonso received the B.S. degree in physics from the Universidad de Córdoba, Córdoba, Spain, in 2005, where he specialized in the simulation of physical systems, the Ph.D. degree from the School of Computing, University of the West of Scotland, Paisley, U.K., in 2010, where he was involved on dynamical chaotic systems, and the master's degree in bioinformatics from the Universidad Internacional de Andalucia, Seville, Spain.

He is currently a Lecturer with the Universidad Loyola Andalucia, Córdoba, Spain. He collaborates with the research group AYRNA, from Universidad de Córdoba, Córdoba, Spain. His current research interests include dynamical systems, emergent collective behavior, and machine learning techniques and heuristics.

Mercedes Torres-Jiménez received the M.Sc. degree in business administration and the Ph.D. degree in economics and business administration from the University of Córdoba, Córdoba, Spain, in 1993 and 2000, respectively.

She was an Associate Professor from 1993 to 2013 with ETEA, a private Business Administration faculty affiliated to the University of Córdoba, and since then at the Universidad Loyola Andalucia, Córdoba, Spain. She is a member of AYRNA (Learning and Artificial Neural Networks) Research Group from 2000. Her current research interests include applications of computational intelligence methods to solve real problems in different areas of economics (especially in agriculture) and also related to International Development Cooperation.