Opening Black Box Data Mining Models Using Sensitivity Analysis

Paulo Cortez
Dep. of Information Systems/Algoritmi R&D Centre
University of Minho
4800-058 Guimarães Portugal
Email: http://www3.dsi.uminho.pt/pcortez

Mark J. Embrechts
Department of Industrial and Systems Engineering
Rensselaer Polytechnic Institute
CII 5219, Troy, NY 12180 - USA
Email: embrem@rpi.edu

***Abstract*—There are several supervised learning Data Mining (DM) methods, such as Neural Networks (NN), Support Vector Machines (SVM) and ensembles, that often attain high quality predictions, although the obtained models are difficult to interpret by humans. In this paper, we open these black box DM models by using a novel visualization approach that is based on a Sensitivity Analysis (SA) method. In particular, we propose a Global SA (GSA), which extends the applicability of previous SA methods (e.g. to classification tasks), and several visualization techniques (e.g. variable effect characteristic curve), for assessing input relevance and effects on the model's responses. We show the GSA capabilities by conducting several experiments, using a NN ensemble and SVM model, in both synthetic and real-world datasets.**

I. Introduction

Due to the advances of Information Technology, business and scientific databases are becoming commonplace. Data Mining (DM) enables the extraction of knowledge from such databases to the domain user or decision maker [1][2]. Under the supervised learning paradigm, the intention is build a data-driven model that learns an unknown underlying function that maps several input variables into one or more output targets. This includes two important DM goals: classification and regression.

Several supervised learning techniques are available, each one with its own purposes and advantages. Often, it is crucial to achieve DM models with an high predictive knowledge, i.e. capable of high quality estimates of the target. Another issue is explanatory power, i.e. if it is possible to extract human understandable knowledge from the data-driven model. Such knowledge is important to check if the obtained model makes sense to the domain experts and if it is interesting (e.g. if it unveils potentially useful, interesting, or novel information) [1][3]. Increasing model interpretability allows a better acceptance of the DM results by the domain users and this is particularly relevant in critical applications, such as control or medicine.

There are supervised DM methods that often lead to data-driven models with a high predictive accuracy, although the obtained models are difficult to understand by humans. Hence, such models are usually treated as "black boxes". This includes a wide range of methods, such as neural networks (NN) (e.g. multilayer perceptrons and radial basis-functions) [4], Support Vector Machines (SVM) and other kernel-based methods [5], and even ensembles (where multiple models are combined to achieve a better predictive performance) [6].

To increase interpretability from black box DM models, there are two main approaches: extraction of rules and use of visualization techniques. The extraction of rules is the most popular solution [7][8][9]. However, such extraction is often based on a process that simplifies the model complexity and may lead to rules that do not accurately represent the original model. For instance, a pedagogical technique can be used to extract the direct relationships between the inputs and outputs of a NN classifier by using a decision tree [10]. While producing human understandable rules, this decision tree discretizes the classifier separating hyperplane, thus leading to information loss. Turning to visualization techniques, several graphical methods were proposed, such as: Hinton and Bond diagrams for NN [11]; showing NN weights and classification uncertainty [12]; and improving the interpretability of kernel-based classification methods [13]. Yet, most of these graphical techniques are specific to a given learning method or DM goal (e.g. classification).

In this work, we propose a novel visualization approach based on a Sensitivity Analysis (SA), which is a simple method that measures the effects on the output of a given model when the inputs are varied through their range of values [14]. While initially proposed for NN, SA can be used with virtually any supervised learning method, such as partial least squares [15] and SVM [16]. This method allows a ranking of the inputs that is based on the amount of output changes that are produced due to disturbances in a given input. Hence, SA has been mostly used as a variable/feature selection method (e.g. to select the least relevant feature that is deleted in each iteration of a backward selection) [14][15][13][16]. However, SA can also be used to explain the model, as recognized in [17] but more explored in [18] and [19], thus opening the black box.

In [17], a computationally efficient one-dimensional (1-D) SA was proposed, where only one input is changed at the time, being the remaining ones hold at their average values. Later, in [18] a two-dimensional (2-D) SA variant was presented as a diagnostic tool to explain the effects of two features on the model. In both studies, only numerical inputs and regression tasks were modeled. In this paper, we

978-1-4244-9927-4/11/$26.00 ©2011 IEEE

propose a global framework for using SA to open black box DM models. In Sections II-A and II-B, we extend the SA method of [17][18] (e.g. to deal with: classification tasks, discrete variables and up to the I-dimensional case, where I is the number of inputs). Also, we present several visualization techniques (Section II-C) that show input relevance and 1-D and 2-D SA effects. In Sections II-D and II-E, we describe the learning methods and datasets adopted. Then, we explore the use of SA and visualization techniques in both synthetic and real-world datasets (Section III). Finally, conclusions are drawn (Section IV).

II. Materials and Methods

A. Sensitivity Analysis Methods

A supervised DM model (M) is adjusted to a dataset D, i.e. training data, made up of N examples of I input variables and one output target (y). Each input ($x_i : i \in \{1, \ldots, I\}$) or output variable ($y$) can be categorical or continuous. Let $\hat{y}$ denote the value predicted by M and $N_{\hat{y}}$ the number of elements returned by M for a single input example. A regression task assumes one continuous output ($\hat{y} \in \Re$ and $N_{\hat{y}} = 1$). For classification, there is a categorical output. Such output can be modeled by M using one discrete output ($N_{\hat{y}} = 1$ and $\hat{y} \in \{G_1, \ldots, G_{N_G}\}$ groups). Another possibility for M is to model class probabilities, where $\sum_{c \in \{G_1, \ldots, G_{N_G}\}} \hat{y}(c) = 1$, $N_{\hat{y}} = N_G$ and $\hat{y}(c)$ is the output probability for class c. Discrete data can be further classified into:

- **ordered** – with $N_G \geq 2$ ordered values (e.g. $y \in$\{yes, no\} or $y \in$\{low, medium, high\});
- **nominal** – non-ordered with $N_G > 2$ classes (e.g. $y \in$\{red, blue, yellow\}).

The 1-D SA works by first holding all input variables at a given baseline vector b (e.g. composed by the mean or median attribute values). Then, each input x_a varies through its range according to a scanning function $scan(x_a, l)$ that returns a sequence with l values taken from x_a. Finally, the respective model responses ($\hat{y}_a$) are used to compute a sensitivity metric. The setup proposed in [17] for regression tasks and continuous values was: $l = 6$, b composed of the mean attribute values and:

$$\begin{array}{l} scan(x_a, l) = seq(\min(x_a), \max(x_a), l) \\ seq(A, B, l) = \{A, A + k, A + 2k, \ldots, B\} \\ k = (A - B)/(l - 1) \end{array} \quad (1)$$

where $seq(A, B, l)$ produces a **regular** sequence from the minimum to the maximum value. The 2-D SA was proposed in [18], working similarly to the 1-D case, except that the sensitivity is analyzed for a pair of inputs (x_{a1}, x_{a2}) that vary simultaneously.

In this paper, we propose a Global SA (GSA) method (Algorithm 1) that extends the previous methods to deal with: discrete variables, distinct scanning functions and a sensitivity that uses F features (ranging from $\#F$ =1-D, up to the $\#F = I$-D case). The S_D jagged array is built using Algorithm 2, while $predict(M, X)$ is a function that returns the responses of model M given the input matrix X (of $N \times I$ size). The REP procedure is equivalent to the R rep function [20] (e.g. REP((1,2),2,2)=(1,1,2,2,1,1,2,2)). The computational complexity for GSA is $O(m_X \times P)$, where P is the complexity when using M to predict one example. When the number of scanned elements is l for all F attributes, then $m_X = l^{\#F}$. Depending on the SA purpose (e.g. ranking inputs or visualization of the model's behavior), the GSA responses ($\hat{y}_{GSA} = X[*, y_{col}]$) can be averaged, according to a given input x_a with l levels or pair of inputs (x_{a1},x_{a2}) with l_1 (for x_{a1}) and l_2 levels:

$$\begin{array}{l} \hat{y}_a = \{\overline{X}[x_{row}, y_{col}] : X[x_{row}, a] = x_{a,j}, j \in \{1, \ldots, l\}\} \\ \hat{y}_{(a1,a2)} = \{\overline{X}[x_{row}, y_{col}] : X[x_{row}, a1] = x_{a1,j} \wedge \\ X[x_{row}, a2] = x_{a2,h}, j \in \{1, \ldots, l_1\}, h \in \{1, \ldots, l_2\}\} \end{array} \quad (2)$$

where $x_{a,j}$ denotes the j-th scanned element from x_a. When modeling probabilities, $\hat{y}_a$ (or $\hat{y}_{(a1,a2)}$) is computed for each individual class probability (e.g. $y_{col} = I+1$ for the first class, G_1).

Algorithm 1 Global Sensitivity Analysis

```
procedure GSA($M$, $S_D$, $F$, $b$, $N_{\hat{y}}$)
    $m_X \leftarrow 1$
    for $a \in F$ do                                   ▷ compute $m_X$ length
        $m_X \leftarrow m_X \times length(S_D[a, *])$
    end for
    $X \leftarrow$ matrix $m_X \times (I + Y_{col})$          ▷ rows × columns
    for $a \in \{1, \ldots, I\}/F$ do
        $X[*, a] \leftarrow b[a]$                  ▷ set $/F$ columns to baseline
    end for
    $e \leftarrow 1$
    for $a \in F$ do                                      ▷ set SA inputs
        $x'_a \leftarrow S_D[a, *]$
        $t \leftarrow m_X/(e \cdot length(x'_a))$
        $X[*, a] \leftarrow$ REP($x'_a, e, t$)                  ▷ replicate $x'_a$
        $e \leftarrow e \cdot length(x'_a)$
    end for
    $y_{col} \leftarrow \{I + 1, \ldots, I + N_{\hat{y}}\}$             ▷ output columns
    $X[*, y_{col}] \leftarrow predict(M, X[*, \{1, \ldots, I\}])$
    Output: $X$          ▷ matrix with SA inputs and responses
end procedure
procedure REP($x$, $each$, $times$)               ▷ auxiliary function
    $x_r = \emptyset$                                      ▷ empty vector
    for $j \in \{1, \ldots, times\}$ do
        $x_e = \emptyset$                                  ▷ empty vector
        for $i \in x$ do
            $x'_e \leftarrow$ vector with $each \times length(x)$ elements
            $x'_e[*] \leftarrow i$          ▷ all $x'_e$ elements are set to $i$
            $x_e \leftarrow c(x_e, x'_e)$              ▷ concatenate operator
        end for
        $x_r \leftarrow c(x_r, x_e)$                   ▷ concatenate operator
    end for
    Output: $x_r$                  ▷ vector with replicates from $x$
end procedure
```

Algorithm 2 Scanning data method

```
procedure SCAN DATA($D$, $F$, $l$)
    for $a \in F$ do
        $S_D[a,*] \leftarrow scan(D[*,a],l)$
    end for
    Output: $S_D$        ▷ jagged array with scanned inputs
end procedure
```

For continuous values, $scan(x_a)$ can be set to the regular sequence (Equation 1). The **quantile** scan can be used as an alternative, where the intention is to sample values from x_a that are more closer to the variable distribution in the dataset:

$$scan(x_a, l) = unique(\{Q(p) : p \in seq(0, 1, l)\}) \qquad (3)$$

where $Q(p)$ is the quantile function for a given probability $p \in [0, 1]$ of x_a and $unique$ is a function that deletes repeated elements from a set. Ordered attributes can be encoded into the numeric ordered scale: $\{1, 2, \ldots, N_G\}$. Then, N_G or l discrete values $\in \{G_1, \ldots, G_{N_G}\}$ (the x_a classes) can be scanned using the regular or quantile sequences over the ordered scale. For nominal attributes, we suggest the use of all discrete values or a random sampling of l distinct values from x_a.

To simplify the computation, which is particularly useful within a feature selection procedure, b can be set to the mean or median attribute values of D [17][18]. In case of ordered attributes, the mean or median index of the ordered scale ($middle$) can be used to select the middle x_a value ($b[a] = N_{middle}$). For nominal attributes, we propose the use of the mode, i.e. most common x_a value. Nevertheless, we should stress that any baseline vector (b) can be used in Algorithm 1, including the input values of a given example from D or average values for a particular cluster (e.g. patients younger than eighteen).

B. Sensitivity Measures

Three sensitivity measures were proposed in [17] for continuous outputs (i.e. regression), namely range (r), gradient (g) and variance (v):

$$\begin{aligned} r_a &= \max(\hat{y}_a) - \min(\hat{y}_a) \\ g_a &= \textstyle\sum_{j=2}^{L} |\hat{y}_{a,j} - \hat{y}_{a,j-1}|/(L-1) \\ v_a &= \textstyle\sum_{j=2}^{L} (\hat{y}_{a,j} - \overline{\hat{y}_a})^2/(L-1) \end{aligned} \qquad (4)$$

For all metrics, the higher the value, the more relevant is the input. Thus, its relative importance (R_a) can be given by [16]:

$$R_a = s_a / \sum_{i=1}^{I} s_i \times 100\,(\%). \qquad (5)$$

where $\hat{y}_{a,j}$ is the sensitivity response for $x_{a,j}$ and s is the sensitivity measure (i.e. r, g or v). For demonstration purposes, Fig.1 shows sensitivity measures for four distinct response curves. The graph shows different rankings, according to the measure considered (e.g. the g curve ranking is: <B and D, A, C>).

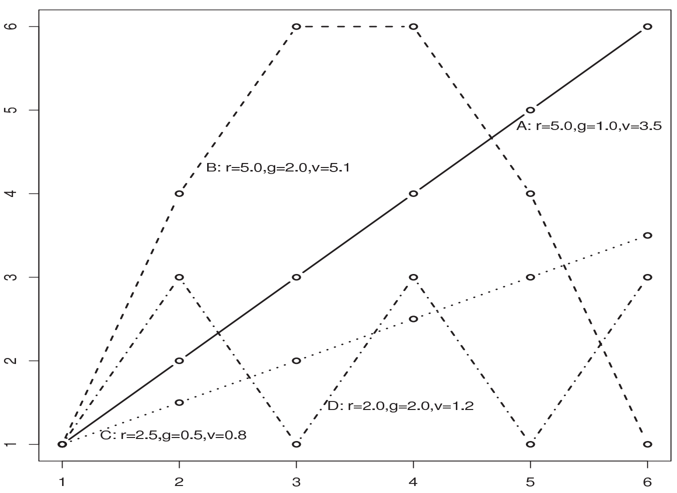

Fig. 1. Example the sensitivity measures for different continuous responses

TABLE I
SENSITIVITY MEASURES FOR A THREE-CLASS CLASSIFICATION EXAMPLE

	nominal (ii)			ordered (i)		
$\hat{y}_a$	r	g	v	r	g	v
AAAA	0.00	0.00	0.00	0.00	0.00	0.00
AAAB	0.67	0.22	0.17	1.00	0.33	0.25
AABB	0.67	0.22	0.22	1.00	0.33	0.33
ABAB	0.67	0.67	0.22	1.00	1.00	0.33
AABC	1.00	0.44	0.28	2.00	0.67	0.92
ABAC	1.00	0.67	0.28	2.00	1.33	0.92

For classification, there are three output domain possibilities: **i**) ordered output, **ii**) nominal output and **iii**) output probabilities. We propose the following classification measures. In case **i**, first encode the discrete values into an ordered scale (e.g. $\{1, 2, \ldots, N_G\}$) and then use Equation 4. For **ii**, first apply the popular One-of-N_G transformation, where one binary variable is assigned to each class (e.g. red→(1,0,0); blue→(0,1,0); yellow→(0,0,1)). Then use:

$$s_a = \overline{s_{a(c)}} : c \in \{G_1, \ldots, G_{N_G}\} \qquad (6)$$

where $s_{a(c)}$ is the sensitivity measure for attribute a and output class c. Under **iii**, use Equation 6 directly.

As an example, we select a three-class classification ($y \in \{\text{A,B,C}\}$). Table I shows the proposed sensitivity measures for an hypothetical $\hat{y}_a$ that is obtained when varying x_a with $l = 4$. As shown by the table, for nominal and ordered scenarios, higher changes in $\hat{y}_a$ tend to result in an increase of the sensitivity measure, although such increase depends on the type of measure used. The probability modeling case, **iii**, we adopt an hypothetical $\hat{y}_a$ for $l = 3$ and when analyzing three inputs (x_1, x_2 and x_3), as shown in Table II. For each $\hat{y}_{a,i}$, $i \in \{1, 2, 3\}$, we present the class probabilities (e.g for $\hat{y}_{1,1}$, $\hat{y}(\text{A}) = 1.0$) and expected class (in brackets, corresponding to the class with the highest probability). Again, higher response changes lead to an increase for all measures.

TABLE II
SENSITIVITY MEASURES FOR A THREE-CLASS PROBABILITY EXAMPLE

	$\hat{y}_1$ (class)	$\hat{y}_2$ (class)	$\hat{y}_3$ (class)
$\hat{y}_{a,1}$	1.0,0.0,0.0 (A)	1.0,0.0,0.0 (A)	0.7,0.1,0.2 (A)
$\hat{y}_{a,2}$	0.8,0.2,0.0 (A)	0.8,0.2,0.0 (A)	0.2,0.7,0.1 (B)
$\hat{y}_{a,3}$	0.6,0.4,0.0 (A)	0.4,0.6,0.0 (B)	0.1,0.2,0.7 (C)
r	0.27	0.40	0.60
g	0.13	0.20	0.40
v	0.03	0.06	0.10

C. Visualization Techniques

We assume that M is a given supervised learning model that was designed to include the lowest set of I input variables (e.g. use of feature selection) in order to achieve a satisfactory prediction accuracy. After applying the GSA algorithm, several visualization methods can be used to open the black box (examples are shown in Section III):

1) Input Importances: To show the input importances, we propose a bar plot with the R_a importances (Equation 5), sorted from the highest to the lowest values. When using multiple runs, the bar plot R_a values can be averaged and whiskers can be added to show confidence intervals.

2) Variable Effect Characteristic Curve: To present the average impact of a given input x_a in the model, we suggest the use of the Variable Effect Characteristic (VEC) curve [19], which plots the the $x_{a,j}$ values (x-axis) versus the (average, if Equation is 2 used) $\hat{y}_{a,j}$ responses (y-axis). Between two consecutive $x_{a,j}$ values, the VEC plot uses a line (interpolation) for continuous values and a horizontal segment for categorical data. To enhance the visualization analysis, nominal x_a values can be sorted according to the average $\hat{y}_{a,j}$ response. Also, several VEC curves can be plotted in the same graph. In such case, the x-axis should be scaled (e.g. within [0,1]) for all x_a values. Also, as shown in [19], the VEC curves can be combined with x_a histograms, to increase the interpretation of the results (i.e. showing both the frequency and sensitivity response values). Similarly to Receiver Operating Characteristic (ROC) analysis, a VEC plot can be built for each target class, when modeling class probabilities (case **iii** of Section II-B). When using multiple runs, the distinct VEC curves can be averaged vertically, over all runs.

3) Variable Effect Characteristic Surface and Contour Plot: Finally, to show the impact of a pair of inputs (x_{a1},x_{a2}), we propose the use of a VEC surface [18] and contour plot, as shown in Section III. The VEC surface provides more detail when compared with the contour plot, yet, a good interpretability is dependent on a correct adjustment of the 3D axis/angle representation.

D. Supervised Learning Methods

While virtually any supervised learning method could be used, we adopt two models: NN and SVM, as implemented in the rminer library of the R tool [21].

The NN is based on an ensemble that computes the average of the predictions of N_r multilayer perceptrons, all with a hidden layer of H neurons with logistic functions. For regression, the output neuron uses a linear function. For a binary classification, there is one output neuron with a logistic function. Under multi-class tasks ($N_G > 2$), there are N_G linear output neurons and the softmax function is used to transform these outputs into class probabilities. The predicted class is given by the highest probability. The training (BFGS algorithm) of each network is stopped when the error slope approaches zero or after a maximum of M_e epochs.

The SVM uses the popular gaussian kernel, which contains the parameter γ. The SVM is fit using the sequential minimal optimization (SMO) learning algorithm. When modeling regression datasets, the ϵ-insensitive cost function adopted. For classification, the model can be set to estimate classes or class probabilities. For multi-class tasks, the one-against-one approach is used. To reduce the search space, the rminer uses heuristics to set the C and ϵ parameters.

Before feeding the NN and SVM models, the nominal inputs are encoded using a One-of-N_G transformation, while the real inputs are standardized to a zero mean and one standard devision. Turning to the NN and SVM hyperparameters (i.e. H and γ), these are set using a simple grid search. For a given hyperparameter set of values, a cross-validation is applied over the training data and the value with the lowest estimation error is selected.

E. Data

In this work, we designed two synthetic functions to create classification and regression synthetic datasets. Also, we selected two real-world problems from the UCI repository [22]: wine quality (classification or regression) and servo (regression). The intention is to use the controlled synthetic functions to assess the sensitivity measure performances, while the real-world data are used to show the SA visualization techniques' capabilities.

For the synthetic data, we set three inputs (x_1, x_2 and x_3), where each input contains $N = 1000$ points under a normal distribution sampling with a mean of $N/2$ and standard deviation of $N/8$. Then, we set (Fig. 2):

$$\begin{array}{ll} y = 0.7 \sin(\pi \cdot \frac{x_1}{2N}) + 0.3 \sin \pi \cdot \frac{x_2}{2N}) & \text{(sin1)} \\ y = \sin(\pi \cdot \frac{x_1}{2N}) \cdot (0.15 \sin \pi \frac{x_2}{0.25N} + 1) & \text{(sin2)} \end{array} \tag{7}$$

In both functions, the most important input is x_1 (with an impact around 70%), followed by x_2, while x_3 has a null effect. The former function uses an additive effect to combine x_1 and x_2, while the latter uses a nonlinear multiplicative combination. The two sin datasets are available at: http://www3.dsi.uminho.pt/pcortez/gsa. For classification, we discretized the outputs into 3 classes, according to: G_1 =A ($y \in [0, 0.65[$), G_2 =B ([0.65,0.75]) and G_3 =C ([0.75,1]).

The white wine quality (wwq) includes 4898 samples from the northwest region of Portugal [16][19]. The regression goal is to predict human expert taste preferences ($y \in \{3, \ldots, 9\}$) based on 11 analytical inputs (continuous values). For classification, we set a binary task, where the goal is to detect very good wine ($y > 6$). The servo is a nonlinear dataset related

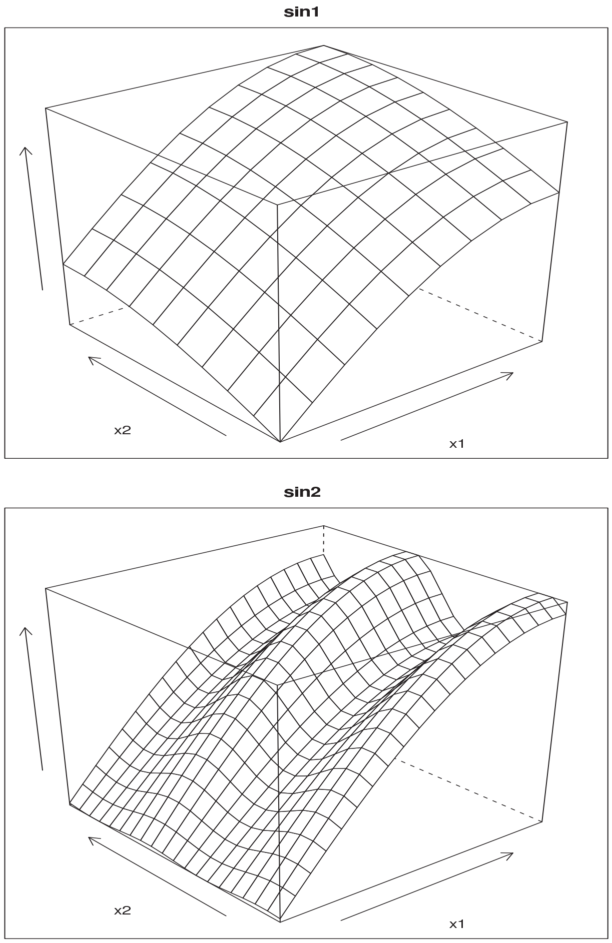

Fig. 2. The synthetic functions (sin1 and sin2)

to rise time of a servomechanism and includes 167 examples with 4 inputs (2 nominal and 2 continuous).

III. Experiments and Results

A. Predictive Power

All experiments reported here were conducted using the rminer library and R tool [21], under a linux server. For NN, we set $N_r = 7$ and M_e =100. The grid search ranges for NN and SVM were $H \in \{1, 2, \ldots, 10\}$ and $\gamma \in \{2^{-15}, 2^{-13}, \ldots, 2^3\}$, in a total of 10 searches for each method. We adopted an internal 3-fold cross-validation (using only training data) to get the estimation error. To assess the predictive capabilities of each method (i.e. test error), we applied 10 runs of 10-fold cross-validation to all datasets except the wine data, where a 3-fold validation was used (since the number of examples is quite large when compared with the remaining datasets). For multi-class classification (case **ii** of Section II-B), the error metric is the overall classification accuracy (ACC) [23]. When modeling class probabilities (case **iii**), we adopt the global Area Under the Curve (AUC), which weights the area of each ROC class according to its prevalence in the data [24]. For regression (**reg**), the Mean Absolute error (MAE) was used [23].

TABLE III
The best model and test error results

dataset	**Model**	**Metric**	**Test Error**
sin1-ii	NN ($\tilde{H} = 4$)	ACC	98.9±0.2%
sin1-iii	NN ($\tilde{H} = 4$)	AUC	99.9±0.0%
sin1-reg	NN ($\tilde{H} = 10$)	MAE	0.00±0.00
sin2-ii	NN ($\tilde{H} = 2$)	ACC	98.3±0.2%
sin2-iii	NN ($\tilde{H} = 2$)	AUC	99.9±0.0%
sin2-reg	NN ($\tilde{H} = 5$)	MAE	0.00±0.00
wwq-iii	SVM ($\tilde{\gamma} = 2^1$)	AUC	86.5±0.0%
wwq-reg	SVM ($\tilde{\gamma} = 2^{-1}$)	MAE	0.46±0.00
servo	NN ($\tilde{H} = 7$)	MAE	0.22±0.01

Table III shows the best model, according to the average test error (last column, shown in terms of the mean value and respective 95% t-student confidence intervals). For each model, the median hyperparameter is also shown in brackets (e.g. first row, $\tilde{H} = 4$). High quality predictive results were achieved. NN is the best method, except for the wine data (wwq), where SVM presents the best performance.

B. Explanatory Power

After assessing the predictive capability of the best models, we show here how to open the black box. To simplify the explanation and analyze just one model for each dataset, the best models of Table III were retrained using all data and the median hyperparameter value (e.g. $H = 4$ for the sin1-ii dataset). Unless stated otherwise, in the next experiments we use the regular scan with $l = 6$ and baseline (b) set to the mean of the attributes.

First, we analyze the synthetic relative input importances (R_a values, Table IV). In the table, each cell shows R_{x_1}, R_{x_2}, R_{x_3} and the cells where $R_{x_1} > R_{x_2}$ and $R_{x_3} = 0$ are shown in bold. The first 1-D setup (as used in [17], which is equivalent to the GSA when $\#F = 1$ for all inputs) shows good input ranking results only for the regression tasks, while the proposed GSA (using $\#F = 3$ and then Equation 2) achieves the pretended ranking for all except 4 cases (i.e. sin1-ii and v; and all sin2-ii input relevances). Regarding the sensitivity measures, both r and g produce relative importances that are closer to x_1 and x_2 influences in y, while v tends to give a higher impact to the most important input (x_1). In the remaining experiments, we adopt the gradient metric.

To demonstrate the visualization of input importances, we selected the wwq-reg dataset, which includes 11 inputs. In this case, the full GSA ($\#F$) is computationally expensive, as it requires the prediction of 6^{11} examples. Thus, we first applied the 1-D ($\#F = 1$) for all 11 inputs. The respective bar plot (left of Fig.3), shows the sulphates as the most relevant attribute. Then, we applied a 2-D ($\#F = 2$) within all $(x_{a1=\text{sulphates}}, x_{a2})$ pairs and then used Equation 2 to compute the R_{a2} values (right of Fig.3). The second bar plot shows a higher ranking for the alcohol and pH levels, i.e., when predicting wine quality, these two inputs have a higher interaction with the sulphate attribute.

TABLE IV

RELATIVE INPUT IMPORTANCES FOR THE SYNTHETIC DATASETS

setup	1-D SA ($\#F = 1$), $l = 6$		
	r	g	v
sin1-ii	0.4,0.6,0.0	0.3,0.7,0.0	0.4,0.6,0.0
sin2-ii	0.4,0.6,0.0	0.2,0.8,0.0	0.4,0.6,0.0
sin1-iii	0.4,0.6,0.0	0.3,0.7,0.0	0.4,0.6,0.0
sin2-iii	0.4,0.6,0.0	0.2,0.8,0.0	0.5,0.5,0.0
sin1-reg	**0.7,0.3,0.0**	**0.7,0.3,0.0**	**0.9,0.1,0.0**
sin2-reg	**0.8,0.2,0.0**	**0.6,0.4,0.0**	**0.9,0.1,0.0**
setup	GSA ($\#F = 3$), $l = 12$		
	r	g	v
sin1-ii	**0.6,0.4,0.0**	**0.7,0.3,0.0**	0.5,0.5,0.0
sin2-ii	0.3,0.3,0.3	0.2,0.6,0.2	0.4,0.4,0.2
sin1-iii	**0.7,0.3,0.0**	**0.6,0.4,0.0**	**0.9,0.1,0.0**
sin2-iii	**0.8,0.2,0.0**	**0.7,0.3,0.0**	**0.9,0.1,0.0**
sin1-reg	**0.7,0.3,0.0**	**0.7,0.3,0.0**	**0.9,0.1,0.0**
sin2-reg	**0.8,0.2,0.0**	**0.6,0.4,0.0**	**0.9,0.1,0.0**

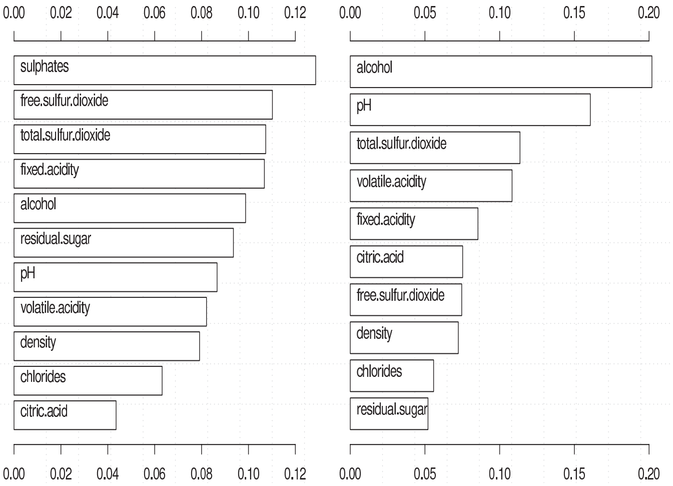

Fig. 3. Bar plots with the 1-D input importances (left) and 2-D interactions with the sulphates input (right) for the wwq-reg dataset

Fig. 4 is an example of how to compare the average impact of a given attribute. Here, we applied GSA with #F=4 and then averaged the $\hat{y}_a$ responses (y-axis). From the figure, it is clear that pgain has the highest impact, with a nonlinear decreasing effect in the servo rise time. Another VEC curve example is given in Fig. 5, where a 1-D SA, with $l = 24$ and b=median of the attributes, was used for the wwq-iii model. The graph shows a nonlinear influence of the pH input (the most relevant attribute: $R_{pH} = 19\%$), where values in the range $[3.4; 3.5]$ produce a higher increase in the probability of being a high quality wine. Finally, Fig. 6 presents a vertically averaged VEC curve (GSA with $\#F = 3$, average of all N_r =7 individual responses of the NN ensemble), and respective t-test 99% confidence interval whiskers for x_1 and sin1-reg.

To visualize interactions of a pair of inputs (2-D GSA, with $F = \{x_{a1}, x_{a2}\}$), several VEC surfaces and contours are shown. In all VEC surfaces, the axis arrow goes in the direction from minimum to maximum value, while the scale of the $\hat{y}_{x_{a1},x_{a2}}$ values is shown at the right (from white to black). Fig. 7 shows two VEC surfaces, for sin2-ii (with $l = 12$; in the right axis the scale A=1, B=2 and C=3 is

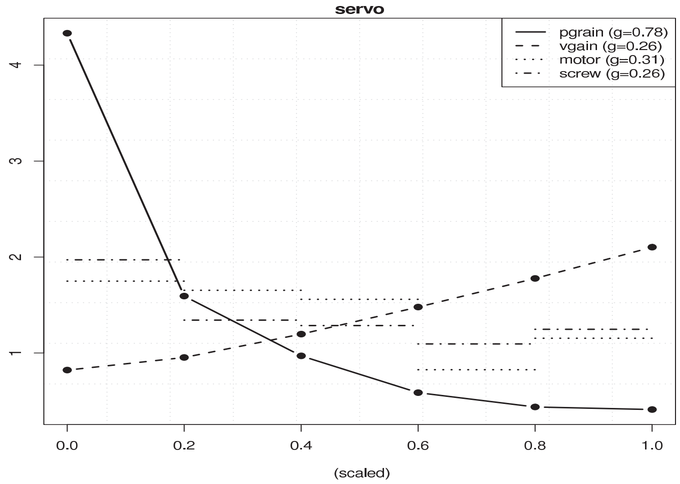

Fig. 4. VEC curves for the four inputs of the servo dataset

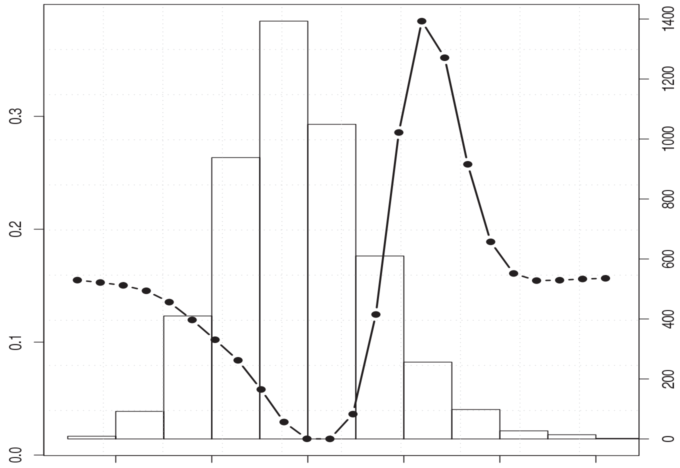

Fig. 5. VEC curve and histogram for the pH input (x-axis) and the respective high quality wine probability (left of y-axis) and frequency (right of y-axis)

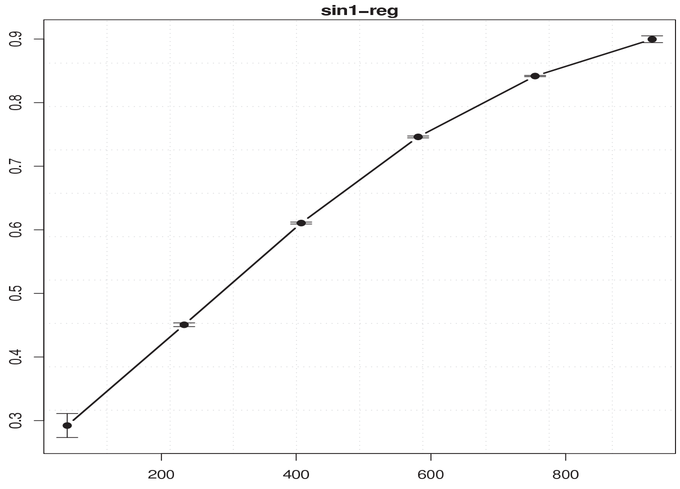

Fig. 6. VEC curve showing x_1 ($x-$axis) impact over sin1-reg

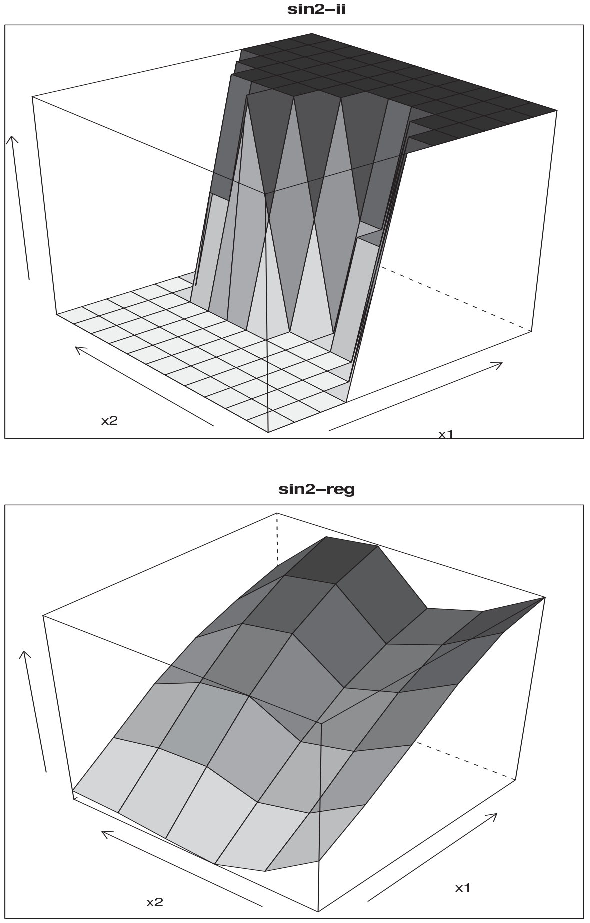

Fig. 7. VEC surfaces for the sin2-ii (top, $l = 12$) and sin2-reg (bottom)

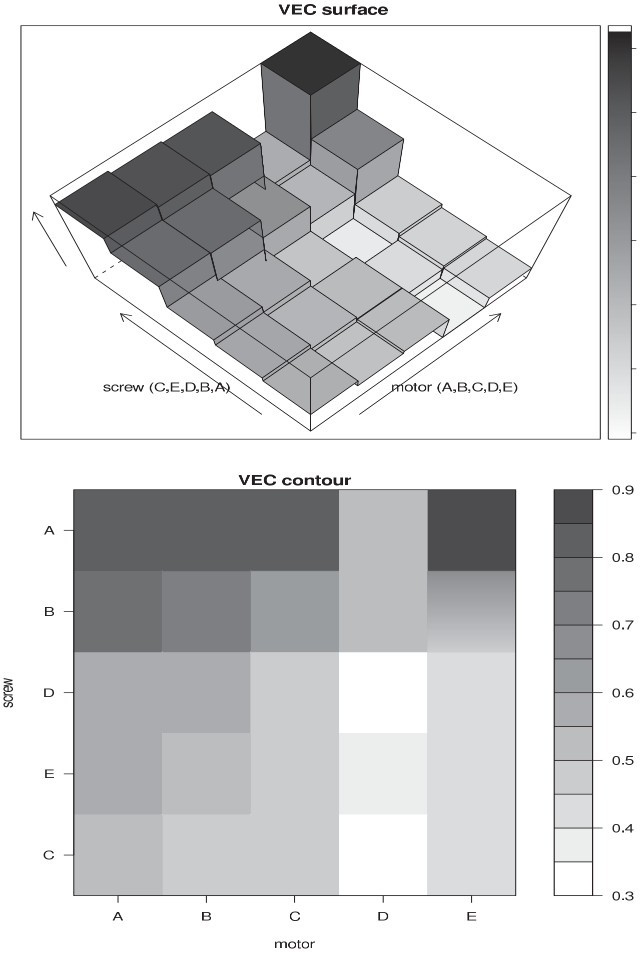

Fig. 8. VEC surface (top) and contour (bottom) for servo ($l = 5$)

used) and sin2-reg. Both curves confirm the highest impact of input x_1. Also, the VEC surface is quite similar to the sin2 function (bottom of Fig. 2). Fig. 8 shows the impact of the categorical motor and screw inputs in the servo data. To ease the visualization, in both surface and contour plots, the screw values were sorted by their average impact in the model. Finally, we selected the most relevant pair for the wwq-reg task ($x_{a1=\text{sulfates}}, x_{a2=\text{alcohol}}$) and applied a 2-D SA with a quantile scan (Fig. 9). The wine VEC surface uses the same levels that appear in the contour x/y-axis. It is interesting to notice that the VEC contour shows two high quality wine regions, (0.39,11.7) and (0.57,9.4), and we believe such knowledge is useful for wine taste experts.

IV. Conclusions

Complex supervised learning Data Mining (DM) methods, such as Neural Networks (NN), Support Vector Machines (SVM) and ensembles, are capable of high quality prediction results and thus are useful to support decision making. However, the obtained models are often treated as black-boxes, since they are difficult to understand by humans. Improving model interpretability increases the acceptance of the DM results by the domain users and this is an important issue in critical applications, such as control or medicine.

In this work, we have shown how virtually any supervised DM model can be "opened" by using the proposed Global Sensitivity Analysis (GSA) algorithm in combination with several visualization techniques, such as the Variable Effect Characteristic (VEC) curve. The effectiveness of this approach was assessed in several synthetic and real-world datasets, using both NN and SVM models. In the future, we intend to explore different learning methods (e.g. partial least squares), more efficient sampling methods to generate the sensitivity analysis responses (e.g. Monte Carlo methods) and enlarge the experiments to include more real-world datasets (e.g. financial data) with a higher number of inputs.

Acknowledgment

The authors would like to thank the anonymous reviewers for their helpful comments.

References

[1] U. Fayyad, G. Piatetsky-Shapiro, and P. Smyth, *Advances in Knowledge Discovery and Data Mining*. MIT Press, 1996.

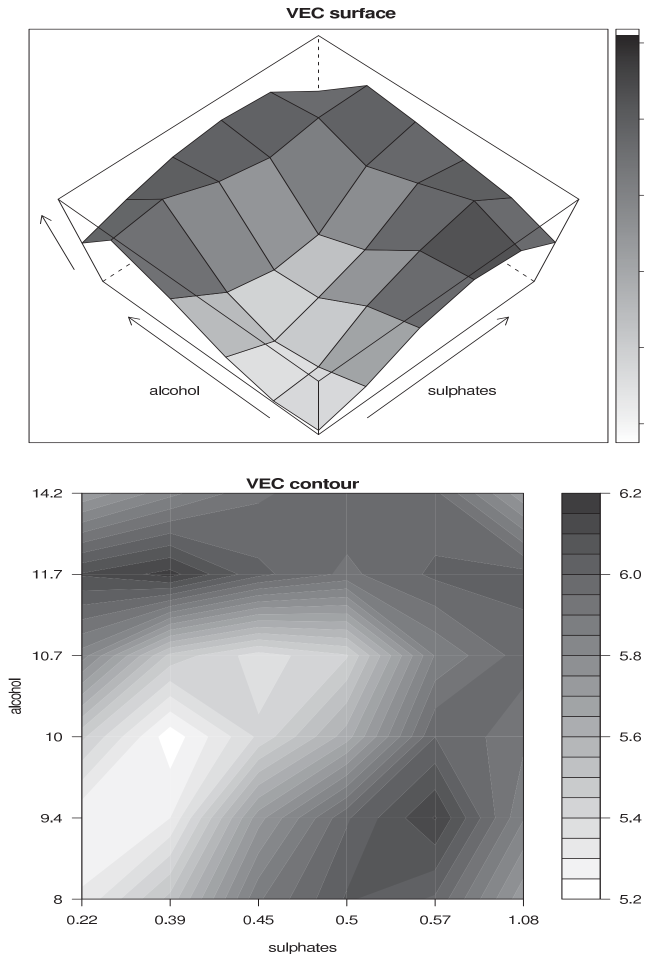

Fig. 9. VEC surface (top) and contour (bottom) for wwq-reg (quantile scan)

[2] E. Turban, R. Sharda, and D. Delen, *Decision Support and Business Intelligence Systems*, 9th ed. Prentice Hall, 2010.

[3] P. Chapman, J. Clinton, R. Kerber, T. Khabaza, T. Reinartz, C. Shearer, and R. Wirth, "CRISP-DM 1.0: Step-by-step data mining guide," *CRISP-DM consortium*, 2000.

[4] S. Haykin, *Neural Networks - A Compreensive Foundation*, 2nd ed. New Jersey: Prentice-Hall, 1999.

[5] N. Cristianini and J. Shawe-Taylor, *An introduction to support Vector Machines (and other kernel-based learning methods)*. Cambridge University Press, 2000.

[6] T. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems, Lecture Notes in Computer Science 1857*, J. Kittler and F. Roli, Eds. Springer-Verlag, 2000, pp. 1–15.

[7] A. Tickle, R. Andrews, M. Golea, and J. Diederich, "The Truth Will Come to Light: Directions and Challenges in Extracting the Knowledge Embedded Within Trained Artificial Neural Networks," *IEEE Transactions on Neural Networks*, vol. 9, no. 6, pp. 1057–1068, 1998.

[8] R. Setiono, "Techniques for Extracting Classification and Regression Rules from Artificial Neural Networks," in *Computational Intelligence: The Experts Speak*, D. Fogel and C. Robinson, Eds. Piscataway, NY, USA, IEEE, 2003, pp. 99–114.

[9] D. Martens, B. Baesens, T. Van Gestel, and J. Vanthienen, "Comprehensible credit scoring models using rule extraction from support vector machines," *European Journal of Operational Research*, vol. 183, no. 3, pp. 1466–1476, 2007.

[10] A. Silva, P. Cortez, M. F. Santos, L. Gomes, and J. Neves, "Mortality assessment in intensive care units via adverse events using artificial neural networks," *Artificial Intelligence in Medicine*, vol. 36, no. 3, pp. 223–234, 2006.

[11] M. Craven and J. Shavlik, "Visualizing learning and computation in artificial neural networks," *International Journal on Artificial Intelligence Tools*, vol. 1, no. 3, pp. 399–425, 1992.

[12] F. Tzeng and K. Ma, "Opening the black box-data driven visualization of neural networks," in *Proceedings of the Visualization (VIS 05)*. IEEE, Oct. 2005, pp. 383–390.

[13] B. Cho, H. Yu, J. Lee, Y. Chee, I. Kim, and S. Kim, "Nonlinear support vector machine visualization for risk factor analysis using nomograms and localized radial basis function kernels," *Information Technology in Biomedicine, IEEE Transactions on*, vol. 12, no. 2, pp. 247–256, 2008.

[14] D. Ruck, S. Rogers, and M. Kabrisky, "Feature selection using a multilayer perceptron," *Journal of Neural Network Computing*, vol. 2, no. 2, pp. 40–48, 1990.

[15] M. Embrechts, F. Arciniegas, M. Ozdemir, M. Momma, C. Breneman, L. Lockwood, K. Bennett, and R. Kewley, "Strip mining for molecules," in *Neural Networks, 2002. IJCNN'02. Proceedings of the 2002 International Joint Conference on*, vol. 1. IEEE, 2002, pp. 305–310.

[16] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Modeling wine preferences by data mining from physicochemical properties," *Decision Support Systems*, vol. 47, no. 4, pp. 547–553, 2009.

[17] R. Kewley, M. Embrechts, and C. Breneman, "Data Strip Mining for the Virtual Design of Pharmaceuticals with Neural Networks," *IEEE Trans Neural Networks*, vol. 11, no. 3, pp. 668–679, May 2000.

[18] M. Embrechts, F. Arciniegas, M. Ozdemir, and R. Kewley, "Data mining for molecules with 2-D neural network sensitivity analysis," *International Journal of smart engineering system design*, vol. 5, no. 4, pp. 225–239, 2003.

[19] P. Cortez, J. Teixeira, A. Cerdeira, F. Almeida, T. Matos, and J. Reis, "Using data mining for wine quality assessment," in *Discovery Science*, J. G. et al., Ed., vol. LNCS 5808. Springer, 2009, pp. 66–79.

[20] R Development Core Team, *R: A language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria, ISBN 3-900051-00-3, http://www.R-project.org, 2009.

[21] P. Cortez, "Data Mining with Neural Networks and Support Vector Machines using the R/rminer Tool," in *Advances in Data Mining – Applications and Theoretical Aspects, 10th Industrial Conference on Data Mining*, P. Perner, Ed. Berlin, Germany: LNAI 6171, Springer, Jul. 2010, pp. 572–583.

[22] A. Asuncion and D. Newman, "UCI Machine Learning Repository, Univ. of California Irvine, http://www.ics.uci.-edu/∼mlearn/MLRepository.html," 2007.

[23] I. Witten and E. Frank, *Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations*. San Francisco, CA: Morgan Kaufmann, 2005.

[24] T. Fawcett, "An introduction to ROC analysis," *Pattern Recognition Letters*, vol. 27, pp. 861–874, 2006.