A multilayer-perceptron based method for variable selection in soft sensor design

Francisco A. A. Souza[a,b,*], Rui Araújo[a,b], Tiago Matias[a,b], Jérôme Mendes[a,b]

[a]*Institute of Systems and Robotics (ISR-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal*
[b]*Department of Electrical and Computer Engineering (DEEC-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal*

Abstract

The paper proposes a new method for variable selection for prediction settings and soft sensors applications. The new variable selection method is based on the multi-layer perceptron (MLP) neural network model, where the network is trained a single time, maintaining low computational cost. The proposed method was successfully applied, and compared with four state-of-the-art methods in one artificial dataset and three real-world datasets, two publicly available datasets (Box Jenkins gas furnace, and gas mileage), and a dataset of a problem where the objective is to estimate the flouride concentration in the effluent of a real urban water treatment plant (WTP). The proposed method presents similar or better approximation performance when compared to the other four methods. In the experiments, among all the five methods, the proposed method selects the lowest number of variables and variables-delays pairs to achieve the best solution. In soft sensors applications having a lower number of variables is a positive factor for decreasing implementation costs, or even making the soft sensor feasible at all.

Keywords: Variable selection, Soft sensors, Multilayer perceptron

1. Introduction

Data-driven soft sensors are inferential models that use easy-to-measure variables (from online available process sensors), for online estimation of hard-to-measure variables, i.e. variables which cannot be automatically measured at all, or can only be measured at high cost, sporadically, or with high delays (e.g. laboratory analysis) [1, 2, 3, 4, 5]. These models are based on measurements which are recorded and provided as historical data. Example of data driven models are support vector regression, multilayer perceptron models and least squares regression.

Nevertheless, when building a data driven soft sensor model to predict a hard-to-measure variable, it is not necessarily true that all the available easy-to-measure variables are relevant or useful. Then, it would be important to learn the data driven model using only relevant variables[1]. Normally, the selection of relevant variables is done by manual selection, by system experts [6, 1, 7]. However, for physically large and highly integrated processes, the selection of the most relevant variables based on process insight may not be feasible [6]. In these cases it is necessary to appeal to automatic methods [6, 1, 8, 9]. Moreover, the selection of variables permits the use of a lower number of input variables, and real sensors, thus decreasing costs, and increasing or enabling feasibility of applications. Furthermore, variable selection can decrease the complexity of the soft sensor prediction model, leaving it less prone to overfitting [1].

This paper proposes a method for variable selection applications by using a multi-layer perceptron (MLP) neural network model with two-layers (MLP-TL) as the basis (Figure 1). The method proposed in this paper is inspired in the method proposed in [10]. Differently from the method in [10], the method proposed in this paper holds on the assumption that the difference between the mean square error (MSE) of two models, one trained with all variables and the other trained with a set of relevant variables is small. Under this assumption, it is possible to justify the approximation of the output of a MLP-TL model, when a variable is removed, by using the automatic ajustment of weights used in [10]. Another point is that, in this work, an exclusion criterion different from the one used in [10] is proposed and used. This new exclusion criterion, which is detailed in Section 5, has shown to provide good results in the experimental part when compared with other four methods.

The proposed method was successfully applied, and compared with four other state-of-the-art methods, one artificial dataset and three real-world datasets, two publicly available datasets (Box Jenkins gas furnace, and gas mileage), and a dataset of a problem where the objective is to estimate the fluoride concentration in the effluent of a real urban water treatment plant (WTP).

In summary, the contributions of the this paper are: (i) A new variable selection method based on an MLP network model, with low computational costs; and (ii) application of the proposed methodologies in real industrial applications, demonstrating superior performance of the proposed methods when compared to other state-of-the-art methods.

*Corresponding author at: Institute of Systems and Robotics (ISR-UC), University of Coimbra, Pólo II, PT-3030-290 Coimbra, Portugal. Tel.: +351 910942012.

Email addresses: `fasouza@isr.uc.pt`, `alexandre.andry@gmail.com` (Francisco A. A. Souza), `rui@isr.uc.pt` (Rui Araújo), `tmatias@isr.uc.pt` (Tiago Matias), `jermendes@isr.uc.pt` (Jérôme Mendes)

[1]The term relevant variables employed here refers to a subset from whole set, where the prediction error of the model trained with this subset is less than or equal to the prediction error of the model trained with the whole set.

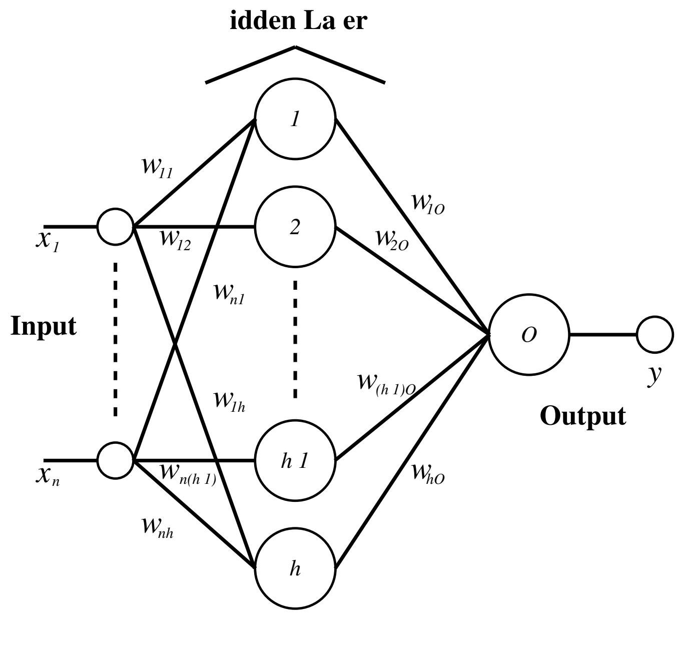

Figure 1: Topology of MLP-TL neural network; O is the output node, $\mathbf{W}_I = \mathbf{W} = [w_{ij}]$ is the $n \times h$ matrix of the weights connecting the inputs to the h hidden layer nodes, and $\mathbf{w}_O = [w_{O1}, \ldots, w_{Oh}]^T$ is the output weights vector. The hidden layer biases $\mathbf{b}_I$ and the output bias b_O are omitted to simplify the diagram.

The paper is organized as follows. Related work on variable selection is discussed in Section 3. The MLP model learning is briefly discussed in Section 4. The new MLP-based variable selection algorithm is proposed in Section 5. Section 6 presents experimental results. Finally, Section 7 gives concluding remarks

2. Notation

In this work the following notation will be used: considering a collection of variables $x_1, \ldots, x_n$, such a collection can be collectively represented or organized either as set of variables $X = \{x_1, \ldots, x_n\}$, or equivalently as a vector of variables $\mathbf{x} = [x_1, \ldots, x_n]^T$.

3. Related work

The problem of selecting relevant variables can be stated as the problem/objective of finding the optimal or suboptimal subset of the whole set of possible input variables, so that the variables in this subset can be used as inputs in a prediction setting to correctly predict the output using an adequate model [11, 12]. To find these optimal or suboptimal variables, a variable selection algorithm is built through two main components: the measure used to quantify the quality of a subset (e.g. the mean square error between the predicted output and the target output, the mutual information between input set and the target), and the variable search procedure to be used in the input space to select the best subset.

A variable search procedure can be defined as the tool to guide the selection of an optimal or suboptimal subset of variables from the whole set. To avoid the high computational cost associated with optimal search strategies, such as exhaustive search, some suboptimal search procedures have been proposed and largely used in variable selection approaches [12]. The most popular are based on greedy methods (e.g. sequential backward search (SBS) and sequential forward search (SFS)). The SBS procedure, proposed by [13], starts with all variables, and at each step one variable that contributes least to the criterion function is removed. SBS stops when a pre-specified number of variables are removed or when the results get satisfactory. The SFS, introduced by [14], starts with an empty subset, and at each step the variable that mostly contributes to increase the criterion function is added to the set of chosen variables. Random search and evolutionary search methods, have been also employed in variable selection tasks [15].

In [16] it was proposed the "minimum redundancy maximum relevance" (mRMR) criterion which attempts to capture variables which have the maximum relevance to the output y_d and, at the same time, to reduce the redundancy between selected variables. This is performed in terms of the mutual information, and the method is independent of the prediction model to be used. According to the mRMR approach, the next feature x_m chosen for inclusion in the set S of selected variables is the following:

$$x_m = \arg \max_{x_j \in \{X \setminus S_{m-1}\}} \left[I(x_j, y_d) - \underbrace{\frac{1}{m-1} \sum_{x_i \in S_{m-1}} I(x_j, x_i)}_{R} \right], \qquad (1)$$

where X is the set of input variables/features, S_{m-1} is the set that contains the $m-1$ features previously selected, $I(x_i, x_j)$ is the mutual information between input variables x_i and x_j, and y_d is the output target variable. The first selected feature is the one that has more information about the output y. A recent study using mutual information to select the best variables was presented in [17] and is called normalized mutual information feature selection (NMIFS). The NMIFS criterion changes the form of how the mRMR criterion is defined. It uses the following normalized version as the redundancy measure:

$$R = \left(\frac{1}{m-1} \sum_{x_i \in S_{m-1}} \frac{I(x_j, x_i)}{\min\left\{H(x_i), H(x_j)\right\}} \right), \qquad (2)$$

where $H(x_i)$ is the entropy value for feature x_i. The NMIFS redundancy value (2) can be replaced for R in the right side of (1). The mRMR and NMIFS methods are the state-of-the-art and most commonly used filter methods for variable selection.

MLP models using the sum of squared prediction error (SSE) as the cost function [18, 19] are widely used for variable selection. In these methods, variable selection is generally based on SBS elimination [19], and it will be refereed as SBS-MLP. The original SBS-MLP variable selection procedure starts with a MLP trained with all variables. Then, useless or less relevant variables are removed in a sequential backward selection procedure based on the prediction error when using and not using a specific feature (when a variable is removed the network is re-trained): it is removed the variable that maintains on smallest SSE on the model after its removal. This method needs to retrain the network $(n(n+1)/2)$ times in the worst case, where n is the input dimensionality.

To overcome the problem of requiring to retrain the network $(n(n+1)/2)$ times in the SBS-MLP case, some methods were proposed in the literature to select the variable to be removed without retraining the network for each variable that is removed. Generally, these methods are based on the recursive feature elimination (RFE) approach, which can be thought as a variant of the SBS procedure, where the model is trained and the variables are ranked according to a certain metric of importance. Thus, a certain number of variables is either selected or discarded according to a pre-selected rule. The model is re-trained and variables are once again ranked; this process continues until some criterion is reached.

Possible metrics to be used are based on sensitivity analysis [20, 21]: the zero-order, first-order methods. Zero-order methods use cost functions based on the MLP input connection weights. In first-order methods the cost function is computed as $\partial y_d/\partial x_i$, and it is interpreted as the perturbation of output variable y_d as a function of the input variables x_i. However, in these cases, when a variable is removed, the network should still be re-trained: this involves training the network $(n-1)$ times, in the worst case. In [10] a MLP network is trained a single time, and the SBS procedure is used to search the best subset of variables. However, instead of retraining the network for each variable that is removed, an adjustment factor is used avoiding the retraining of the network several times. The quality of each variable is based on a zero-order cost function.

4. MLP network learning

This section reviews the multilayer perceptron (MLP) neural network learning. A MLP neural network model with two-layers, Figure 1, and a sufficient number of neurons h and proper weights can uniformly approximate any continuous function to any accuracy [22]. It is a parametrized model which has the following form:

$$f(\mathbf{x};\boldsymbol{\theta}) \;=\; l\Big(g\left(\mathbf{x}^T\mathbf{W}_I + \mathbf{b}_I\right)\mathbf{w}_O + b_O\Big), \tag{3}$$

where $f(\mathbf{x};\boldsymbol{\theta})$ is the MLP network output, $\boldsymbol{\theta}$ is the set of weight and bias parameters, $\mathbf{x}$ is the input vector, $\mathbf{W}_I = \mathbf{W} = [w_{ij}]$ is the $n \times h$ matrix of the weights connecting the n inputs to the h hidden layer nodes, $\mathbf{b}_I = \mathbf{b} = [b_1, \ldots, b_h]$ is the vector of biases of the hidden layer nodes. The output weights that connect the hidden neurons with the output neuron and the output bias are represented by $\mathbf{w}_O = [w_{O1}, \ldots, w_{Oh}]^T$ and b_O, respectively. $g(\cdot)$ and $l(\cdot)$, represent the activation functions of the nodes of the hidden layer, and output layer, respectively. In this work $g(\cdot)$ is tangent sigmoid, and $l(\cdot)$ is a linear function.

Given a set of training data $\mathcal{F} = \{(\mathbf{x}(k), y_d(k))\ ;\ k = 1, \ldots, m\}$, the set of weight parameters $\boldsymbol{\theta}$ is chosen so as to minimize the *empirical risk*,

$$R_{emp}\left[f(\mathbf{x};\boldsymbol{\theta}), y_d\right] = \frac{1}{m}\sum_{k=1}^{m} J\left[f(\mathbf{x}(k);\boldsymbol{\theta}), y_d(k)\right], \tag{4}$$

where $J(\cdot)$ is a loss function. Using the approximation error

$$\varepsilon = y_d - f(\mathbf{x};\boldsymbol{\theta}), \tag{5}$$

and defining the *loss function* as

$$J\left[f(\mathbf{x};\boldsymbol{\theta}), y_d\right] = \varepsilon^2 = \left[f(\mathbf{x};\boldsymbol{\theta}) - y_d\right]^2, \tag{6}$$

then the *empirical risk* $R_{emp}\left[f(\mathbf{x};\boldsymbol{\theta}), y_d\right]$ becomes the well know mean square error (MSE),

$$E_{mse}(y, y_d) = \frac{1}{m}\sum_{k=1}^{m}\left[y(k) - y_d(k)\right]^2, \tag{7}$$

which is used in this work, and where for simplicity the predicted output is defined as $y(k) = f(\mathbf{x}(k);\boldsymbol{\theta})$. The MLP is trained to minimize the MSE using the Levenberg-Marquardt algorithm [23].

5. Proposed variable selection algorithm

In this section, the proposed method is derived. As discussed before, the method is inspired in the method proposed in [10], called here as Iteratively Adjusted Neural Network (IANN). Essentially, the IANN method makes use of a trained neural network model, with all inputs, and sequentially removes the useless variables according to a proposed exclusion criterion in a SBS procedure. The exclusion criterion, described in Section 5.3, ranks the input variables according to their influence to the prediction of the output. However, when a variable is removed the IANN performs an adjustment of the existing model instead of retraining again all the network.

The method proposed in this paper is based on the IANN idea, but differs mainly on the exclusion criterion. This new exclusion criterion is based on the empirical risk, and at each iteration the variable selected to be removed is the one which contributes least to predict the target output. The proposed criterion is detailed below in this section, and it has shown to provide good results in the experimental part when compared with other four methods. Another, smaller, difference is that, based on the assumption that the difference between the mean square error (MSE) of two models, one trained with all variables and the other trained with a set of relevant variables is small, it is possible to justify the approximation of the output of a MLP-TL model, when a variable is removed, by using the automatic adjustment of weights used in the IANN method [10].

5.1. Motivation

The variable selection proposed here, holds on the following assumption:

Assumption 1. *Let the difference between the MSE of two MLP-TL models, both with h hidden neurons, and one trained with all variables and the other trained with a subset of variables, be equal to some value ϵ:*

$$E_{mse}^{all} - E_{mse}^{rel} = \epsilon, \tag{8}$$

where E_{mse}^{all} and E_{mse}^{rel} are the mean square error of the model trained with all variables, and with the subset of variables, respectively. Then, assume that $|\epsilon|$ is small if and only if (iff) the subset of variables is a sufficient subset of relevant variables. A subset of relevant variables is sufficient iff it contains all the necessary input variables which are required to correctly predict the target variable.

From the above assumption, the error of an MLP-TL model trained with all variables provides similar results when compared with the error of an MLP-TL model trained with a subset of sufficient relevant variables.

The main characteristic of the proposed method is the automatic adjustment of the network when a variable is removed, in contrast to retraining the network. The goal is to reduce the computation costs. The following subsection describes the proposed MLP network adjustment method.

5.2. MLP network adjusment

Assume a MLP-TL model trained with a dataset $\mathcal{F}$ by minimizing the mean square error (7) to obtain an approximator of the form (3). When a variable x_r is removed from the set X, it generates a new/derived dataset $\mathcal{F}_{-r} = \left\{\left(\mathbf{x}_{-r}^T(k), y_d(k)\right) \ ; \ k = 1, \ldots, m\right\}$, where $\mathbf{x}_{-r}^T$ is the input vector without variable r and $X_{-r} = X \setminus \{x_r\}$. Retraining a MLP-TL network using the $\mathcal{F}_{-r}$ dataset with h hidden neurons, the following model is generated:

$$y_{-r} = l\left(g\left(\mathbf{x}_{-r}^T \mathbf{W}_{I,-r} + \mathbf{b}_{I,-r}\right)\mathbf{w}_{O,-r} + b_{O,-r}\right), \tag{9}$$

where $\mathbf{W}_{I,-r}$, $\mathbf{b}_{I,-r}$, $\mathbf{w}_{O,-r}$ and $b_{O,-r}$ are the new matrix of input weights, input bias, output matrix weights and output bias, respectivelly, and with h hidden layer nodes.

Using Assumption 1, the error $E_{\text{mse}}(y_d, y) - E_{\text{mse}}(y_d, y_{-r}) = \epsilon$ is small if variable r is irrelevant. On the other hand, if variable r is relevant the error ϵ will be large. Then, the following approximation holds:

$$E_{\text{mse}}(y_d, y) = E_{\text{mse}}(y_d, y_{-r}) + \epsilon \tag{10}$$

$$\frac{1}{m}\sum_{k=1}^{m}\left[y(k) - y_d(k)\right]^2 = \frac{1}{m}\sum_{k=1}^{m}\left[y_{-r}(k) - y_d(k)\right]^2 + \epsilon \tag{11}$$

$$\sum_{k=1}^{m}\left[y(k) - y_d(k)\right]^2 = \sum_{k=1}^{m}\left[(y_{-r}(k) + \hat{\epsilon}(k)) - y_d(k)\right]^2, \tag{12}$$

$$\sum_{k=1}^{m} y(k) = \sum_{k=1}^{m}(y_{-r}(k) + \hat{\epsilon}(k)), \tag{13}$$

where the values $\hat{\epsilon}(k)$ in (13) are the contributions of the individuals errors of each sample, so that $y_{-r}(k) + \hat{\epsilon}(k) = y(k)$, and so that (11) is valid. For a sample k and replacing (9) in (13):

$$l\left(g\left(\mathbf{x}(k)^T\mathbf{W}_I + \mathbf{b}_I\right)\mathbf{w}_O + b_O\right) \\ = l\left(g\left(\mathbf{x}_{-r}^T\mathbf{W}_{I,-r} + \mathbf{b}_{I,-r}\right)\mathbf{w}_{O,-r} + b_{O,-r}\right) + \hat{\epsilon}(k) \tag{14}$$

which means that for a sample k, the output of one model is equal to the output of the other plus a error $\hat{\epsilon}(k)$. Setting the output weights of the right side of equation (14), equal to $\mathbf{w}_{O,-r} = \mathbf{w}_O$ and $b_O = b_{O,-r}$ and inserting the error $\hat{\epsilon}(k)$ into the first layer of MLP-TL model, Eq. (14) becomes:

$$l\left(g\left(\mathbf{x}(k)^T\mathbf{W}_I + \mathbf{b}_I\right)\mathbf{w}_O + b_O\right) \\ = l\left(g\left(\mathbf{x}_{-r}^T\mathbf{W}_{I,-r} + \mathbf{b}_{I,-r} + \Delta(k)\right)\mathbf{w}_O + b_O\right). \tag{15}$$

where $\Delta(k)$ is the $h \times 1$ vector representation of $\hat{\epsilon}(k)$ into the first layer of the MLP-TL model, so that equation (14) is valid.

Then, through Eq. (15) it is possible to update the weights without retraining the network. After a variable x_r and the associated weights w_{rj} ($j = 1, \ldots, h$) are removed, the remaining weights w_{ij} are adjusted using factors δ_{ij} obtained from the following $h \times m$ equations:

$$\sum_{x_i \in X}\left[x_i(k)w_{ij}\right] = \sum_{x_i \in X_{-r}}\left[x_i(k)(w_{ij} + \delta_{ij})\right], \\ j = 1, \ldots, h, \quad k = 1, \ldots, m, \tag{16}$$

where for all i such that $x_i \in X_{-r}$, and $j = 1, \ldots, h$, δ_{ij} is the adjustment to weight w_{ij} that connects input x_i to hidden neuron j. As can be noticed in (16), the δ_{ij} are the adjustment factors for the remaining weights, so that the input to each node in the hidden layer remains constant. Simple algebraic manipulations of (16) yield:

$$\sum_{x_i \in X_{-r}} \delta_{ij}x_i(k) = w_{rj}x_r(k), \\ j = 1, \ldots, h, \quad k = 1, \ldots, m. \tag{17}$$

A way to solve (17) for the unknowns δ_{ij} (for all i such that $x_i \in X_{-r}$, $j = 1, \ldots, h$) is to use the conjugate gradient precondition normal equation (CGPCNE) method [24], which provides a good and fast least-squares solution.

Thus, y_{-r} (9) can be approximated as follows without retraining the network, but instead by adjusting the input weights:

$$y_{-r}^* = l\left(g\left(\mathbf{x}_{-k}^T\mathbf{W}_{I,-r}^* + \mathbf{b}_I\right)\mathbf{w}_O + b_O\right), \tag{18}$$

where y_{-r}^* is the new output prediction, and $\mathbf{W}_{I,-r}^*$ is the matrix of input weights updated according to the δ_{ij} obtained from (17).

The computation complexity per iteration of the MLP-TL training by the backpropagation algorithm is proportional to the total number of network weights $|W|$ and the number of samples m, so that the overall complexity is equal to $O(em|W|)$, where e is the number of training iterations. On the other hand, the computation complexity of each iteration of the CGPCNE to determine the δ_{ij}'s values is proportional to the number of input weights and the number of available samples. Generally, the number of iterations needed to solve Eq. (17) is very small, in a way that the overall computation complexity associated with the CGPCNE algorithm is $O(m|W_I|)$, where $|W_I|$ is the total number of input weights. Clearly, adjusting the network with the δ_{ij} is much cheaper than to retrain the MLP several times.

5.3. IANN ranking criterion

In the IANN algorithm the variable x_r selected to be removed in each iteration is the one that contributes least to the output prediction, and it is assumed that important input variables have large weights. Thus, the variable selected to be removed x_r at each iteration in the one that minimizes the following criterion:

$$x_r = \arg\min_{x_i \in X} \sum_{j=1}^{h} \sum_{k=1}^{m} \left[x_i(k) w_{ij} \right]^2 . \tag{19}$$

This variable can be interpreted as the input having the smallest total amount of feedforward propagated information, or it can also be seen as the variable x_r with the smallest energy with respect to the trained network. However, this method has the disadvantage of failing to remove redundant variables, which is very common in soft sensor applications, leading to poor performance when the variables in the input set are redundant.

5.4. Proposed ranking criterion

The importance of a variable x_r can be measured by considering whether the removal of x_r reduces or increases the *empirical risk* (7). A reduction indicates that the absence of x_r is irrelevant to the model and an increase suggests that it is relevant to the model. In this way, the exclusion evaluation function of x_r is defined as the following difference:

$$\mathcal{E}_{-r} \;=\; E_{mse}(y_{-r}, y_d) - E_{mse}(y, y_d), \tag{20}$$

where y_d is the desired output. The above equation computes the difference of the empirical risk in the presence and absence of x_r, so the higher the value of $\mathcal{E}_{-r}$, the more important is the r-th variable. Moreover, a negative value of $\mathcal{E}_{-r}$ indicates that x_r is irrelevant to the model.

In the proposed method, $E_{mse}(y^*_{-r}, y_d)$ is used instead of $E_{mse}(y_{-r}, y_d)$ in (20). This means that instead of retraining the network to obtain y_{-r} (9), the input weights are adjusted using the method described in Subsection 5.2, generating model y^*_{-r} (18). Thus, the exclusion evaluation function (20) is redefined as:

$$\mathcal{E}^*_{-r} \;=\; E_{mse}(y^*_{-r}, y_d) - E_{mse}(y, y_d), \tag{21}$$

The proposed variable selection method is detailed in Algorithm 1, where the loop starts with a trained MLP-TL network $M(\mathcal{F})$. At each iteration of the loop, the importance measure of every variable x_j is calculated. This is done by temporarily removing x_j from the dataset $\mathcal{F}$, readjusting the network using (16) and subsequently measuring the importance of x_j using (21). In each iteration of the proposed method algorithm, the least important variable x_r is selected and removed from the MLP-TL network. Then, the MLP-TL is readjusted according to the removal of x_r (retaining the most favorable network). At the output of the algorithm, the set S contains the input variables ranked (ordered in a selection rank) according to the exclusion evaluation function (21).

Algorithm 1 Steps of the proposed variable selection scheme

```
1: Input: Dataset: $\mathcal{F} = \{(\mathbf{x}(k), y_d(k))\ ;\ k = 1, \ldots, m\}$; Set of variables: $X$; Ordered set of variables: $S := \emptyset$;
2: Output: Ordered set $S$, containing the features ranked according to their importance;
3: Set $M(\mathcal{F}) \leftarrow$ "A MLP-TL trained with dataset $\mathcal{F}$";
4: for $k = n$ down to 1 do
5:     For each $x_j \in X$, compute $\mathcal{E}^*_{-j}$, Eq. (21), and let $x_r = \arg\min_{x_j \in X} \left(\mathcal{E}^*_{-j}\right)$.
6:     $X \leftarrow X \setminus \{x_r\}$, "Remove variable $x_r$ that has the lowest value of $\mathcal{E}^*_{-r}$".
7:     Set $S \leftarrow S \cup \{x_r\}$, "Update the set $S$, adding $x_r$ in the $k$-th position".
8:     Set $M(\mathcal{F}) \leftarrow$ "MLP-TL network updated by removing the $x_r$ input and adjusting the remaining weights according to (16), and $w_{ij}^{(\mathrm{new})} = w_{ij}^{(\mathrm{old})} + \delta_{ij}$".
9: end for
```

Dataset	$\lvert U \rvert$	Train/Test	Architecture	# Epochs
Friedman	10	250/250	$K-8-1$	500
Box-Jenkins	2	145/145	$K-4-1$	80
Gas-Mileage	6	196/196	$K-3-1$	100
WTP	11	176/176	$K-10-1$	200

Table 1: Data Sets Description. $|U|$, is the number of variables in the original input vector. Train/Test indicates the number of exemplars in the corresponding dataset. Architecture indicates the number of input, hidden, and output nodes used in the MLP-TL, where $K = n, \ldots, 1$, according to Algorithm 1.

6. Experimental results

This section presents experimental results in one artificial dataset, and in three real-world datasets, two publicly available datasets (Box Jenkins gas furnace, and gas mileage), and a dataset of a problem where the objective is to estimate the fluoride concentration in the effluent of a real urban water treatment plant (WTP). For comparison purposes, the following variable selection algorithms were implemented: mRMR [16], NMIFS [17], IANN [10], and the original SBS-MLP [13, 19].

6.1. Experimental settings

For all algorithms and datasets, half of the available data was used for training and the other half was used for test. All considered MLP networks have one hidden layer with a tangent hyperbolic activation function and a linear activation function in the output layer, and are trained with the Levenberg-Marquardt backpropagation algorithm [23] in batch mode. The weights were initialized using the Nguyen-Widrow method [25].

The optimal number of hidden neurons h used in all methods was determined by training the MLP-TL model with all variables in a 10-fold cross-validation [26] scheme using the training data set, and the selected number of hidden neurons was the one that produced the highest average cross-validation accuracy among these ten realizations. Table 1 describes the

	x_1	x_2	x_3	x_4	x_5	x_6	x_7	x_8	x_9	x_{10}
Prop.	✓	✓	✓	✓	✓	×	×	×	×	×
IANN	✓	✓	✓	✓	✓	✓	×	×	✓	✓
SBS-MLP	✓	✓	✓	✓	✓	×	×	×	×	×
NMIFS	✓	✓	✓	✓	✓	×	×	×	×	×
mRMR	✓	✓	✓	✓	✓	×	×	×	×	×

Table 2: Set of the selected of input variables by all methods in the Friedman dataset. The selected variables are marked with a tick.

	Prop.	IANN	SBS-MLP	NMIFS	mRMR
MSE	0.007	0.008	0.007	0.007	0.007
CC	0.972	0.967	0.972	0.972	0.972
Time [s]	10.87	1.37	49.54	0.01	0.01
$\|S\|$	5	8	5	5	5

Table 3: Performance results on the test dataset of the five methods on the Friedman dataset: MSE, CC, computation time (Time), and number of selected variables, $|S|$.

datasets, and parameters used in the experiments.

The computation of the respective exclusion criterion in the MLP-TL model for use in the SBS-MLP algorithm and in the proposed algorithm, was performed by using a 10-fold cross-validation scheme using the training data set. The values considered to decide the exclusion were the average cross-validation of the respective criteria, among these ten realizations.

The approximation performance of the soft sensors is evaluated using the mean square error (MSE) and the correlation coefficient (CC) between predicted and desired output, in the test data and the execution time in all methods was considered as the time necessary to rank all the variables with their respective criteria.

6.2. *Experiment I - Friedman dataset*

The Friedman artificial dataset [27] consist of 10 input variables $\mathbf{x} = [x_1, x_2, \ldots, x_{10}]^T$ generated independently of each other and uniformly distributed over $[0, 1]$. The target variable y_d is a function of the first five variables:

$$y_d = 10\sin(\pi x_1 x_2) + 20(x_3 - 0.5)^2 + 10x_4 + 5x_5 + \mathcal{N}(0, 1), \quad (22)$$

where $\mathcal{N}(0, 1)$ is Gaussian noise with zero mean and unit variance. Thus, variables x_1, x_2, x_3, x_4, x_5 are relevant, while the rest are irrelevant. As the optimal subset of variables is known a priori, this dataset is an excellent way to validate the proposed variable selection method. Therefore, in this dataset the focus is the capability of all methods in the selection the relevant variables, removing the irrelevant variables.

The results of application of all methods are presented in Tables 2, 3, and in Figure 2. All the methods except the IANN method have the capability of correctly selecting the set of relevant input variables. In IANN method, the irrelevant variables x_6, x_9 and x_{10} were selected in addition to the five relevant variables. As in the proposed method, in SBS-MLP, in NMIFS, and in mRMR the set of selected input variables was the same, the

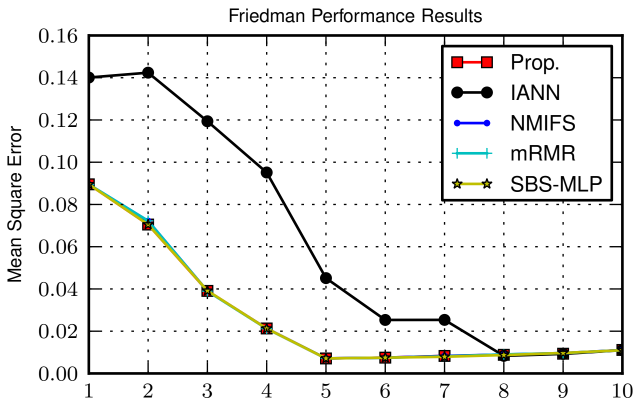

Figure 2: Error rates on the Friedman test dataset as a function of the number of top ranked variables used in the prediction model.

fitting performance in test data was similar. As expected, the selection of irrelevant variables by IANN led in a loss of fitting performance in test data. Concerning the computational time needed to select the set of input variables, the slowest method was the SBS-MLP, followed by the proposed method and by IANN. The fastest methods were the NMIFS and the mRMR methods.

6.3. *Experiment II - Box-Jenkins process dataset*

The Box-Jenkins gas furnace process data[2] was recorded from a combustion process of a methane-air mixture, and consists of 296 data points $[y(k), u(k)]$. The input $u(k)$ is the gas flow rate into the furnace and the output $y(k)$ is the carbon dioxide (CO_2) concentration in the outlet gas. The sampling interval h is 9 [s]. To predict $y(k)$, the following set of possible variables and delays is considered and examined $X = \{y(k-1), y(k-2), y(k-3), y(k-4), u(k-1), u(k-2), u(k-3), u(k-4), u(k-5), u(k-6)\}$. The number of neurons used was $h = 4$ for all methods.

The five variable selection algorithms were applied to select the best variables and respective time-lags. The variables were ranked according to their influence on the output, and the performance results are displayed in Fig. 3 in terms of MSE, where the x-axis represents the number of top-ranked input variables.

Analyzing the MSE and the correlation performance criteria in Table 4, it can be noted that the results are quite similar on the test dataset for the proposed method and the SBS-MLP. Moreover, the proposed method and the SBS-MLP method select the lowest number of variables (only five). Figure 3 shows that the proposed method attains the best performance value when the set of the top-ranked input variables has five variables. The other tested algorithms, IANN, NMIFS and mRMR, to reach the best values in the performance criteria, have selected seven or eight variables. It can be concluded in this case study that the proposed method has the same performance

[2]Provided by IEEE Neural Networks Council Standards Committee Working Group on Data Modeling Benchmarks. Available: http://www.stat.wisc.edu/~reinsel/bjr-data/gas-furnace .

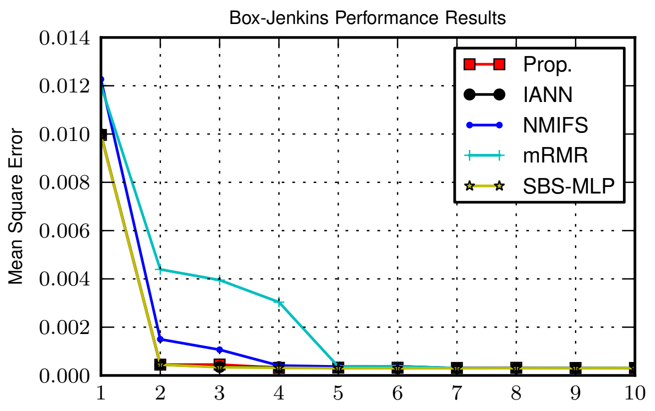

Figure 3: Error rates on the Box Jenkins test dataset as a function of the number of top ranked variables used in the prediction model.

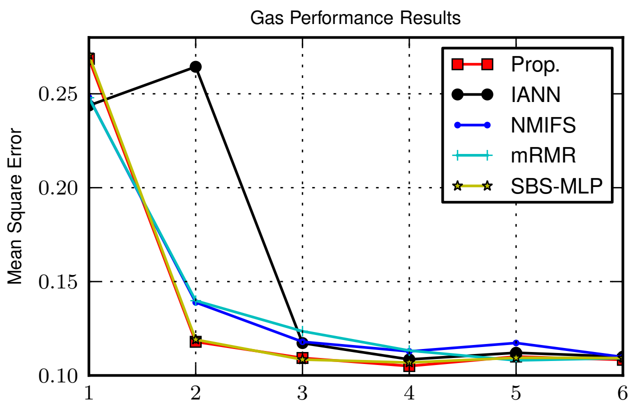

Figure 4: Error rates on the gas mileage test dataset as a function of the number of top ranked variables used in the prediction model.

	Prop.	IANN	SBS-MLP	NMIFS	mRMR
MSE	0.0015	0.0015	0.0017	0.0016	0.0016
CC	0.988	0.988	0.987	0.988	0.988
Time [s]	19.7	2.3	4264.3	0.086	0.0709
$\|S\|$	3	7	3	7	8

Table 4: Performance results on the test dataset of the five methods on the Box Jenkins dataset: MSE, CC, computation time (Time), and number of selected variables, $|S|$.

	Prop.	IANN	SBS-MLP	NMIFS	mRMR
MSE	0.105	0.113	0.104	0.114	0.117
CC	0.95	0.94	0.95	0.94	0.94
Time (s)	7.1	1.2	905	0.025	0.029
$\|S\|$	4	6	4	6	6

Table 5: Performance results on the test dataset of the five methods on the gas mileage dataset: MSE, CC, computation time (Time), and number of selected variables, $|S|$.

of the traditional SBS-MLP method, with a 216 times lower computational cost, and both methods achieve the best trade off between the lowest number of selected input features while maintaining good results in terms of MSE and correlation values. The best variables selected by the proposed method and by SBS-MLP were $\{y(t-1), x(t-3), y(t-2), x(t-6), x(t-2)\}$ and $\{y(t-1), x(t-3), y(t-2), y(t-3), x(t-2)\}$, respectively. As can be noticed, among the two sets of five selected variables there are four variable in common. This divergence can be explained because the SBS-MLP parameters are adjusted using the gradient descent algorithm, while the proposed method uses the model adjustement given by (9)-(15). Anyway, despite this difference on the selected variables, both results have equal prediction performance, while the proposed method executes the selection of variables faster than the SBS-MLP.

6.4. Experiment III - automobile Gas mileage dataset

The automobile gas mileage dataset corresponds to a problem of predicting the number of miles per gallon (MPG). It is a six input, single output regression problem. The gasoline consumption needs to be predicted based on the following input variables $u_1, u_2, u_3, u_4, u_5, u_6$, respectively: number of cylinders, displacement, horsepower, weight, acceleration and model year. The original data is available in the UCI (University of California at Irvine) Machine Learning Repository[3]. The set of considered input variables is $X = \{u_1, u_2, u_3, u_4, u_5, u_6\}$.

[3]Available: http://archive.ics.uci.edu/ml/datasets/Auto+MPG .

Despite the already small number of input variables that exist in this dataset, the prediction accuracy is highly correlated with the best variables selected, as can be seen in Fig. 4. This graphic shows that the generalization capacity, measured by the MSE, is directly related to the selected subsets. As a function of the number of best selected variables, the convergences of the proposed method and SBS-MLP methods in terms of MSE are similar, and are the fastest convergences when compared with the other methods. However, the computation time of the SBS-MLP algorithm is more than 100 times greater when compared with the proposed method. Moreover, according to Table 5, and similarly to Experiment I (Section 6.3), the prediction performance results (MSE, and CC) are quite similar for all methods. For the SBS-MLP and proposed methods the selected subsets are $\{u_4, u_6, u_2, u_5\}$. Thus, it can be concluded that the variables $\{u_4, u_6, u_2, u_5\}$ have enough representativeness for the prediction setting.

6.5. Experiment IV - water treatment plant

In the fourth experiment, the objective is to estimate the fluoride concentration in the effluent of a real-world urban water treatment plant (WTP). In this plant, the following steps are taken to treat the water: Pre-treatment, Coagulation, Flocculation, Sedimentation, Filtration and Disinfection. A short summary about the process is given as follows. Raw water is pre-treated prior to the main processes within the WTP. The Pre-treatment done in the plant are the algae control, and treatment to remove metals such as manganese and iron, where the later is done through the addition of chlorine. The Coagula-

Variable	Description
u_1	Chlorine in the raw water;
u_2	Chlorine in the effluent;
u_3	Turbidity in the raw water;
u_4	Turbidity in the coagulated water;
u_5	Turbidity in the effluent;
u_6	pH in the raw water;
u_7	pH in the coagulated water;
u_8	Ph in the effluent;
u_9	Color in the raw water;
u_{10}	Color in the coagulated water;
u_{11}	Color in the effluent;
y_d	Fluoride in the effluent.

Table 6: Variables of the water treatment plant dataset.

tion and Flocculation stages are designed to help the removal of dissolved and suspended particles, causing water clarification (i.e. removal of turbidity). During Coagulation, chemical coagulants are added to raw water aiming to neutralize the electrical charges of the fine particles present in the water. After Coagulation, the water is gentle mixed during the Flocculation stage, facilitating the agglomeration of fine particles, so generating flocs which can then be easily removed in the subsequent stages. The Sedimentation stage prepares the water for effective filtration, by allowing the flocs to settle by gravity. After sedimentation, only small unsettled particles remain in the water, and are then removed in the Filtration stage. In Filtration, the suspended particles from water, and micro-organisms in general, are removed by passing the water through a filter, such as sand. As the water passes through the filter, floc and impurities get stuck in it and the clean water goes through. The clean water from the Filtration stage is then treated with chloride in the Desinfection stage. After the last stage, the amount of fluoride in the water is determined. The value of concentration of fluoride in the effluent water is necessary to proceed with its correction (which is done by adding more fluoride into the water; this procedure is know as fluoridation).

The fluoride is a normal constituent of natural water samples and its concentration in the input of WTP, on raw water, is constant. However, during the water treatment process, the concentration of fluoride in the water decreases, which is caused by the cleaning process. The value of fluoride in the effluent is measured in laboratory once every day, and the objective of the methodology here proposed is to provide the fluoride concentration value at each 2 hours using a soft-sensor. The major concern about this problem is to know what are the best input variables and respective delays for the soft sensor. The dataset of plant variables that is available for learning consists of 11 input variables, $U = \{u_1, \dots, u_{11}\}$, one target output variable to be estimated, y_d, and 352 exemplars/samples. The variables correspond to physical values, such as pH, turbidity, color of the water and others. Table 6 presents further details about the variables.

The WTP is a long duration process, where the incoming water (called raw water) goes to the influent point, and it takes about 24 [h] to reach the effluent point which is the point measurement of the fluoride. The sampling interval for the variables measured by sensors is 2 [h].

	Prop.	IANN	SBS-MLP	NMIFS	mRMR
MSE	0.0515	0.0612	0.0516	0.0574	0.0568
CC	0.8657	0.8406	0.8708	0.8539	0.8572
Time [s]	2213.3	57.4	42049.1	12.68	10.57
$\lvert S\rvert$	6	28	5	14	18

Table 7: Performance results of the five methods the WTP test dataset: MSE, CC, computation time (Time), and number of selected variables, $|S|$.

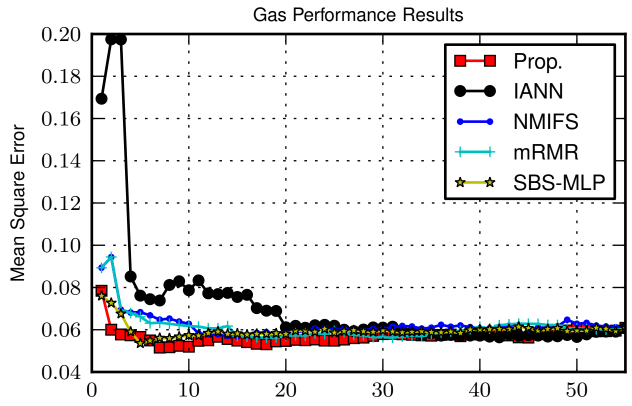

Figure 5: Error rates on the WTP test dataset, as a function of the number of top ranked variables used in the prediction model.

Thus, for the variables which are measured at the point of raw water influent (u_1, u_3, u_4, u_6, u_7, u_9, and u_{10} - Table 6), the possible time-lags must be considered within a range of 18-26 [h]. Let $n_x(j)$ be the set of possible time-lags, measured in time samples, for variable u_j ($j = 1, \dots, 11$). Then, $n_x(j) = \{9, 10, 11, 12, 13\}$, for $j = 1, 3, 4, 6, 7, 9, 10$. For those variables measured at the effluent point (u_2, u_5, u_8, and u_{11}), the possible time-lags are considered to be in a range of 0-8 [h]. Thus, $n_x(j) = \{0, 1, 2, 3, 4\}$, for $j = 2, 5, 8, 11$. Then, the size of the input set becomes equal to $|X| = 55$.

The five variable selection algorithms were applied and the results are presented in Table 7. The proposed method and the SBS-MLP algorithm have both similar results in terms of prediction performance as measured by the CC and MSE on the test data set, but the proposed method has a much lower computatinal time. The worst variable selection algorithm, in terms of CC and MSE, was the IANN, followed by the NMIFS and mRMR. Figure 5 shows that both the proposed method and SBS-MLP converge fast (in terms of the number o top-ranked variables) to the best solution, while the other methods require more variables to converge to a solution of similar prediction performance.

Moreover, with respect to the number of selected variables and also in the plot of error rates versus the number of top-ranked variables (Figure 5), it is possible to note the slight difference between the results of the SBS-MLP and the proposed method, which contrasts with the results of the previous experi-

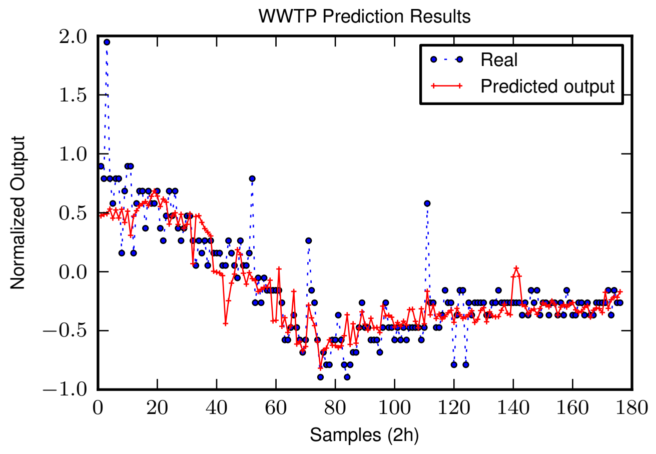

Figure 6: Fluoride prediction for the WTP test dataset. The sampling interval is 2 [h].

ments. However, this divergence is plausible because of the different number of input variables in the experiments. The Box-Jenkins and gas mileage data sets have 10 and 6 input variables, respectively, and the the WTP data set has 55 input variables. When working with a large number of input variables, as in the case of the WTP problem, the variable selected to be removed at each iteration can differ between the proposed method and the SBS-MLP. This happens because in the SBS-MLP all parameters of the MLP-TL are readjusted following the gradient descent learning algorithm, while the proposed method follows another approach, which is the model adjustement by (9)-(15) and it is not possible to assure that the both adjustments of the network (by the gradient descent learning algorithm and by (9)-(15)) will produce the same parameters.

The set of selected variables in the proposed algorithm, is composed by 6 variables: $S = \{u_5(k-3), u_3(k-9), u_7(k-12), u_9(k-12), u_1(k-12), u_9(k-10)\}$. From Table 6, it is possible to see that the first two variables selected by the proposed algorithm, were the turbidity in the effluent and the turbidity in the coagulated water. These two variables are related with the quality of cleaning water during the treatment process (by measuring the difference between the turbidity in the effluent and in the coagulated water, it is possible to observe how effective was the cleaning process) and some portion of fluoride is retained during this process. The third and the fourth selected variables were the pH in the coagulated water and the Color in the raw water. Both variables, as well as the turbidity in the coagulated water (the first selected variable), are related to the Coagulation stage. In the Coagulation stage, the amount of coagulant added to the water is directly linked to the reduction of the concentration of fluoride in the water. This happens because during the Coagullation stage, a portion of fluoride sticks in the floc due to charge neutralization, and is then removed during the subsequent stages. An interesting fact is that the fifth selected variable is the amount of Cloride added to the raw water which seems to contribute to the fluoride reduction during the cleaning process. The prediction is shown in Figure 6.

7. Conclusion

In this paper a new variable selection algorithm was proposed and compared with four state-of-the-art methods. In a series of four experiments, the proposed variable selection method has been shown to be feasible and effective. It has been shown that the proposed method has similar prediction performance when compared to the traditional SBS-MLP based on MLP error algorithm, and has the advantage of having lower computation cost. The proposed method presents similar or better approximation performance when compared to the other four methods. In the experiments, among all the five methods, the proposed method selects the lowest number of variables to achieve the best solution. In soft sensors applications, having a lower number of input variables is a positive factor for decreasing implementation costs (e.g. lower numbers of hardware sensors and/or laboratory analysis), or even making the soft sensor feasible at all.

The proposed methodology is dependent on the information content on the dataset. Thus, when applying it, it is necessary to assure that the data set is as representative as possible. Another drawback is that it is not possible to assure the causal relationship among the selected variables. A way to overcome this limitation is to apply the model with the selected set of variables and to validate it through the performance analysis phase. The reliability of the method is increased when the number and representativeness of the available samples increase. Future directions of this work are to research on the implementation of the method in a online manner, further increasing the applicability.

Acknowledgments

This work was supported by Mais Centro Operacional Program, financed by European Regional Development Fund (ERDF), and Agência de Inovação (AdI) under Project SInCACI/3120/2009.

Francisco Souza has been supported by Fundação para a Ciência e a Tecnologia (FCT) under grant SFRH/BD/63454/2009.

References

[1] L. Fortuna, S. Graziani, M. G. Xibilia, Soft Sensors for Monitoring and Control of Industrial Processes, Springer, 2007.
[2] P. Kadlec, B. Gabrys, S. Strandt, Data-driven soft sensors in the process industry, Computers & Chemical Engineering 33 (4) (2009) 795–814.
[3] B. Lin, S. B. Jøgensen, Soft sensor design by multivariate fusion of image features and process measurements, Journal of Process Control 21 (4) (2011) 547–553.
[4] Y. Wu, X. Luo, A novel calibration approach of soft sensor based on multirate data fusion technology, Journal of Process Control 20 (10) (2010) 1252–1260.
[5] J. Deng, L. Xie, L. Chen, S. Khatibisepehr, B. Huang, F. Xu, A. Espejo, Development and industrial application of soft sensors with online bayesian model updating strategy, Journal of Process Control 23 (3) (2013) 317–325.

[6] K. Warne, G. Prasad, S. Rezvani, L. Maguire, Statistical and computational intelligence techniques for inferential model development: a comparative evaluation and a novel proposition for fusion, Engineering Applications of Artificial Intelligence 17 (8) (2004) 871–885.

[7] H. J. Galicia, Q. P. He, J. Wang, A reduced order soft sensor approach and its application to a continuous digester, Journal of Process Control 21 (4) (2011) 489–500.

[8] O. Ludwig, U. Nunes, R. Araújo, L. Schnitman, H. A. Lepikson, Applications of information theory, genetic algorithms, and neural models to predict oil flow, Communications in Nonlinear Science and Numerical Simulation 14 (7) (2009) 2870–2885.

[9] M. Sheikhan, N. Mohammadi, Neural-based electricity load forecasting using hybrid of GA and ACO for feature selection, Neural Computing and Applications 21 (2012) 1961–1970.

[10] G. Castellano, A. M. Fanelli, Variable selection using neural-network models, Neurocomputing 31 (1-4) (2000) 1–13.

[11] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1-2) (1997) 273–324.

[12] I. Guyon, An introduction to variable and feature selection, Journal of Machine Learning Research 3 (2003) 1157–1182.

[13] T. Marill, D. Green, On the effectiveness of receptors in recognition systems, IEEE Transactions on Information Theory 9 (1) (1963) 11–17.

[14] A. W. Whitney, A direct method of nonparametric measurement selection, IEEE Transactions on Computers C-20 (9) (1971) 1100–1103.

[15] S. Chatterjee, A. Bhattacherjee, Genetic algorithms for feature selection of image analysis-based quality monitoring model: An application to an iron mine, Engineering Applications of Artificial Intelligence 24 (5) (2011) 786–795.

[16] H. Peng, F. Long, C. Ding, Feature selection based on mutual information: Criteria of max-dependency, maxrelevance, and minredundancy, IEEE Transactions on Pattern Analysis and Machine Intelligence 27 (8) (2005) 1226–1238.

[17] P. A. Estévez, M. Tesmer, C. A. Perez, J. M. Zurada, Normalized mutual information feature selection, IEEE Transactions on Neural Networks 20 (2) (2009) 189–201.

[18] H. Liu, H. Motoda, Feature Selection for Knowledge Discovery and Data Mining, Kluwer Academic Publishers, Norwell, MA, USA, 1998.

[19] E. Romero, J. M. Sopena, Performing feature selection with multilayer perceptrons, IEEE Transactions on Neural Networks 19 (3) (2008) 431–441.

[20] M. Gevrey, I. Dimopoulos, S. Lek, Review and comparison of methods to study the contribution of variables in artificial neural network models, Ecological Modelling 160 (3) (2003) 249–264.

[21] I.-C. Yeh, W.-L. Cheng, First and second order sensitivity analysis of mlp, Neurocomputing 73 (10-12) (2010) 2225–2233.

[22] K. Hornik, M. Stinchcombe, H. White, Multilayer feedforward networks are universal approximators, Neural Networks 2 (5) (1989) 359–366.

[23] M. T. Hagan, M. B. Menhaj, Training feedforward networks with the marquardt algorithm, IEEE Transactions on Neural Networks 5 (6) (1994) 989–993.

[24] Å. Björck, T. Elfving, Accelerated projection for computing pseudoinverse solutions of systems of linear equations, BIT Numerical Mathematics 19 (2) (1979) 145–163.

[25] D. Nguyen, B. Widrow, Improving the learning speed of 2-layer neural networks by choosing initial values of the adaptive weights, in: Proceedings of the 1990 International Joint Conference on Neural Networks (IJCNN 1990), Vol. 3, 1990, pp. 21–26.

[26] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proc. 14th International Joint Conference on Artificial Intelligence, Vol. 2, Morgan Kaufmann, 1995, pp. 1137–1143.

[27] J. H. Friedman, Multivariate adaptive regression splines, Annals of Statistics 19 (1) (1991) 1–67.