Soft Computing
https://doi.org/10.1007/s00500-020-04734-w

METHODOLOGIES AND APPLICATION

LASSO multi-objective learning algorithm for feature selection

Frederico Coelho[1] · **Marcelo Costa**[1] · **Michel Verleysen**[2] · **Antônio P. Braga**[1]

Abstract
This work proposes a new algorithm for training neural networks to solve the problems of feature selection and function approximation. The algorithm applies different weight constraint functions for the hidden and the output layers of a multilayer perceptron neural network. The LASSO operator is applied to the hidden layer; therefore, the training provides automatic selection of relevant features and the standard norm regularization function is applied to the output layer. Therefore, we propose a multi-objective training algorithm that is able to select the important features while solving the approximation problem.

Keywords Supervised learning · Feature selection · Multi-objective · *LASSO*

1 Introduction

Feature selection aims at minimizing redundancy among input variables and maximizing their relevance in relation to the output variable. Redundancy elimination is usually accomplished in the input space by identifying overlapping and information sharing between pairs or within groups of input variables. Relevant variables are those that are capable of discriminating output events represented by the output variable. The problem is often treated with univariate approaches, such as by ranking variables according to their discrimination ability. Although multivariate relevance indexes have been reported in the literature (Kira and Rendell 1992), the problem of finding the most relevant feature set is combinatorial and its solution can be prohibitive in higher dimensions. Present feature selection methods aim at representing the problem with polynomial-time algorithms, although global convergence to the optimal feature set cannot be guaranteed. Wrappers (Guyon and Road 2008) offer an alternative approach for solving the problem with a multivariate approach; however, sensitivity analysis and combination of input variables are still required.

Feature selection is usually accomplished prior to learning, as represented in Fig. 1a, and is often considered as preprocessing, before the actual learning model is induced, so the learned model $f(\mathbf{x}, \mathbf{w})$, may suffer from a poorly selected feature set $(x_1, x_2, \cdots, x_n) \in \mathbf{x}$. This may impose a dilemma to the feature selection and inductive learning problems, since they are accomplished independently, although they do in fact depend of each other. The wrapper approach for feature selection is based on a model that learns with the complete set of features to further discard the irrelevant ones. Again, this is a two-step approach, which is highly dependent of how well the learned model $f(\mathbf{x}, \mathbf{w})$ represents the general function $f_g(\mathbf{x})$. In case the learned function is representative, then the feature selection method can rely on $f(\mathbf{x}, \mathbf{w})$ as a wrapper; however, inducing a representative and general $f(\mathbf{x}, \mathbf{w})$ is not straightforward, since the problem itself is also characterized by a dilemma between model bias and variance (Geman et al. 1992).

Wrappers or any other approach that is based on model sensitivity should rely on a model that is capable of properly representing the data generator function. Learning should be accomplished by implicitly or explicitly trading-off error and model complexity (Braga et al. 2006; Teixeira 2000). For instance, when one searches for the proper number of

Communicated by V. Loia.

✉ Frederico Coelho
fredgfc@ufmg.br

Marcelo Costa
macosta.est@gmail.com

Michel Verleysen
michel.verleysen@uclouvain.be

Antônio P. Braga
apbraga@ufmg.br

[1] Universidade Federal de Minas Gerais, Belo Horizonte, Brazil

[2] Université Catholique de Louvain, Ottignies-Louvain-la-Neuve, Belgium

Published online: 03 February 2020

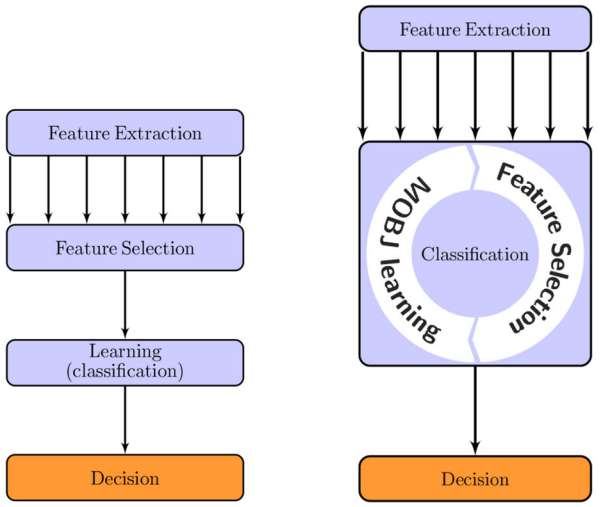
parameters of a model in a learning task, an implicit trade-off between model error and model capacity is accomplished. In many formulations of such learning problem, model capacity is represented by the magnitude of the weights, since it is also associated with the separation margin in classification and regression problems (Vapnik and Cortes, 1995a). In neural networks this learning dilemma can be treated with multi-objective optimization, which involves jointly minimizing learning set error and model structure (Braga et al. 2006; Teixeira 2000), often represented by the L_2 norm of the weight vector.

This paper presents a novel view of feature selection and learning problems, which is based on accomplishing the two tasks jointly instead of selecting features and then learning. The principle seems to find grounds on selective learning by humans, which involves jointly learning and selecting features (Broadbent 1958). However, learning from data is inherently a trade-off problem (Gacek and Pedrycz 2011; Bartlett 1997; Vapnik 1995b), multi-objective learning (MOBJ).

Braga et al. (2006) and Teixeira (2000) were adopted for inducing the classification model. The general scheme for feature selection and learning is represented in Fig. 1b.

The interaction between the two tasks represented in Fig. 1b is possible due to the layered structure of neural networks which allows for different objective functions to be minimized on each layer. The magnitude of the weights, represented by their norm, usually adopted in MOBJ learning, is a single parameter that can control smoothness response and effective capacity. L_2-norm is usually considered to represent complexity and to maximize separation margin; however, L_1-norm representation of LASSO (Tibshirani 1996a) (*Least Absolute Shrinkage Operator*) has also been adopted in the present work, since it may result on sparse solutions, parameter elimination and, of course, on feature selection. In particular, for MLPs (multilayer perceptron), weight elimination at the input layer may result on input variables selection.@@

LASSO was applied in other works in order to select features. In Rampone and Russo (2012), a method was developed to deal with databases with missing information. It is based on an algorithm that is able to infer Disjunctive Normal Form Boolean formula (DNF 1994) on low syntactic complexity variables. It defines a relevance index based on a membership function. The extended algorithm uses this function as a greedy criterion and selects the most relevant variables one at a time. Also in Yamada et al. (2014) the authors consider a feature-wise kernelized LASSO in order to identify nonlinear input–output dependency. Its main idea is to apply a nonlinear transformation in a feature-wise manner applied to particular kernel functions. By doing so, non-redundant variables strongly correlated with output variables can be found in terms of kernel-based independence measures, such as the Hilbert-Schmidt independence criterion (Gretton et al. 2005). A MOBJ learning algorithm for MLPs with different penalty functions for each layer is presented in this work. The L_1-norm penalty function is applied to the hidden (or input) layer, whereas the L_2-norm is applied to the output layer. The minimization of the two functions jointly with MSE (minimum square error) allows for complexity control and data set fitting. In addition, LASSO solutions that result from the L_1-norm penalty function, at the input layer, may also result on a minimum set of input features.



(a) Sequential approach to feature selection and learning.

(b) Select features while learning.

Fig. 1 Classification and feature selection in machine learning

This work is organized as follows: Sect. 2 reviews the MOBJ learning, while Sect. 3 deals with LASSO for feature selection. Section 4 reviews the training of MLP layers and explores the independent training for different layers. The proposed feature selection method is presented in Sect. 5. Sections 6 and 7 present the experiments and results using synthetic and real case data. Section 8 presents discussion and conclusion.

2 Multi-objective learning

Feature selection with wrappers is quite dependent of model quality and of how well $f(\mathbf{x}, \mathbf{w})$ represents the generator function $f_g(\mathbf{x})$. The poorer the model, the less representative the resulting selected features. It is well known that learning is a multi-objective problem that requires minimization of empirical and structural risks (Vapnik 1995b) or balance between bias and variance (Geman et al. 1992). The problem has, therefore, an intrinsic multi-objective nature since it involves the optimization of two conflicting objective functions (Teixeira 2000). In a more general way, multi-objective

learning can be described by introducing error and complexity objective functions $\phi_e(\omega)$ and $\phi_c(\omega)$, with the first one representing the empirical risk and the second one the structural risk. It can be formulated as the following vector optimization problem:

$$\min_{\omega \in \Omega}(\phi_e(\omega), \phi_c(\omega)), \tag{1}$$

where ω is the vector of network parameters in the parameter space Ω.

Since the two objective functions are conflicting in the region of interest, the solution of the problem in expression 1 is a Pareto-optimal front $\Omega^* \subseteq \Omega$, in which the elements $\omega^* \in \Omega^*$ satisfy the conditions

$$\forall \omega : \begin{cases} \phi_e(\omega) \geq \phi_e(\omega^*), \\ \phi_c(\omega) \geq \phi_c(\omega^*). \end{cases} \tag{2}$$

In other words, the optimization problem results on the optimal solutions that represent the best compromise between the two objectives, so for every solution $\omega \notin \Omega^*$, there are others in Ω^* that have lower complexity and error. From the Optimization perspective, the final solution should be selected from the Pareto set Ω^*, so multi-objective learning involves first generating Ω^* and then selecting one of its solutions. The original MOBJ learning algorithm (Teixeira 2000) considers $\phi_e(\omega) = \sum e^2$ and $\phi_c(\omega) = ||\mathbf{w}||^2$.

3 LASSO for feature selection

Similar to MOBJ learning, LASSO (*Least Absolute Shrinkage and Selection Operator*) (Tibshirani 1996b) is defined as a constrained optimization problem that aims at minimizing the residual sum of squares (error) subject to the sum of the absolute weights being less than a constant t.

$$\begin{aligned} \mathbf{w}^* = arg\ \min \tfrac{1}{N} \textstyle\sum_{j=1}^{N} \left(d_j - y(\mathbf{w}, \mathbf{x}_j)\right)^2 \\ subject\ to: \textstyle\sum_i |w_i| \leq t \end{aligned} \tag{3}$$

The method is very similar to the L_2 norm constrain (MOBJ) approach, however, with subtle but important differences. To illustrate the differences between the MOBJ and LASSO methods, we present a single perceptron with hyperbolic tangent activation function, one input and two weights: the input weight w and the bias weight, b. The perceptron's output equation is: $y(x_i) = tanh(w.x_i + b)$. The following patterns: $(x_i, y_i) = \{(-3, -0.4), (2, -0.9)\}$ define the training set. The error surface is shown in Fig. 2.

A perceptron with linear activation function has an elliptical error surface centered at the full least squares estimates. However, the nonlinear activation function turns the surface irregular but with a distinct minimum at $w_o = -0.207$ and

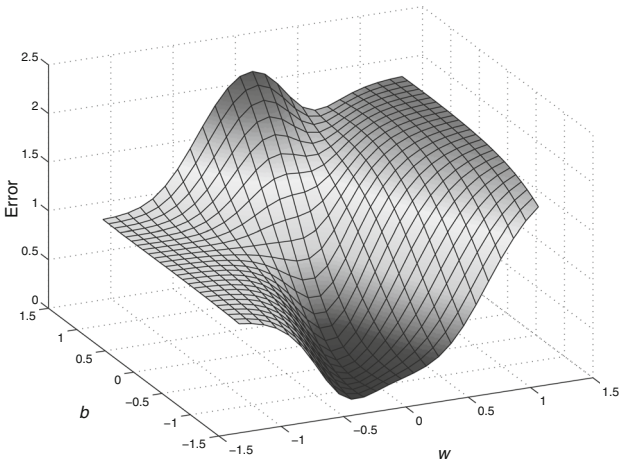


Fig. 2 Perceptron's error surface for the training set: $(x_i, y_i) = \{(-3, -0.4), (2, -0.9)\}$

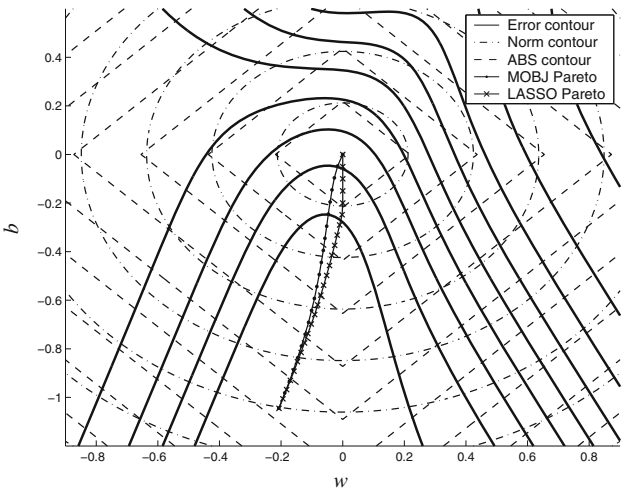


Fig. 3 Error, norm and LASSO contours

$b_o = -1.045$. Both norm and sum of the absolute weights functions have their minimum at the origin ($w = 0, b = 0$). The associated Pareto sets for norm and absolute weights functions are sets of solutions that start from origin and end at the minimum error point. To compare the solutions conditioned to the previous constraints, the error contours as well as the norm and the LASSO contours are shown in Fig. 3. The constraint region for the norm is the disk $w^2 + b^2 \leq \eta$ while that for LASSO is the diamond $|w| + |b| \leq t$. Both methods find the first point where the error contours hit the constraint regions which represent a solution with minimum error conditioned to the respective constraint. Unlike the disk, the diamond has corners; if the solution occurs at a corner, then it has one weight equal to zero. When the number of weights is larger than 2, the diamond becomes a rhomboid and has many corners, flat edges and faces, with many more opportunities for the estimated weights to be zero (Hastie et al. 2009).

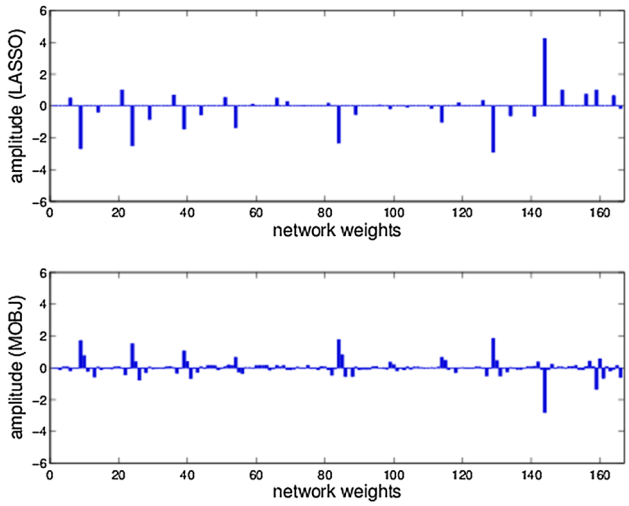


Fig. 4 Comparison between two network solutions, one obtained with LASSO and the other with MOBJ. As can be observed, LASSO's solution is sparse, with many null weights, what may result on input selection when adopted at the input layer

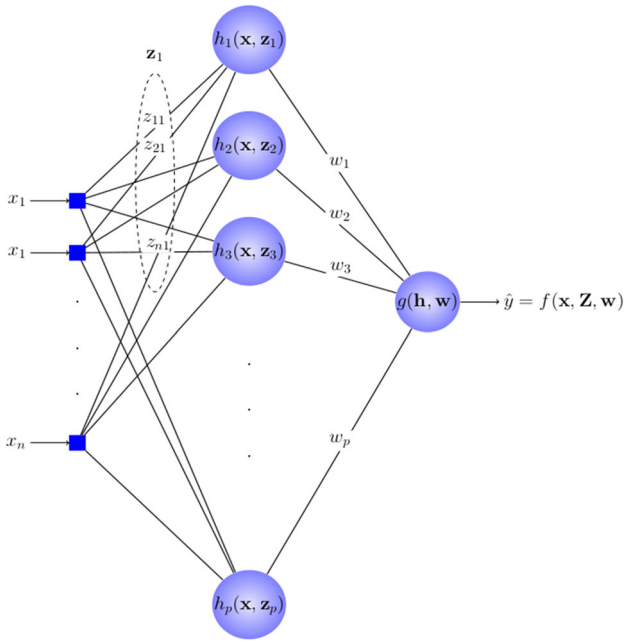


Fig. 5 General representation of a neural network of one output and two layers

Although MOBJ and LASSO methods have common solutions, the paths between origin and minimum error are quite different. Figure 3 shows that the norm solutions are nonzero for any constraint except at the origin. The LASSO approach has a subset of solutions where w is null, what may result on feature selection. As for MLPs, the LASSO approach adopted at the input layer may result on weight elimination and input variable selection. The comparison between MOBJ and LASSO presented in Fig. 4 shows the difference of sparseness between the two approaches. The figure shows the magnitudes of all weights of a MLP trained with the two approaches. As can be observed, sparseness is much higher in LASSO solutions, since it has resulted on many null weights that can be eliminated. The same does not happen with MOBJ solution, which has many solutions with small magnitudes, but not null. The approach presented in this paper suggests that LASSO is used for optimizing input weights, whereas MOBJ is adopted at the output weights, so that separation margin is maximized.

4 Dependent versus independent training of hidden and output layers

Consider, for instance, without loss of generality, that our problem is aimed at a two-layer MLP and that we aim at obtaining a large margin classifier as well as at selecting input features, as discussed in previous sections. A general schematic representation for a two-layer network structure for binary classification problems is presented in Fig. 5. Input to output mapping is accomplished by the two transformations $h(\mathbf{x}, \mathbf{Z})$ and $g(\mathbf{h}, \mathbf{w})$, where $\mathbf{Z}$ is the matrix containing all hidden layer weights and $\mathbf{w}$ is the vector of output weights. The network function can then be represented as $f(\mathbf{x}, \mathbf{Z}, \mathbf{w})$ with argument $\mathbf{x}$ and parameters $\mathbf{Z}$ and $\mathbf{w}$. In this paper, we aim at L_1 norm minimization for $\mathbf{Z}$ and at L_2 norm minimization for $\mathbf{w}$ so that feature selection is accomplished at the input layer and margin maximization at the output.

According to Fig. 5, for every input vector $\mathbf{x}_i$ a corresponding vector $\mathbf{h}_i$ is mapped into the hidden layer space, where margin maximization actually happens. Therefore, separation margin ρ is actually optimized in relation to the mapped vectors $\mathbf{h}_i$, so it depends on the separation hyperplane characterized solely by the weight vector $\mathbf{w}$ and known to be inversely proportional to the L_2 norm of the weights (Vapnik 1995b), so $\rho \propto \frac{1}{\|\mathbf{w}\|}$. Most methods, like MOBJ, that consider the norm of the weights for margin maximization minimize the weights for the augmented vector $[\mathbf{w}, \mathbf{Z}]$. Another well-known example of such an approach is regularization, which is based on the objective function $\min_\omega \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 + \lambda \|\mathbf{w}\|^2$, where λ is the regularization parameter and the weight vector $\mathbf{w}$ is composed of hidden z_{jk} and output weights w_j ($\mathbf{w} = [w_j, z_{jk}]$), i.e., $\|\mathbf{w}\|^2 = \sum w_j^2 + \sum z_{jk}^2$. Therefore, with the regularization approach, both hidden and output norms are minimized simultaneously (Haykin 2001; Vapnik and Cortes, 1995a).

Consider, therefore, that our binary classification problem characterized by the hyperplane $\mathbf{w}$ is defined in intermediate layer space as $\mathbf{w}_*^T \mathbf{h}_i \geq 0 \; \forall \; \mathbf{h}_i \in C_1$ e $\mathbf{w}_*^T \mathbf{h}_i < 0 \; \forall \; \mathbf{h}_i \in C_2$ and that the maximum margin is given by Eq. 4 (Vapnik 1995b).

$$\rho(\mathbf{w}^*) = \frac{\mathbf{h}_s \cdot \mathbf{w}^*}{||\mathbf{w}^*||} \tag{4}$$

where $\mathbf{h}_s = [h_1(\mathbf{x}_s), h_2(\mathbf{x}_s), \cdots, h_p(\mathbf{x}_s)]^T$ is the nearest vector to the hyperplane, $\mathbf{x}_s$ is the corresponding input vector and $\mathbf{w}^*$ is the maximum margin separator. Equation 4 can be expanded as follows:

$$\rho(\mathbf{w}^*) = \frac{h_1(\mathbf{x}_s, \mathbf{z}_1)w_1}{||\mathbf{w}^*||} + \frac{h_2(\mathbf{x}_s, \mathbf{z}_2)w_2}{||\mathbf{w}^*||} + \cdots + \frac{h_p(\mathbf{x}_s, \mathbf{z}_p)w_p}{||\mathbf{w}^*||} \tag{5}$$

$$\rho(\mathbf{w}^*) = \frac{h_1\left(\sum_i^n x_{si}z_{i1}\right)w_1}{||\mathbf{w}^*||} + \frac{h_2\left(\sum_i^n x_{si}z_{i2}\right)w_2}{||\mathbf{w}^*||} + \cdots + \frac{h_p\left(\sum_i^n x_{si}z_{ip}\right)w_p}{||\mathbf{w}^*||} \tag{6}$$

For small values of z_{ki}, the approximation

$$h_k(\sum_i x_{si}z_{ki}) \approx \sum_i x_{si}z_{ki},$$

could be made, so Equation 6 could be rewritten as Eq. 7 and then in Eqs. 8 and 9, since the sigmoidal functions tend to respond close to their inflection points in such a condition.

$$\begin{aligned}\rho(\mathbf{w}^*) = &\frac{x_{s1}z_{11}w_1}{||\mathbf{w}^*||} + \frac{x_{s2}z_{21}w_1}{||\mathbf{w}^*||} + \cdots + \frac{x_{sn}z_{n1}w_1}{||\mathbf{w}^*||} \\ &+\frac{x_{s1}z_{12}w_2}{||\mathbf{w}^*||} + \frac{x_{s2}z_{22}w_2}{||\mathbf{w}^*||} + \cdots + \frac{x_{sn}z_{n2}w_2}{||\mathbf{w}^*||} \\ &+\frac{x_{s1}z_{1p}w_p}{||\mathbf{w}^*||} + \frac{x_{s2}z_{2p}w_p}{||\mathbf{w}^*||} + \cdots + \frac{x_{sn}z_{np}w_p}{||\mathbf{w}^*||}\end{aligned} \tag{7}$$

$$\rho(\mathbf{w}^*) = \sum_j^p \frac{x_{s1}z_{1j}w_j}{||\mathbf{w}^*||} + \sum_j^p \frac{x_{s2}z_{2j}w_j}{||\mathbf{w}^*||} + \cdots + \sum_j^p \frac{x_{sn}z_{nj}w_j}{||\mathbf{w}^*||} \tag{8}$$

$$\rho(\mathbf{w}^*) = \sum_{i=1}^n \sum_{j=1}^p \frac{x_{si}z_{ij}w_j}{||\mathbf{w}^*||} \tag{9}$$

Furthermore, it can be shown that LASSO weights are, on average, larger than MOBJ weights. Let us first consider that both hidden and output layers are trained considering L_2-norm, and then, suppose a final multi-objective MLP solution with error e_0. This solution has a L_2 norm of r^2, i.e., $r^2 = \sum_{ij} z_{ij}^2$. Let $\mathbf{z} = (z_1, \ldots, z_p)$ be the hidden weight vector of dimension p. Using polar coordinates, the elements of vector $\mathbf{z}$ can be represented as linear combination of sines and cosines associated with a vector of angles of dimension $p-1$, $\Theta = (\theta_1, \ldots, \theta_{p-1})$:

$$z_i = \begin{cases} r\ sin(\theta_1) & i = 1. \\ r\ \prod_{k=1}^{i-1} cos(\theta_k)sin(\theta_i) & 2 \le i \le p-1. \\ r\ \prod_{k=1}^{i-1} cos(\theta_k) & i = p. \end{cases} \tag{10}$$

From Eq. 10, the absolute value of weight z_i, $|z_i|$ can be written as:

$$|z_i| = \begin{cases} r\ |sin(\theta_1)| & i = 1. \\ r\ |\prod_{k=1}^{i-1} cos(\theta_k)sin(\theta_i)| & 2 \le i \le p-1. \\ r\ |\prod_{k=1}^{i-1} cos(\theta_k)| & i = p. \end{cases} \tag{11}$$

Furthermore, vector $\mathbf{z}$ can be rearranged into many forms so that any weight z_j, $j = 1, \ldots, p$ can occupy the first position. As a consequence, the first angle θ_1 in vector Θ can be associated with any weight z_j. Now let us consider that hidden layer is trained considering LASSO norm. The LASSO constraint, $\sum_i |z_i| = r$ drives some of the weights toward zero, i.e., $sin(\theta_i) = 0, \exists i$. As mentioned, if the LASSO solution occurs at a corner of the rhomboid, then $sin(\theta_j) = 1, \exists j$ and $j \neq i$. Thus, if $sin(\theta_j) = 1$, then $|z_j| = r \cdot sin(\theta_j) = r$. For nonzero weights not located in corner of the rhomboid, $0 \le |sin(\theta_k)| \le 1$ or $0 \le |cos(\theta_k)| \le 1$, i.e., for these weights $|z_i| < r$. Therefore, weights located in the corner of the rhomboid are larger as compared to weights not located in the corner.

The weights not located in the corner can be written as an L_2 norm solution with a smaller radius $r^* < r$ given by Eq. 12

$$r^* = \begin{cases} r/|sin(\theta_1)| & i = 1. \\ r/|\prod_{k=1}^{i-1} cos(\theta_k)sin(\theta_i)| & 2 \le i \le p-1. \\ r/|\prod_{k=1}^{i-1} cos(\theta_k)| & i = p. \end{cases} \tag{12}$$

Figure 6 illustrates the differences between L_2 norm and LASSO constraints using a vector $\mathbf{z}$ with two weights, $\mathbf{z} = (z_1, z_2)$. Using polar coordinates: $z_1 = r\ sin(\theta)$ and $z_2 = r\ cos(\theta)$. Results were generated assuming $z_1^2 + z_2^2 = r^2$ and $|z_1| + |z_2| = r$ for $\theta \in \{0, \ldots, 2\pi\}$. Under these similar constraint values, LASSO solutions, located in the corner, are larger than L_2 solution located in the inner circle, which has a smaller radius $r^* = r/|sin(\theta)|$. The L_2 norm solution, not located in the corner, achieves a larger weight if the radius is larger than r, as shown in the outside circle. In general, LASSO solutions generate few nonzero weights with larger absolute values as compared to L_2 norm solutions.

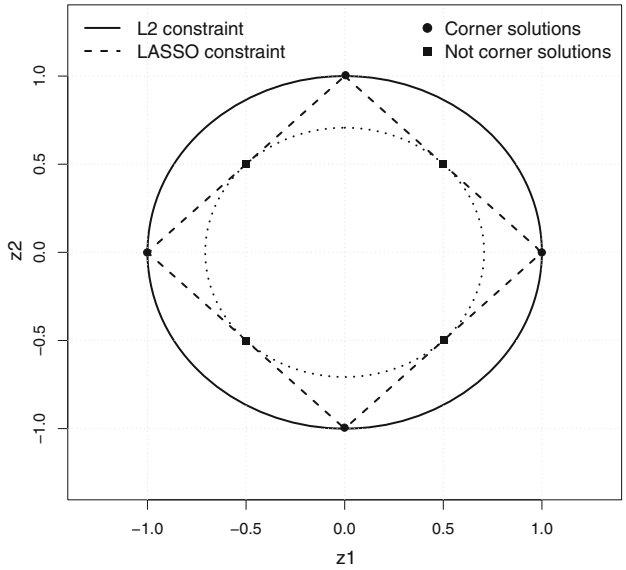


Fig. 6 Comparison of L_2 and LASSO solutions

5 Using norm and LASSO penalty functions for different layers

In addition to minimize MSE, we aim also at eliminating the input weights related to irrelevant features and to adapt the norm of the weights in the output and input layers in order to maximize generalization. Therefore, the new *MOBJ* optimization equation using LASSO and norm constraints in hidden and output layers is

$$\min_{\mathbf{w}} \sum_{i=1}^{n} \left(y_i - \hat{y}_i\right)^2 + \lambda_1 \sum_j w_j^2 + \lambda_2 \sum_{jk} |z_{jk}| \,, \tag{13}$$

where λ_1 and λ_2 are the L_2-norm and LASSO penalty parameters, respectively.

The *LASSO* term in equation 13 can be rewritten as $\sum_{jk} |z_{jk}| = \sum_k \sum_j |z_{jk}| = \sum_k |z_k|$, where $|z_k| = \sum_j |z_{jk}|$. Thus, $|z_k|$ is the sum of the absolute weights in the hidden layer related to feature k. If $|z_k| = 0$, then it can be said that the feature k is not connected to the network and, therefore, does not contribute to the output.

If feature l is found to be irrelevant during training, it can be assumed that $|z_l| = 0$. Therefore, the optimization problem can be rewritten as

$$\begin{aligned} &\min_{\mathbf{w}} \sum_i \left(y_i - \hat{y}_i\right)^2 \\ &subject\ to: \sum_{jk} |z_{jk}| \le t_1, \\ &\qquad\qquad |z_l| = 0 \\ &\qquad\qquad \sum_j w_j^2 \le t_2 \end{aligned} \tag{14}$$

Thus, if n_l features were identified as irrelevant features, then the following constraint can be applied $|z_l| = 0, \ \forall\, l = \{1, 2, \ldots, n_l\}$.

The optimization problem defined in Eq. 14 forces all weights in the hidden layer connected to irrelevant features to be confined to zero. Nevertheless, the irrelevant features are not known in advance. Thus, the optimization algorithm should select the irrelevant features. It is also worth noticing that one feature can be said to be irrelevant, conditioned on the value of t_1, that is, for smaller values of t_1, the features which were classified as irrelevant can be classified as relevant features for larger values of t_1. In practice, the constraint $|z_l| = 0$ means that the values of z_{jl} are set to zero.

In order to identify irrelevant features during training, statistical outlier detection techniques were investigated. Let $|z_k|$ be the random variable of interest. Conditioned to the value of t_1, it is expected that some k features will achieve smaller values of $|z_k|$, as compared to the remaining features. Statistical methods such as boxplot, Dixon's text, Grubbs's test, z-score, among others [18], are presented in the literature. The z-score test was used to detect the irrelevant features. Therefore, the following algorithm is proposed:

- Conditioned on the values of t_1 and t_2, the multi-objective optimization problem presented in Equation 14 is solved without excluding features.
- After convergence, the sum of the absolute weights for each feature $|z_k|$ is calculated. In sequence, the logarithm of $|z_k|$, $log|z_k|$, is calculated and normalized, i.e., the observed mean is subtracted from each value, and the remaining value is divided by the sample standard deviation.
- The features with standardized value below -2, which represents the lower bound of a normal distribution with cumulative probability of 2.5%, are classified as outliers and have their weights set to zero, $|z_l| = 0$. The remaining weights are updated using Eq. 14 until final convergence.

The logarithm of $|z_k|$ is the statistic of interest because if the constraint t_1 is close to zero, then most of the features have smaller values of $|z_k|$. In such situation there is not enough evidence to eliminate weights. As the value of t_1 increases, then weights associated with important features will increase, whether weights of irrelevant features will not increase.

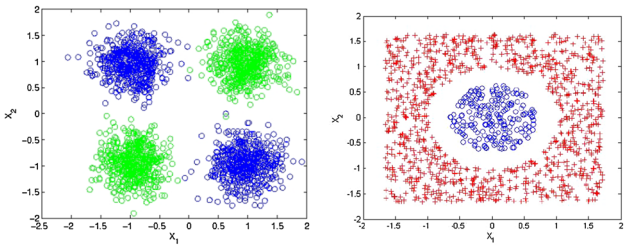
The *ellipsoidal algorithm* (Bland et al. 1980) is applied to solve the optimization problem presented in Eq. 14.

6 Experiments

Two synthetic and nine real data sets of classification problems were chosen to evaluate the performance of the proposed algorithm. The algorithm is implemented in a feed

forward multilayer perceptron network with only one hidden layer. Each hidden neuron is implemented with a hyperbolic tangent activation function and the output unit with a linear one. For each problem, the first run was done in order to set the best network parameters such as the number of neurons in the hidden layer. After network initialization, the data set is randomly split into training and test sets in a ratio of 70% and 30%, respectively. In sequence, the neural network model is trained using the proposed algorithm and the training set. The accuracy of the trained neural network is evaluated using the test set. This procedure is repeated 10 using a tenfold cross-validation scheme. For each training fold, the weights of the irrelevant features were estimated as $|z_l| = 0$ and the mean results are presented in Sect. 7. A support vector machine (SVM) (Vapnik 1992) is also used to solve the same problems using the same training and testing sets for all ten runs, in order to compare the results. First, the SVM is applied to the entire data set (all features are considered) and, in sequence, it is applied considering only the features selected by the proposed method. The available data sets are described next. All but the two synthetic problems are from Dua and Graff (2017).

- *SONAR* the SONAR data set comprises sonar responses. The task is to discriminate between sonar signals bounced off a metal cylinder and those bounced off a roughly cylindrical rock. This data set has 208 instances with 60 features.
- *PEN* the Pen-Based Handwritten Digits data set comprises digit samples from 44 different writers. We considered only instances of the digits 6 and 9. The data set has 16 features and 2,111 instances.
- *ILPD* the Indian Liver Patient Data set comprises 416 liver patient instances and 167 nonliver patient instances. The data set has 10 features.
- *IONO* the Johns Hopkins University Ionosphere database comprises 34 features with 351 patterns of radar returns from the ionosphere. Radar returns are classified into two classes: good returns showing evidence of some type of structure in the ionosphere, or signals that did not pass through the ionosphere.
- *GLASS* Data set containing examples of chemical analysis of seven glass types. Only types 1 and 2 are considered in this work. For these classes, there are 146 observations of 9 features.
- *HOUSE* This data set contains housing data for 506 census tracts from city of Boston. It has 12 features. The objective is to classify whether the house will have a median value larger than 20.000 USD per squared meter or not.



(a) Simulated XOR data set. **(b)** Simulated PCIRC data set.

Fig. 7 Simulated data sets

- *DIABETES* Data set contains 768 observations with 9 features from Indian patients. The objective is to classify patients into two classes: positive or negative for diabetes.
- *MFEAT* This data set comprises 76 features of handwritten numerals (0 to 9) extracted from a Dutch collection of utility maps. For this work, only the observations from numerals 0 and 1 are considered. The 76 features represent the Fourier coefficients of character shapes.
- *POP* The climate model simulation data set comprises 20 features and 540 observations. The goal is to predict the failure or success of a simulation outcome.
- *XOR* The data set comprises synthetic data set built with an exclusive-OR function. The problem has a total of 7 features, two important features (two dimensional problem), three additional noise features and two more features that are equal to the first, to which a random noise was added. This data set comprises 2000 instances. Figure 7a shows the two classes defined by the two important features.
- *PCIRC* The data set comprises a synthetic problem having two different classes. This problem has a total of 10 features of which the first two are the most relevant, and the last two are equal to the first two to which a random noise was added. There are eight more random features that are completely irrelevant. The problem has 2277 instances. Figure 7b shows the two classes defined by the two important features.

7 Results

Tables 1 and 2 list the results obtained for the experiments proposed in Sect. 6. The mean values and the standard deviations for accuracy are presented for the tenfold cross-validation. Table 1 also lists the average number of selected variables at the end of selected variables using the proposed methods.

Table 1 Accuracy results for LASSOMOBJ

problem	LASSOMOBJ	# selected features/ # Total
PCIRC	0.995 ± 0.003	4/10
XOR	0.992 ± 0.005	4/7
ILPD	0.711 ± 0.032	4/10
Sonar	0.740 ± 0.034	31/60
Iono	0.858 ± 0.030	33/34
Pen	0.999 ± 0.001	10/16
Glass	0.652 ± 0.093	6/9
House	0.894 ± 0.025	7/12
Diabetes	0.700 ± 0.039	5/8
Mfeat	1.000 ± 0.000	38/76
Pop	0.955 ± 0.016	8/20

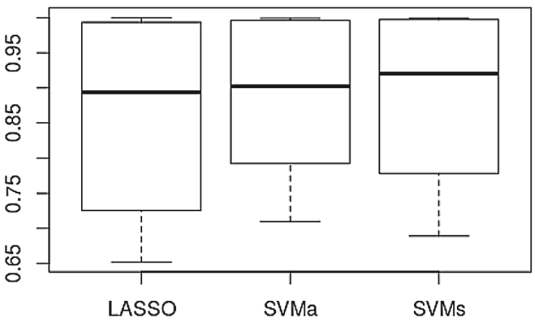


Fig. 8 Result equivalence. *LASSO* label refers to the proposed method, *SVMa* for the SVM method applied to all features and *SVMs* to only selected features

Table 2 Accuracy results for SVM

problem	SVM (all features)	SVM (selected features)
PCIRC	0.999 ± 0.001	0.999 ± 0.001
XOR	0.997 ± 0.001	0.999 ± 0.001
ILPD	0.710 ± 0.031	0.700 ± 0.056
Sonar	0.802 ± 0.064	0.846 ± 0.031
Iono	0.902 ± 0.037	0.946 ± 0.013
Pen	0.999 ± 0.001	0.999 ± 0.001
Glass	0.782 ± 0.048	0.711 ± 0.034
House	0.855 ± 0.040	0.861 ± 0.032
Diabetes	0.754 ± 0.024	0.689 ± 0.018
Mfeat	0.995 ± 0.004	0.996 ± 0.004
Pop	0.927 ± 0.015	0.920 ± 0.019

The Friedman statistical test was applied. This test considers the null hypothesis that at least one result would not be equivalent, i.e., the MOBJ-LASSO result is different from at least one result of the SVM methods. Results showing a *p-value* of 0.93 indicate the rejection of the null hypothesis, showing the equivalence between the results obtained by

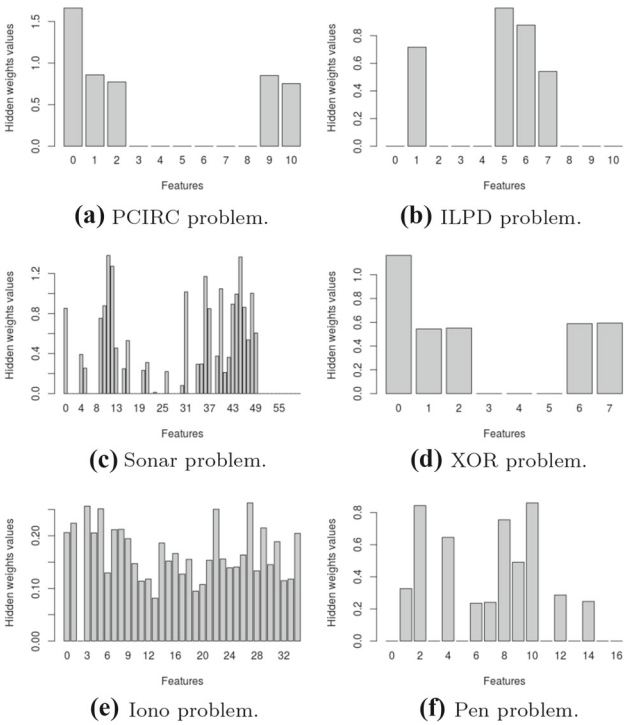


Fig. 9 Weight values assigned to selected features for each problem. Features with weight equal to zero are discarded

the two methods. Therefore, there is evidence that MOBJ-LASSO and SVM results are similar (see Fig. 8).

As shown in Tables 1 and 2 and in the result of the statistical tests, the proposed method is equivalent in performance to the benchmark method for classification problems. Basically, even in problems where the MOBJ-LASSO method does not achieve good accuracy, SVM achieves either the same result or is slightly better, but still, statistically equivalent. Nevertheless, MOBJ-LASSO was able to select features which were important for the classification problem. Consequently, MOBJ-LASSO provides a more effective understanding of the problem, since it provides variable selection.

Figure 9 illustrates the features selected using the proposed method. Features with weight of zero are identified as irrelevant and were discarded. The bar plots show the average value of the weights associated with each feature, where feature 0 comprises the *Bias* term of the hidden neuron. For the two synthetic data sets, only the most important features were selected. Note that for the PCIRC, problem features 1, 2, 9 and 10 are the most relevant and for the XOR problem features 1, 2, 6 and 7 are also the most important, as selected by the method (see Fig. 9a and d). Finally, the number of selected features was closer to the number of available features using the IONO data set.

8 Conclusion

The main objective of this work was to perform feature selection. Usually feature selection is a preprocessing step before training. This work proposes a multi-objective algorithm that selects the important variables while solving the function approximation problem. The algorithm proposed in this paper applies the L_1-norm function (LASSO operator) to the hidden layer of an MLP network so that, when performing the training, automatic selection of the relevant input variables occurs.

In general, both hidden and output weights of MLPs are adjusted in order to minimize the mean squared error function. Furthermore, a penalty function can be applied to adjust properly the MLP complexity, i.e., to improve the prediction of the MLP. One alternative to selecting features or inputs in the training process is to use different penalty functions for input and output layers. We propose using the L_1-norm function in the hidden layer and the L_2-norm function in the output layer. Furthermore, we evaluate separately the sum of the absolute values of the hidden weights, which are connected to the same feature (input), $|z_I|$. Consequently, features with larger L_1-norm values are compared to features with smaller L_1-norm values, using a simple statistical test. Features with smaller L_1-norm values are classified as irrelevant features and are excluded from the MLP. Features with larger L_1-norm values are classified as relevant features. Thus, this procedure allows training the MLP and selecting relevant features automatically.

The experiments were designed to show that the algorithm is able to generalize well while selecting a set of relevant variables. Two synthetic data sets and nine real problem databases found in the literature were considered in the tests, with different quantities of features and samples. The performance of the algorithm was compared to the performance of SVM which is one of the state-of-the-art learning machines.

Experimental results show that the proposed method achieves high performance, which are statistically equivalent to SVM, however, with a reduced set of features.

Acknowledgements This work was funded by CAPES and CNPq, and we would like to thanks FAPEMIG.

Compliance with ethical standards

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

References

(1994) Encyclopaedia of Mathematics (set). Springer Netherlands
Bartlett PL (1997) For valid generalization the size of the weights is more important than the size of the network. In: Mozer MC, Jordan MI, Petsche T (eds) Advances in neural information processing systems, vol 9. The MIT Press, Cambridge
Bland RG, Goldfarb D, Todd MJ (1980) The ellipsoid method: a survey. Technical report, Ithaca
Braga A, Takahashi R, Costa M, Teixeira R (2006) Multi-objective algorithms for neural networks learning. In: Jin Y (ed) Multi-objective machine learning, studies in computational intelligence, vol 16. Springer, Berlin, pp 151–171
Broadbent D (1958) Perception and communication. Pergamon Press, London
Dua D, Graff C (2017) UCI machine learning repository. http://archive.ics.uci.edu/ml
Gacek A, Pedrycz W (2011) ECG signal processing, classification and interpretation: a comprehensive framework of computational intelligence. Springer, New York
Geman S, Bienenstock E, Doursat R (1992) Neural networks and the bias-variance dilemma. Neural Comput 4:1–58
Gretton A, Bousquet O, Smola A, Schölkopf B (2005) Measuring statistical dependence with Hilbert–Schmidt norms. In: International conference on algorithmic learning theory. Springer, pp 63–77
Guyon I, Road C (2008) Practical feature selection : from correlation to causality. IOS Press, Amsterdam, pp 1–17
Hastie T, Tibshirani R, Friedman J (2009) The elements of statistical learning: data mining, inference and prediction, 2nd edn. Springer, New York
Haykin S (2001) Redes Neurais. Princípios e Prática, Bookman
Kira K, Rendell LA (1992) The feature selection problem: traditional methods and a new algorithm. AAAI. MIT Press, Cambridge, pp 129–134
Rampone S, Russo C (2012) A fuzzified brain algorithm for learning DNF from incomplete data. Electron J Appl Stat Anal 5(2). http://siba-ese.unisalento.it/index.php/ejasa/article/view/11409
Teixeira BTRS Roselito (2000) Improving generalization of MLPS with multi-objective optimization. Neurocomputing 35(1–4):189–194
Tibshirani R (1996a) Regression shrinkage and selection via the lasso. J R Stat Soc B 58:267–288
Tibshirani R (1996b) Regression shrinkage and selection via the lasso. J R Stat Soc (Ser B) 58:267–288
Vapnik V, Boser (1992) A training algorithm for optimal margin classifiers. In: Fifth annual workshop on computational learning theory, San mateo, pp 1–152
Vapnik VN, Cortes C (1995a) Support vector networks. Mach Learn 20:273–297
Vapnik V (1995b) The nature of statistical learning theory. Springer, New York
Yamada M, Jitkrittum W, Sigal L, Xing EP, Sugiyama M (2014) High-dimensional feature selection by feature-wise kernelized lasso. Neural Comput 26(1):185–207. https://doi.org/10.1162/NECO_a_00537 pMID: 24102126