Information Sciences 543 (2021) 157–179

Contents lists available at ScienceDirect

Information Sciences

journal homepage: www.elsevier.com/locate/ins

MINE: A framework for dynamic regressor selection

Thiago J.M. Moura [a,b,*], George D.C. Cavalcanti [b], Luiz S. Oliveira [c]

[a] Instituto Federal da Paraíba (IFPB), Av. Primeiro de Maio, 720, João Pessoa, Brazil
[b] Centro de Informática (CIn), Universidade Federal de Pernambuco (UFPE), Av. Jornalista Anibal Fernandes s/n, Recife, Brazil
[c] Universidade Federal do Paraná (UFPR), Rua Cel. Francisco Heraclito dos Santos, 100, Curitiba, Brazil

ARTICLE INFO

Article history:
Received 2 September 2019
Received in revised form 9 June 2020
Accepted 21 July 2020
Available online 26 July 2020

Keywords:
Regressors
Ensemble of regressors
Measures
Combination
Dynamic regressor selection

ABSTRACT

Dynamic Regressor Selection (DRS) techniques aim to select the most competent regressors from an ensemble per test pattern. So, for each test pattern, only a subset of the most competent regressors are used to estimate its target value. Hence, the central issue in DRS techniques is how to define the competence of the regressors which is usually defined using a single measure, such as the performance of the regressor in the local region of the feature space around the test pattern, called the region of competence. However, no single measure is the best for any task. In this work, we present the Meta INtEgration (MINE) framework that selects and combines the most competent regressors from an ensemble during the evaluation of a given test pattern. MINE uses different measures extracted from the region of competence as a criterion for the selection and combination of the regressors. In contrast to traditional combination schemes where all the regressors are weighted to produce the final answer, MINE selects the best regressors per pattern on-the-fly and combines them to predict the value of the test pattern. Comprehensive experiments on 20 regression datasets show that MINE compares favorably to literature techniques.

1. Introduction

Ensemble learning refers to techniques that generate different models, with some degree of diversity, which are combined to make a prediction, either in classification or regression problems. The advantage of ensembles concerning single models has been reported in terms of increased robustness and accuracy for both classification [1–3], and regression problems [4–6].

Ensemble-based systems contain three main modules [7]: (1) Generation, (2) Selection, and (3) Combination. In the generation module, a training set is used to create the ensemble. The ensemble is said homogeneous when a single learning algorithm is used to train all the models; otherwise, it is called heterogeneous. In the second module, only one model or a subset of the ensemble is selected. Finally, when a subset of the ensemble is selected, the models are combined to estimate the target value of a given test pattern. Over the last two decades, researchers have been dedicating efforts to improve the quality of the ensemble [5,8], and also searching for alternatives to better select and combine the models [9–11].

Regarding the selection module, it can be either static or dynamic. In the static approach, the selection is performed before the evaluation of the test pattern using the information extracted from the training [12] or validation set [13]. So,

* Corresponding author at: Unidade Acadêmica 2 (UA2), Instituto Federal da Paraíba (IFPB), Av. Primeiro de Maio, 720, João Pessoa, Brazil.
E-mail addresses: thiago.moura@ifpb.edu.br (T.J.M. Moura), gdcc@cin.ufpe.br (G.D.C. Cavalcanti), luiz.oliveira@ufpr.br (L.S. Oliveira).

https://doi.org/10.1016/j.ins.2020.07.056
0020-0255/

the selected models are used to estimate the target value of all test patterns. In the dynamic approach, a different subset of the ensemble is selected for each new test pattern. In dynamic selection techniques, each model is expected to be an expert in a specific local region of the feature space that is known as region of competence. So, for each test pattern, the most competent models are selected in the region of competence where the test pattern is located. Recent works have shown that dynamic selection techniques outperform static selection [14,15,7].

When the selected subset of the ensemble contains more than one model, they should be combined. The combination can be performed using a simple rule such as the mean or the weighted mean. In general, the weighted mean presents better precision than the mean [11], and its weight can be defined statically or dynamically. The former uses the same weight vector for any test pattern, while in the latter, the weights are defined according to the performance of the models in the region of the feature space where the test pattern is located [16]. On using static, i.e., equal weights for the whole test set, it is not adequate for some test patterns. This disadvantage of static weighting is the fundamental argument for using dynamic (non-constant) weights [17,18].

The crucial issue in dynamic selection systems is to define the criterion to measure the competence of the models. It is expected that the better the competence of the dynamically selected models, the higher the precision of the whole system. An usual manner to measure the competence consists in calculating the accumulated error of a given model in the neighborhood of the test pattern [19,20]. However, the literature on dynamic classifier selection (DCS) shows that using only the accumulated error in the region of competence is not enough to correctly calculate the competence of the classifiers [21–23].

Dynamic regression selection (DRS) algorithms consist of, given a query pattern and an ensemble of regressors, selecting the best regressors to predict the query pattern. This selection is commonly performed by calculating the error of each regressor in the region of competence and selecting the regressors that attain the smallest errors. DRS approaches found in the literature commonly use only one error measure to estimate the competence of the regressors in the region of competence, as in [19,20]. Moura et al. [24] evaluated eight different measures of competence and concluded that none of them has superior performance for different tasks. In other words, we argue that selecting one different measure per task or combining all the measures may increase the precision of DRS systems. DRS can benefit from the combination of several measures instead of relying on a single one, in the task to select/combine the regressors.

Thus, we devised an approach that combines competence measure aiming at improving the DRS system precision. To validate such an idea, we introduce the Meta INtEgration (MINE) framework for DRS. In MINE, for each test pattern, different measures are extracted from the region of competence. These measures are dynamically weighted and used as a criterion to select the best regressors for a specific pattern. Instead of having only one set of weights for all the patterns, MINE defines a different set of weights on-the-fly per pattern. Such a strategy is particularly interesting because it gives different importance to each measure; moreover, it is known that there is not a single best measure for all the cases.

The contribution of this work is two-fold. Firstly, we proposed a DRS framework that can operate in three different scenarios: (i) the selection of a single regressor given a test pattern (MINE-Selection (MINE-S)); (ii) all the regressors in the ensemble are weighted and combined (MINE-Weighting (MINE-W)); and, (iii) a subset of the ensemble is dynamically selected per test pattern (MINE-Weighting with Selection (MINE-WS)). Secondly, we present a robust study that constructs homogeneous ensembles where the base learning algorithm is selected per regression problem.

To evaluate the performance of the MINE framework and show the relevance of the measures adopted with homogeneous ensembles, we carried out a set of extensive experiments on 20 regression problems. We compare the MINE framework against state-of-the-art DRS techniques, and individual regressor trained with the whole training set. Our experimental results show that the adopted measures are useful for the DRS with homogeneous ensembles and validate our proposal that better results are achieved when using multiple measures.

This paper is organized as follows: Section 2 presents the related works. Section 3 describes the proposed framework for DRS. Section 4 shows the methodology and experiments used to evaluate the proposed framework. Section 5 presents the conclusions about the research.

2. Related works

This section reviews the literature about selection and combination of regressors. Table 1 presents the related works focusing on three aspects: (i) selection strategy that indicates whether the technique is static or dynamic; (ii) ensemble type that indicates whether the ensemble is homogeneous or heterogeneous; and (iii) selection criterion that indicates what is the measure used as the criterion to define the competence of the regressors from the ensemble. The value "error" in the column "Selection Criterion" indicates that an error measure is used as a criterion to select the regressors.

Perrone et al. [11] defined two ways to combine the models from an ensemble: Basic Ensemble Method (BEM), and Generalized Ensemble Method (GEM). In the BEM, the combination of the models is performed using the mean among the regressors, where all the regressors have the same importance. In the GEM, the models are combined using the weighted mean where the weights are inversely proportional to the errors generated in the training set or the validation set. These weights are constants; it means that the weights do not change during the evaluation of query patterns. Experiments were carried out with homogeneous ensembles of Multilayer Perceptrons, and the GEM performed similar or better than an individual regressor or BEM.

Table 1
Related works.

Method	Static/Dynamic	Ensemble Type	Selection Criterion
Perrone et al. [11]	Static	homogeneous	error
Partalas et al. [13]	Static	homogeneous	error
Rooney et al. [19]	Dynamic	homogeneous	error
Moreira et al. [20]	Dynamic	heterogeneous	error
Rooney et al. [25]	Dynamic	homogeneous	error
Sergio et al. [26]	Dynamic	heterogeneous	error
Moura et al. [24]	Dynamic	homogeneous	error

Partalas et al. [13] presented an algorithm to select the best subset of regressors from an ensemble. Their algorithm uses greedy search (forward selection and backward elimination) to select the best subset of the regressors based on the performance in a validation set. The selection is static, and once the subset of the ensemble is defined, it will be the same for all test patterns. Ensembles are generated using Neural Networks and Support Vector Machine. The Forward Selection algorithms achieved the best performance.

Rooney et al. [19] proposed three DRS algorithms that use as selection criterion the accumulated error in the region of competence. Two different learning algorithms were used: linear regression and 5-NN (5 nearest neighbors). For each learning algorithm, they generated homogeneous ensembles using Random Subspace [1]. It follows that dynamic selection techniques have superior performance when compared to a single individual regressor. Among the DRS techniques, the results show the DW algorithm with better performance than the DWS. Later, Moreira et al. [20] used these three DRS algorithms with the difference that the errors are weighted by the distance between the test pattern and its neighbors. This work is the latest on dynamic selection and combination of regressors, and it uses the DRS algorithms with homogeneous ensembles. Moreira et al. [20,27] describe the three algorithms as follows:

- Dynamic Selection (DS) – it selects the regressor with the lowest accumulated error in the region of competence. The errors are weighted by the distance between the neighborhood patterns and the test pattern. Only a single regressor is selected and the combination is not required. The estimation of the test pattern is the value returned by the selected regressor.
- Dynamic Weighting (DW) – it combines all the regressors of the ensemble using the weighted mean. For each test pattern x_j, its region of competence Ψ is calculated; Ψ is composed of K patterns. For each pattern in Ψ, a weight is calculated using Eq. (1):

$$d_k = \frac{\frac{1}{dist_k}}{\sum_{j=1}^{K}\left(\frac{1}{dist_j}\right)} \tag{1}$$

  where $dist_k$ is the distance between a pattern $t_k \in \Psi$ and the test pattern x_j.
  The vector $\{d_1, d_2, \ldots, d_K\}$ is used to calculate the weight α_i of the regressor $\hat{f}_i$, using Eq. (2):

$$\alpha_i = \frac{\frac{1}{\sqrt{\sum_{k=1}^{K}(d_k \times sqe_{k,i})}}}{\sum_{n=1}^{N}\left(\frac{1}{\sqrt{\sum_{k=1}^{K}(d_k \times sqe_{k,n})}}\right)} \tag{2}$$

  where N is the ensemble size, k represents the index of the neighbor, and $sqe_{k,i}$ is the squared error of the regressor i calculated using the pattern $t_k \in \Psi$.
- Dynamic Weighting with Selection (DWS) – it combines a subset of the regressors. The regressors with the accumulated error in the upper half of the error interval $E_i > (E_{max} - E_{min})/2$ are discarded, where E_{max} is the largest accumulated error of any regressor and E_{min} is the lowest accumulated error of any regressor. The measure to calculate the performance of the regressors from the ensemble is the same than the DW algorithm and the remaining regressors are combined using the same strategy of the DW.

Moreira et al. [20] presented a different experimental protocol when compared to [19]. This work uses heterogeneous ensembles with size four and tests variations in the region of competence's size. The conclusion is that DW and DWS performed better than the DS algorithm. Also, this work shows that the size of the region of competence is problem-dependent.

The wMetaComb [25] is a technique for regression problems that fuses two combination techniques: Stacking [28] and the DWS algorithm. In the wMetaComb, the estimated value of the test pattern is the weighted mean of the predictions of two combination techniques. The weights to combine the techniques are calculated based on the errors during the training

phase. wMetaComb mixes two combination techniques. The DWS could be exchanged for DS ou DW. The results showed that the technique has slightly superior performance than the two combination techniques when performed individually.

In Sergio et al. [26], the proposed technique, for each test pattern, selects the best combination of models, instead of individual ones. When a test pattern is evaluated, the region of competence is found, and the combiner (Mean, Median, or Softmax [29]) with the lowest error rate in the region is chosen to evaluate the test pattern. The experiments were carried out using chaotic time series forecasting and show that the dynamic selection of the combiners performs better than each of them individually. Also, the proposed technique performs better than each model of the ensemble.

Finally, Moura et al. [24] compared eight competence measures using the DS, DW, and DWS algorithms presented by Moreira et al. [20,27]. Among the measures, seven of them were adapted for the first time to this task. Their homogeneous ensemble was composed of a hundred models trained with the CART [30]. This study showed that none of the eight measures obtains a superior performance for all datasets used in the experiments. They concluded that the selection of the best measure should be performed in a problem-dependent fashion. They also highlight that instead of choosing the best measure, an alternative is to combine the measures to obtain better performance rates.

It is important to notice that the technical literature uses only the error either as a selection criterion or as the measure to calculate the weights for the combination of the regressors. In the static selection, the error is calculated using the training set or the validation set. In the dynamic selection, the error calculated in the region of competence is used as a selection criterion. The proposed framework presents an approach that uses not only the error but the composition of other measures as a criterion for the DRS. In addition, the new framework is not limited to the use of a specific learning algorithm for the generation of homogeneous ensembles, but it chooses a suitable one for each regression problem.

3. MINE framework

The Meta INtEgration (MINE) framework architecture (Fig. 1) is divided into four phases: Learning Algorithm Selection, Generation, Optimization, and Generalization. In the first phase, the best learning algorithm is selected for the task under

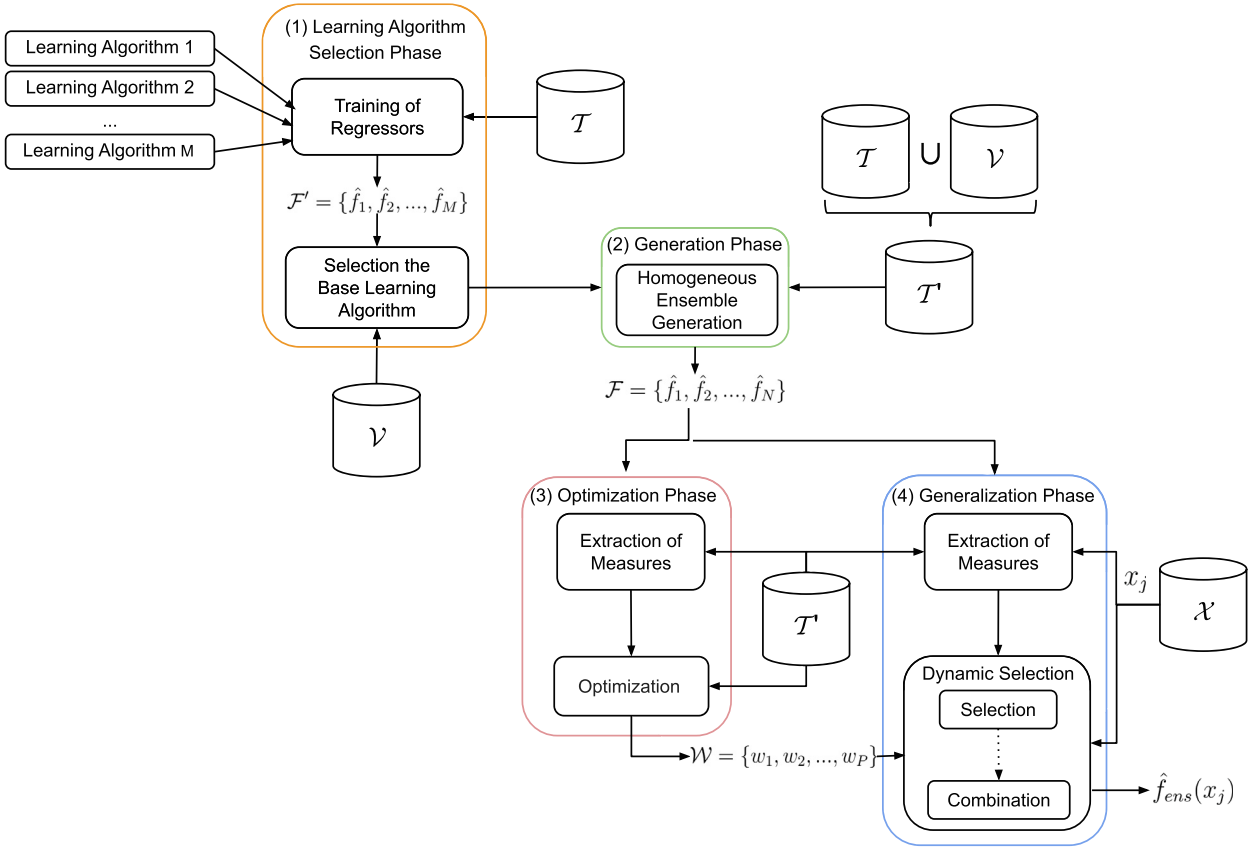

Fig. 1. Architecture of MINE framework. In Learning Algorithm Selection Phase, $\mathcal{T}$ and $\mathcal{V}$ are the sets of Training and Validation, respectively. $\mathcal{F}' = \{\hat{f}_1, \hat{f}_2, ..., \hat{f}_M\}$ is a heterogeneous set of regressors generated in the first phase. In Generation Phase, $\mathcal{T}'$ is the training set used to train the homogeneous ensemble $\mathcal{F} = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_N\}$. $\mathcal{W} = \{w_1, w_2, \ldots, w_P\}$ is the vector of weights resulting from the Optimization Phase. In the Generalization Phase, $\mathcal{X}$ is the test set, x_j is a pattern from test set, and $\hat{f}_{ens}(x_j)$ is the ensemble estimative for the pattern x_j.

analysis and a homogeneous ensemble using this learning algorithm is generated in the second phase. After, some competence measures are extracted and the Optimization phase calculates a weight for each measure; the more important the measure, the greater your weight. The last phase selects a subset of the ensemble to predict the value of the query pattern. These four phases are detailed in the following sections.

3.1. Learning algorithm selection

This phase aims at selecting the learning algorithm (among M) given the training set $\mathcal{T}$, and the validation set $\mathcal{V}$. So, M regressors $\mathcal{F}' = \{\hat{f}_1, \hat{f}_2, \ldots, \hat{f}_M\}$ are trained, each one using a distinct learning algorithm (Training of Regressors module). After, the performance of the M regressors is evaluated using the validation set $\mathcal{V}$, and the base learning algorithm that minimizes the MSE on $\mathcal{V}$ is selected (Selection the Base Learning Algorithm module). This learning algorithm is used to generate the homogeneous ensemble for the task under analysis in the next phase.

3.2. Generation

This phase generates a homogeneous ensemble $\mathcal{F}$ containing N regressors. The learning algorithm selected in previous phase is employed to train all the regressors using distinct sets generated with the Bagging (Bootstrap AGGegatING) algorithm [31].

3.3. Optimization

Dynamic regressor selection systems use the error of the predictions in the region of competence as a criterion to dynamically select the best regressors. Moura et al. [24] evaluated eight different measures and showed that none of them is the ideal choice when used isolated and also that the best measure depends on the task. As stated before, we advocate that the combination of different measures is a better alternative than using only one. So, this phase aims at generating a vector of weights $\mathcal{W} = \{w_1, w_2, \ldots, w_P\}$ that gives different importance for each measure $m_i, i = \{1, 2, \ldots, P\}$, and it is composed of two modules: Extraction of Measures, and Optimization. In the next section, eight measures are defined, so, $P = 8$.

3.3.1. Extraction of measures

A total of eight measures $\{m_1, m_2, \ldots, m_8\}$ are extracted from the region of competence and they correspond to different criteria to analyze the behavior of each regressor. Measure m_1 captures the diversity among the regressors $\hat{f}_n \in \mathcal{F}$ using the variance of their estimations. On the other hand, m_2, m_3, and m_7 capture different points of view of the prediction error. The dispersion and centrality of the error in the region of competence are calculated using m_4 and m_5 respectively. The similarity between the estimation of a pattern and the observed values of its nearest neighbors is calculated using measure m_6. And finally, m_8 measures the error of the nearest neighbor.

In the next equations, $f(t_k)$ refers to the observed value of the neighborhood pattern t_k, $\hat{f}_n(t_k)$ is the estimated value of the pattern t_k given by the regressor $\hat{f}_n$, and d_k is the inverse of the normalized distance in the interval $[0; 1]$. So, the smaller the distance the greater the value of d_k, according to Eq. (1).

In the Eq. (1), $\{dist_1, dist_2, \ldots, dist_K\}$ is a vector where each element is a distance measure between the neighbor pattern from the training set $\mathcal{T}'$ and the training pattern t_i, and K is the neighborhood size. The measures are extracted from the region of competence $\Psi = \{t_1, t_2, \ldots, t_K\}$ for each pattern t_i, where t_k is a pattern from the same training set $\mathcal{T}', \forall k \in \{1, 2, \ldots, K\}$.

The eight measures calculated for each regressor $\hat{f}_n$ are described below.

- m_1 – *Variance*: the variance of the neighbors estimated values. The variance is calculated for each regressor using the estimated values for the patterns in the region of competence, according to Eq. (3):

$$m_{1,n} = Var(\hat{f}_n(t_1), \hat{f}_n(t_2), \ldots, \hat{f}_n(t_K)) \tag{3}$$

  This measure is inspired in the work of Tresp et al. [32], whose variance of the estimated values is used as weight in the static combination of artificial neural networks.
- m_2 – *Sum Absolute Error*: the sum of the absolute errors is calculated in the region of competence, weighted by d_k, according to Eq. (4):

$$m_{2,n} = \sum_{k=1}^{K} \left| f(t_k) - \hat{f}_n(t_k) \right| \times d_k \tag{4}$$

- m_3 – *Sum Squared Error*: the sum of the squared errors is calculated using the inverse of the distances d_k as weights, according to Eq. (5):

$$m_{3,n} = \sum_{k=1}^{K} (f(t_k) - \hat{f}_n(t_k))^2 \times d_k \tag{5}$$

- m_4 – *Minimum Squared Error*: this measure is the lowest value of the errors, weighted by the distance d_k. The measure m_4 is computed using Eq. (6):

$$m_{4,n} = \min_{1 \leqslant k \leqslant K} \{(f(t_k) - \hat{f}_n(t_k))^2 \times d_k\} \tag{6}$$

- m_5 – *Maximum Squared Error*: this measure is the maximum value of the errors, weighted by the distance d_k. The measure m_5 is computed using Eq. (7):

$$m_{5,n} = \max_{1 \leqslant k \leqslant K} \{(f(t_k) - \hat{f}_n(t_k))^2 \times d_k\} \tag{7}$$

Considering that m_4 and m_5 define an interval, these measures present mean and variance, it means that, the interval contains information about implicit measures of dispersion (error variance) and centrality (error mean) of the squared error in the region of competence.

- m_6 – *Neighbor's Similarity*: the sum of the differences between the estimated value of the validation pattern from validation set $\mathcal{T}'$ and the observed values of each neighborhood pattern, weighted by the inverse of the distance. The measure m_6 is computed using Eq. (8):

$$m_{6,n} = \sum_{k=1}^{K} (f(t_k) - \hat{f}_n(t_i))^2 \times d_k \tag{8}$$

where $\hat{f}_n(t_i)$ is the estimated value of the regressor $\hat{f}_n$ for t_i. t_i is the pattern being tested in the leave-one-out process. The goal of the measure m_6 is to find the degree of similarity between the estimation of the pattern $t_i \in \mathcal{T}'$ and the observed values of the nearest neighbors $\{t_1, t_2, \ldots, t_K\}$. This is the only measure that uses the estimated value for the test pattern ($\hat{f}_n(t_i)$). So far as we know, this measure is unprecedented and is defined by the authors of this work.

- m_7 – *Root Sum Squared Error*: the root of sum squared errors in the region of competence, with the errors weighted by d_k. The measure m_7 is computed using Eq. (9):

$$m_{7,n} = \sqrt{\sum_{k=1}^{K} (f(t_k) - \hat{f}_n(t_k))^2 \times d_k} \tag{9}$$

Root squared error is more stable and less sensitive to the difference between the maximum and the minimum errors, while squared error is very sensitive to extreme error values. The measures m_3 and m_7 present different points of view from the error calculated in the region of competence. These two measures have a high correlation, but using them together allows a better balance in the weights of the combination [33]. Also, m_3 and m_7 produce the same result when a single regressor is chosen to estimate a test pattern, but different results in the combination of the regressors [24].

- m_8 – *Closest Squared Error*: the error obtained by the regressor only on the nearest neighbor. The measure m_8 is computed using Eq. (10):

$$m_{8,n} = (f(t_1) - \hat{f}_n(t_1))^2 \tag{10}$$

For each pair (pattern $t_i \in \mathcal{T}'$, regressor $\hat{f}_n$), the eight measures are extracted from the region of competence of the pattern t_i and produces a vector $M_{i,n} = \{m_{1,n}, m_{2,n}, \ldots, m_{8,n}\}$ where each element is the value of one measure.

3.3.2. Optimization

This module uses a Genetic Algorithm (GA) [34] to obtain one weight per measure using the vectors $M_{i,n}$ described in the last section. Algorithm 1 shows the pseudo-code of the optimization process whose output is the vector $\mathcal{W} = \{w_1, w_2, \ldots, w_P\}$ that minimizes the Mean Squared Error (*MSE*) of the training set $\mathcal{T}'$.

Algorithm 1 Optimization Process

Input: Ensemble $\mathcal{F}$; Training set $\mathcal{T}'$; Neighborhood size K; Population size L
Output: $\mathcal{W}_{best}$: Best Individual
1: $Pop = InitialPopulation(L)$;
2: **repeat**
3: $MSE_{Pop} = \varnothing$; {set with the MSE of all individuals}
4: **for** each individual $\{w_1, w_2, \ldots, w_P\}$ in Pop **do**
5: $SE = 0$
6: **for** each pattern t_i in $\mathcal{T}'$ **do**
7: $\mathcal{T}' = \mathcal{T}' - t_i$ Leave-one-out
8: Find the region of competence Ψ of t_i using $\mathcal{T}'$
9: **for** each $\hat{f}_n$ in $\mathcal{F}$ **do**
10: Calculate the measures $\{m_{1,n}, m_{2,n}, \ldots, m_{P,n}\}$ using Ψ
11: $\alpha_n = \sum_{p=1}^{P} w_p \times m_{p,n}$
12: **end for**
13: $\hat{f}_{ens}(t_i) = DynamicSelection(\mathcal{F}, \mathcal{A}, t_i)$
14: $SE = SE + (f(t_i) - \hat{f}_{ens}(t_i))^2$
15: **end for**
16: $MSE = SE / |\mathcal{T}'|$
17: $MSE_{Pop} = MSE_{Pop} \cup MSE$
18: **end for**
19: $BestInds = SaveBestIndsElitism(MSE_{Pop}, Pop)$
20: $Pop = GenerateOffspring() \cup BestInds$
21: **until** $MSE = 0$ or reach maximum iteration
22: $\mathcal{W}_{best} = BestInd(MSE_{Pop}, Pop)$
23: **return** $\mathcal{W}_{best}$

In line 1, the initial population is generated. The population is composed of L individuals, and each is a vector of weights $\mathcal{W}$ whose size is given by the number of measures. In this way, each individual is represented by a set of P genes and each gene is a real value $w_p \in \mathrm{IR}, \forall p \in \{1, 2, \ldots, P\}$.

From line 4 to 18, the fitness function of the genetic algorithm is calculated. For each pattern $t_i \in \mathcal{T}'$, the region of competence Ψ is defined (line 8) and the measures are extracted for each regressor $\hat{f}_n \in \mathcal{F}$ (lines 9 and 10). Line 11 shows the weighted combination of the measures to compute a new vector $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$, where N is the number of regressors from the ensemble.

Dynamic Selection uses the vector $\mathcal{A}$ to estimate the value $\hat{f}_{ens}(t_i)$ for the pattern t_i (line 13) and the squared error is computed in line 14. The estimated value $\hat{f}_{ens}(t_i)$ can be the result of one of the following DRS techniques: (i) MINE-S – dynamic selection of a single regressor from the ensemble; (ii) MINE-W – combination of all the regressors from the ensemble; or (iii) MINE-WS – dynamic selection and combination of a subset of regressors from the ensemble. These DRS techniques are explained in Section 3.4. The framework works for each DRS technique separately. In other words, the optimization process is technique-dependent.

The *MSE* is computed in line 16, and this is the fitness function (Eq. 11) of the optimization procedure.

$$fit(ind) = \frac{1}{|\mathcal{T}'|} \sum_{i=1}^{|\mathcal{T}'|} (f(t_i) - \hat{f}_{ens}(t_i))^2 \tag{11}$$

where ind is an individual that belongs to Pop, t_i is a pattern from the training set $\mathcal{T}'$, $f(t_i)$ is the observed value of the pattern t_i, and $|\mathcal{T}'|$ is the number of instances in the training set $\mathcal{T}'$.

The *MSE* of all individuals are stored into the set MSE_{Pop}, and after to finish all the individuals, the best ones (lower *MSE*), are selected (line 19) to compose the new offspring (line 20). At the end of the algorithm, the best individual (lowest *MSE*) is selected and stored into $\mathcal{W}_{best}$ (line 22).

In the proposed framework, any optimization algorithm can be employed. For the sake of simplicity, we adopt genetic algorithm. Its parameters, mutation, crossover, and elitism, are discussed in Section 4.2. Besides, the weights per measure generated by this optimisation procedure are evaluated in Section 4.8.

3.4. Generalization

In this phase, the estimated value $\hat{f}_{ens}(x_j)$ is calculated for each test pattern x_j. This phase consists of two modules: Extraction of Measures and Dynamic Selection. The Extraction of Measures module receives as input the ensemble $\mathcal{F}$, the test set $\mathcal{X}$ and the training set $\mathcal{T}'$. This module works similarly as described in Section 3.3.1, where for each test pattern $x_j \in \mathcal{X}$, the region of competence is defined using $\mathcal{T}'$ and the measures are extracted for each regressor $\hat{f}_n \in \mathcal{F}$. The Dynamic Selection module receives as input the measures extracted in the previous module, the ensemble $\mathcal{F}$, the test set $\mathcal{X}$, and the weights $\mathcal{W}$ calculated in the Optimization Phase. This module is responsible for calculating the competence of the regressors using as criterion the combination of the measures. After the combination of the measures, $\hat{f}_{ens}(x_j)$ is computed as the final estimation for test pattern x_j.

The Dynamic Selection module contains two submodules: Selection and Combination. The first one is responsible to select one or more regressors from the ensemble per test pattern. If more than one regressor is selected, the Combination submodule is performed. The Combination submodule can also combine all the regressors directly, without executing the Selection submodule.

In this work, three ways of using the MINE framework are proposed: (i) MINE-S – dynamic selection of a single regressor from the ensemble; (ii) MINE-W – combination of all the regressors from the ensemble; and (iii) MINE-WS – dynamic selection and combination of a subset of regressors from the ensemble.

3.4.1. Dynamic selection

This module aims at selecting the best regressor(s) per test pattern x_j from the ensemble of regressors $\mathcal{F}$ given the vector of weights $\mathcal{W}$ calculated in the Optimization Phase. If more than one regressor are selected, they should be combined to produce the estimated value of the test pattern.

The selection process is based on the competence of each regressor $\hat{f}_n$ in the estimation of the value of x_j. The competence of $\hat{f}_n$ is calculated using α_n (Eq. 12) that multiplies each measure $m_{p,n}$ by its respective weight $w_p \in \mathcal{W}$. It is important to remember that each vector of measures $M_{j,n} = \{m_{1,n}, m_{2,n}, \ldots, m_{P,n}\}$ is calculated using the regressor $\hat{f}_n$ and the region of competence of the test pattern x_j; so, this vector is regressor-dependent.

$$\alpha_n = \sum_{p=1}^{P} w_p \times m_{p,n} \tag{12}$$

where $m_{p,n}$ is the measure p calculated for the regressor $\hat{f}_n$ in the region of competence, w_p is the weight of the measure p in the vector $\mathcal{W}$, and α_n is the result of the measures combination for each regressor $\hat{f}_n$ from the ensemble $\mathcal{F}$.

After the evaluation of the competence of each regressor, we have a vector $\mathcal{A}$ that is calculated dynamically using the region of competence, and is responsible to select and combine the regressors. Depending on how the regressors are selected using $\mathcal{A}$, we propose three techniques of DRS using the MINE framework: MINE-S, MINE-W, and MINE-WS that are described below. These variations are similar to the ones in [20], but they use a different measure to calculate the weights to combine the regressors.

Using the MINE framework, one of the three proposed techniques can be used during the execution of the Optimization and Generalization phases. In addition to the proposed techniques, MINE framework can be modified to meet another strategy not foreseen in this work.

3.4.1.1. MINE-S. Some measures ($m_{p,n}$) capture different points of view of the error calculated in the region of competence per regressor $\hat{f}_n$. So, it is correct to say that the lower the weighted sum of these measures given by α_n, the more competent is the regressor $\hat{f}_n$. Thus, MINE-Selection selects the regressor that obtains the lowest value of $\alpha_n \in \mathcal{A}$, for each test pattern $x_j \in \mathcal{X}$. The regressor index is selected using Eq. (13):

$$index = \underset{1 \leqslant n \leqslant N}{\operatorname{argmin}}(\{\alpha_1, \alpha_2, \ldots, \alpha_N\}) \tag{13}$$

and the estimated value for the test pattern is calculated using Eq. (14):

$$\hat{f}_{ens}(x_j) = \hat{f}_{index}(x_j) \tag{14}$$

where $\hat{f}_{index}(x_j)$ is the estimated value for the test pattern x_j. Algorithm 2 presents the pseudo-code of the MINE-S.

3.4.1.2. MINE-W. MINE-Weighting combines all the regressors from the ensemble $\mathcal{F}$ using the vector $\mathcal{A} = \{\alpha_1, \alpha_2, \ldots, \alpha_N\}$. For each test pattern, the estimated value is the weighted mean of the regressors estimates. The values $\alpha_n \in \mathcal{A}$ are normalized using Eq. (15):

$$\tilde{\alpha}_n = \frac{\frac{1}{\alpha_n}}{\sum_{n=1}^{N} \frac{1}{\alpha_n}}. \tag{15}$$

So, the estimated value for the test pattern x_j is computed using Eq. (16):

$$\hat{f}_{ens}(x_j) = \sum_{n=1}^{N} \tilde{\alpha}_n \times \hat{f}_n(x_j). \tag{16}$$

Algorithm 3 presents the pseudo-code of the MINE-W.

Algorithm 2 Selecting using MINE-S

Input: Ensemble $\mathcal{F}$; Training set $\mathcal{T}'$; Test set $\mathcal{X}$; Vector of Weights $\mathcal{W}$; Neighborhood size K
Output: *MSE*: Mean Squared Error

$SE = 0$
for each test pattern x_j in $\mathcal{X}$ **do**
  Find the region of competence Ψ of x_j using $\mathcal{T}'$
  for each $\hat{f}_n$ in $\mathcal{F}$ **do**
    Calculate the measures $\{m_{1,n}, m_{2,n}, \ldots, m_{P,n}\}$ using Ψ
    $\alpha_n = \sum_{p=1}^{P} w_p \times m_{p,n}$
  end for
  $index = \text{argmin}_{1 \leqslant n \leqslant N}(\{\alpha_1, \alpha_2, \ldots, \alpha_N\})$
  $\hat{f}_{ens}(x_j) = \hat{f}_{index}(x_j)$
  $SE = SE + (f(x_j) - \hat{f}_{ens}(x_j))^2$
end for
$MSE = SE/ \mid \mathcal{X} \mid$
return *MSE*

Algorithm 3 Combining all the regressors using MINE-W

Input: Ensemble $\mathcal{F}$; Training set $\mathcal{T}'$; Test set $\mathcal{X}$; Vector of Weights $\mathcal{W}$; Neighborhood size K
Output: *MSE*: Mean Squared Error

$SE = 0$
$\mathcal{A} = \varnothing$
for each test pattern x_j in $\mathcal{X}$ **do**
  Find the region of competence Ψ of x_j using $\mathcal{T}'$
  for each $\hat{f}_n$ in $\mathcal{F}$ **do**
    Calculate the measures $\{m_{1,n}, m_{2,n}, \ldots, m_{P,n}\}$ using Ψ
    $\alpha_n = \sum_{p=1}^{P} w_p \times m_{p,n}$
    $\mathcal{A} = \mathcal{A} \cup \alpha_n$
  end for
  for each α_n in $\mathcal{A}$ **do**
    $\tilde{\alpha}_n = (1/\alpha_n)/(\sum_{n=1}^{N}(1/\alpha_n))$
  end for
  $\hat{f}_{ens}(x_j) = \sum_{n=1}^{N} \tilde{\alpha}_n \times \hat{f}_n(x_j)$
  $SE = SE + (f(x_j) - \hat{f}_{ens}(x_j))^2$
end for
$MSE = SE/ \mid \mathcal{X} \mid$
return *MSE*

3.4.1.3. MINE-WS. In MINE-Weighting with Selection, for each test pattern, the regressors with $\alpha_n > (\alpha_{max} - \alpha_{min})/2$ are removed from the ensemble, i.e., the values of $\mathcal{A}$ in the upper half of the difference between the largest and lowest values are discarded. For the remaining regressors, they are combined using Eqs. (15) and (16). Algorithm 4 presents the pseudo-code of the MINE-WS.

Algorithm 4 Selecting and Combining the regressors using MINE-WS

Input: Ensemble $\mathcal{F}$; Training set $\mathcal{T}'$; Test set $\mathcal{X}$; Vector of Weights $\mathcal{W}$; Neighborhood size K
Output: *MSE*: Mean Squared Error

$SE = 0$
$\mathcal{A} = \varnothing$
for each test pattern x_j in $\mathcal{X}$ **do**
  Find the region of competence Ψ of x_j using $\mathcal{T}'$
  for each $\hat{f}_n$ in $\mathcal{F}$ **do**
    Calculate the measures $\{m_{1,n}, m_{2,n}, \ldots, m_{P,n}\}$ using Ψ
    $\alpha_n = \sum_{p=1}^{P} w_p \times m_{p,n}$
    $\mathcal{A} = \mathcal{A} \cup \alpha_n$
  end for
  $\tilde{\mathcal{F}} = \mathcal{F}$
  $\tilde{\mathcal{A}} = \mathcal{A}$
  for each $\hat{f}_n$ in $\mathcal{F}$ **do**
    if $\alpha_n > (\alpha_{max} - \alpha_{min})/2$ **then** {Selecting}
      $\tilde{\mathcal{F}} = \tilde{\mathcal{F}} - \hat{f}_n$
      $\tilde{\mathcal{A}} = \tilde{\mathcal{A}} - \alpha_n$
    end if
  end for
  $N = size(\tilde{\mathcal{F}})$
  for each α_n in $\tilde{\mathcal{A}}$ **do** {Combining}
    $\tilde{\alpha}_n = (1/\alpha_n)/(\sum_{n=1}^{N}(1/\alpha_n))$
  end for
  $\hat{f}_{ens}(x_j) = \sum_{n=1}^{N} \tilde{\alpha}_n \times \hat{f}_n(x_j)\ \hat{f}_n \in \tilde{\mathcal{F}}$
  $SE = SE + (f(x_j) - \hat{f}_{ens}(x_j))^2$
end for
$MSE = SE/\ |\ \mathcal{X}\ |$
return *MSE*

To measure the performance of each proposed technique, the mean squared error (*MSE*) is calculated according to Eq. (17):

$$MSE = \frac{\sum_{j=1}^{J} f(x_j) - \hat{f}_{ens}(x_j))}{J} \tag{17}$$

where J is the size of the test set $\mathcal{X}$, $f(x_j)$ is the observed value of the test pattern x_j and $\hat{f}_{ens}(x_j)$ is the result of the generalization phase.

4. Experiments

The experiments were performed using a total of 20 regression datasets. Table 2 shows the main features of the datasets including the sources that are: the personal page of Prof. Luís Torgo,[1] UCI Repository,[2] and Delve Repository.[3] To facilitate the implementation of the framework, we used datasets with only numeric (integer or real) attributes, except for the Abalone dataset, in which the categorical attribute sex was converted to binary using two bits.

In Section 4.1 the entire experimental protocol is described. Section 4.2 presents the parameters of the genetic algorithm used in the optimization module (Section 3.3). In Section 4.3, the experiments present the results of the Learning Algorithm Selection Phase, where the regressors are tested using a validation set. In this phase, for each dataset, the best learning algorithm is chosen. Also, the experiments present the results of MINE-S compared against the DS algorithm (Section 4.4). In Section 4.5, the results of MINE-W and MINE-WS are compared against DW and DWS algorithms respectively. In Section 4.6, the results of MINE techniques are compared against Individual Regressor, Mean, and Median. Section 4.8 analyzes the importance of each measure extracted from the region of competence per dataset.

[1] http://www.dcc.fc.up.pt/~ltorgo/Regression/DataSets.html.
[2] http://http://archive.ics.uci.edu/ml/.
[3] http://www.cs.toronto.edu/~delve/.

Table 2
Datasets characteristics.

Dataset	Instances	Features	Source
Abalone	4177	8	UCI
Airfoil Self Noise	1503	5	UCI
Bank32NH	8192	32	Delve
Bank8FM	8192	8	Delve
Breast Cancer	194	32	Torgo
CCPP [35,36]	9568	4	UCI
Comp Act	8192	22	Delve
Comp Act Small	8192	8	Delve
Concrete [37]	1030	9	UCI
Delta Ailerons	7129	6	Torgo
Delta Elevators	9517	6	Torgo
Housing	506	13	UCI
Kinematics	8192	8	Delve
Machine	209	6	Torgo
Puma32H	8192	32	Delve
Puma8NH	8192	8	Delve
Stock	950	9	Torgo
Triazines [38,39]	186	60	Torgo
Wine Q. Red [40]	1599	12	UCI
Wine Q. White [40]	4898	12	UCI

Table 3
Mean and standard deviation of the results calculated in 20 replications, obtained for each regressor used to compare. For each dataset, the best result is in bold. Error values are in the scale 10^{-3}.

Dataset	CART	LINEAR	FANN-1	FANN-2	SVR (RBF)	SVR (Linear)	SVR (POLY 3)	RBF	3-NN	5-NN
Abalone	9.78 (0.13)	6.29 (0.03)	5.85 (0.15)	**5.72 (0.06)**	6.43 (0.01)	6.49 (0.02)	6.00 (0.05)	6.20 (0.05)	7.26 (0.09)	6.66 (0.06)
Airfoil Self Noise	7.01 (0.25)	16.46 (0.05)	5.92 (0.76)	**3.73 (0.55)**	11.20 (0.09)	16.94 (0.05)	10.53 (0.14)	12.34 (0.30)	7.02 (0.28)	9.89 (0.26)
Bank32NH	20.21 (0.30)	10.39 (0.01)	**10.08 (0.11)**	10.20 (0.13)	13.57 (0.04)	11.53 (0.01)	27.28 (0.32)	17.10 (0.13)	23.38 (0.14)	21.56 (0.12)
Bank8FM	2.74 (0.04)	2.34 (0.00)	1.33 (0.01)	**1.30 (0.01)**	1.76 (0.01)	2.38 (0.00)	1.66 (0.02)	3.48 (0.13)	15.90 (0.16)	15.95 (0.12)
Breast Cancer	120.20 (7.50)	80.82 (4.42)	102.36 (7.98)	95.32 (7.61)	76.11 (2.00)	75.41 (3.37)	312.73 (44.44)	**75.15 (2.27)**	87.26 (3.49)	76.54 (3.04)
CCPP	3.27 (0.06)	3.65 (0.00)	2.95 (0.02)	2.93 (0.03)	3.06 (0.00)	3.67 (0.00)	3.08 (0.00)	3.16 (0.01)	2.66 (0.03)	**2.58 (0.02)**
Comp Act	1.23 (0.01)	9.59 (0.10)	**0.64 (0.04)**	0.65 (0.04)	1.10 (0.01)	15.64 (0.33)	0.72 (0.04)	4.26 (0.28)	0.90 (0.05)	0.94 (0.05)
Comp Act Small	1.62 (0.03)	9.99 (0.07)	**0.96 (0.02)**	1.00 (0.04)	1.27 (0.01)	15.33 (0.35)	1.01 (0.05)	2.57 (0.10)	1.11 (0.02)	1.05 (0.02)
Concrete	7.75 (0.45)	17.08 (0.12)	**5.91 (0.29)**	6.43 (0.29)	9.62 (0.10)	18.23 (0.26)	7.27 (0.20)	10.05 (0.37)	13.68 (0.38)	13.81 (0.32)
Delta Ailerons	2.28 (0.04)	1.60 (0.00)	1.53 (0.10)	1.53 (0.02)	1.58 (0.00)	1.64 (0.00)	**1.51 (0.00)**	1.62 (0.01)	1.84 (0.02)	1.68 (0.01)
Delta Elevators	4.56 (0.05)	2.88 (0.00)	**2.80 (0.01)**	2.80 (0.02)	2.82 (0.02)	2.89 (0.00)	2.83 (0.00)	2.93 (0.02)	3.64 (0.02)	3.29 (0.02)
Housing	10.68 (1.10)	11.90 (0.32)	8.77 (1.04)	9.28 (1.27)	9.51 (0.23)	12.78 (0.20)	**6.51 (0.84)**	10.46 (0.61)	11.44 (0.93)	12.92 (0.71)
Kinematics	21.61 (0.32)	20.30 (0.01)	4.49 (0.10)	**4.04 (0.12)**	4.89 (0.02)	20.78 (0.02)	10.34 (0.06)	16.12 (0.22)	8.39 (0.08)	7.40 (0.06)
Machine	4.93 (1.60)	4.07 (0.66)	7.38 (3.00)	7.59 (3.43)	7.68 (0.36)	4.71 (0.21)	5.43 (1.90)	**3.08 (1.01)**	5.30 (0.91)	6.15 (0.65)
Puma32H	3.61 (0.04)	23.09 (0.03)	**1.32 (0.07)**	1.48 (0.42)	21.22 (0.06)	23.53 (0.03)	19.04 (0.27)	27.58 (0.07)	28.53 (0.20)	25.76 (0.15)
Puma8NH	30.21 (0.32)	33.78 (0.02)	**17.07 (0.06)**	17.13 (0.06)	18.17 (0.06)	35.12 (0.05)	18.10 (0.05)	31.84 (0.15)	28.33 (0.25)	25.52 (0.22)
Stock	1.73 (0.13)	7.04 (0.04)	1.15 (0.08)	1.24 (0.09)	1.35 (0.02)	7.32 (0.09)	1.24 (0.02)	3.56 (0.30)	**0.66 (0.03)**	0.78 (0.03)
Triazines	21.91 (2.04)	29.82 (2.98)	34.68 (4.49)	31.64 (3.41)	**21.58 (0.75)**	24.37 (1.47)	44.03 (5.68)	25.04 (0.63)	22.52 (1.27)	22.81 (1.24)
Wine Q. Red	24.00 (0.86)	17.04 (0.07)	17.49 (0.30)	17.56 (0.50)	**16.61 (0.10)**	17.30 (0.08)	17.30 (0.30)	16.83 (0.12)	19.75 (0.39)	18.91 (0.25)
Wine Q. White	19.03 (0.34)	15.82 (0.04)	**14.46 (0.21)**	14.65 (0.24)	14.68 (0.03)	15.91 (0.02)	14.73 (0.26)	15.58 (0.07)	14.83 (0.21)	14.47 (0.14)

4.1. Experimental protocol

For each dataset, all data attributes were normalized into the interval [0,1]. The experiments were conducted using 20 replications, and for each replication, the configurations used are described in the next subsections.

4.1.1. Ensemble generation

So, in this phase, a set of regressors with size $M = 10$ is generated. All the regressores are generated using the whole training set $\mathcal{T}$. Ten learning algorithms were used in this phase: CART [30], LINEAR, feedforward neural network with one hidden layer (FANN-1) with 10 neurons, a feedforward neural network with two hidden layers (FANN-2), with 5 and 10 neurons in each of the layers, Support Vector Regression (SVR) with RBF kernel, SVR with Linear kernel, SVR with polynomial

order 3 kernel, RBF network with 10 neurons, 3-nearest neighbor (3-NN) and 5-nearest neighbor (5-NN). The learning algorithms were used with default settings found in MATLAB[4] without any specific adjustment. We believe that a fine-tuning process of these parameters can improve the results of the whole framework.

In the second phase (Generation Phase), homogeneous ensembles with different sizes $N = \{5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$ are generated using Bagging, a sampling with replacement, as in Bootstrap AGGregatING [31]. Bagging generates distinct datasets, using sampling with replacement. The outputs of the Bagging are N training sets $\{\mathcal{T}'_1, \mathcal{T}'_2, \ldots, \mathcal{T}'_N\}$, and each is used to train one regressor $\hat{f}_i \in \mathcal{F}$. All sets $\mathcal{T}'_i$ have the same size as the original training set $\mathcal{T}'$.

4.1.2. Framework validation

For each replication in the Learning Algorithm Selection Phase, a 10-fold cross-validation is carried out using 70% of the folds for the training set ($\mathcal{T}$) and 20% for the validation set ($\mathcal{V}$). From the Generation Phase onwards, a 10-fold cross-validation is carried out, and each replication uses 90% of the folds for training ($\mathcal{T}'$) and 10% for testing set ($\mathcal{X}$). The result of each replication is the arithmetic mean of the *MSE* calculated for the 10 testing sets used in the cross-validation.

4.1.3. Region of competence

In [20], experiments were performed varying the size of the region of competence K in the interval $\{2, 4, 6, 8, 10, 12, 14, 16, 18, 20, 25, 30\}$. They concluded that the appropriate size for the neighborhood is problem-dependent, so they fixed the size of the region of competence with $K = 10$. Analyzing works of classification [21,22], time-series forecasting [26], and regression [19], it can be verified that the size of the region of competence is fixed for better validation and comparison of the results. The main objective is to compare and validate state-of-the-art techniques against the proposed techniques regardless the size of the region of competence. Thus, according to [20], we fixed the size of the region of competence to $K = 10$ for all the experiments using DRS techniques.

4.1.4. State-of-the-art techniques

The algorithms DS, DW, and DWS use only one error measure as a criterion to select the most competent regressor [20]. For these techniques, we used the same experimental protocol of the MINE *framework*: the same data sets, learning algorithms to generate the ensemble and the size of the region of competence was fixed to K = 10.

For each regressor used in comparison, a 10-fold cross-validation is carried out using 90% of the folds for the training set ($\mathcal{T}'$) and 10% for testing set ($\mathcal{X}$). For the state-of-the-art techniques, the result for each replication is the arithmetic mean of the *MSE* calculated for the 10 testing sets used in the cross-validation.

4.1.5. Hypothesis tests

Non-parametric hypothesis tests were performed for pairwise comparison between the results obtained using the proposed techniques against the results obtained using state-of-the-art DRS techniques, and against the results obtained using classical combination techniques. Wilcoxon signed rank tests were used to compare two paired samples from the same population, each pair being independent, randomly selected, as suggested in [41]. The null hypothesis H_0 indicates whether the two methods have the same performance and the alternative hypothesis H_1 verifies whether the proposed techniques performs better (lowest error). A significance level of 5% was adopted for left-tailed. Values marked with • indicate that the null hypothesis must be rejected and there is evidence that the alternative hypothesis is correct ($pValue \leqslant 0.05$). In other words, the proposed technique achieves superior performance compared to the other techniques.

4.2. Genetic algorithm configurations

This section presents the parameters of the genetic algorithm used in the optimization module and all of them were defined empirically. For all replications, the genetic algorithm was configured as follows:

- Population Size: 80.
- Fitness Limit: 0.
- Crossover fraction: 0.8.
- Mutation Function: Gaussian with 0 mean and standard deviation 1.0.
- Maximum Generations: 100 $\times$ 8 genes = 800.
- Stall Generations Limit: 40.
- Elitism: Best 8 individuals move on to the next generation.
- Initial Population: 71 individuals randomly generated with the values of the genes in the interval [0,1], and nine individuals initialized according to Matrix 18. The first line of the matrix shows the first chromosome initialized with 1 for all genes. The other chromosomes of the matrix have 1 in only one gene.

[4] https://www.mathworks.com/products/matlab.html.

$$firstPop = \begin{bmatrix} 1 & 1 & 1 & 1 & 1 & 1 & 1 & 1 \\ 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} \quad (18)$$

Regarding the genetic algorithm (GA), previous works [42,43] suggest that a generic GA (without adjusted parameters) tends to converge as well as a GA with adjusted parameters (specific). The main difference is in the processing time, where a generic GA has a processing time much longer than a specific one.

So, we adopted the default Matlab parameters for the genetic algorithms that are the standard values commonly used for these kinds of algorithms. We believe that a fine-tuning process of these parameters can improve the results of the whole framework. However, we aim at comparing different combination strategies and not to optimize each parameter.

The exception are elitism and populations size whose parameters were defined as the number of genes in each chromosome and as $10\times$ the number of genes: $80(71 + 9)$ (9 from Matrix 18 and more 71 randomly chromosomes), respectively.

4.3. Learning algorithm selection phase results

This section presentes the Learning Algorithm Selection Phase results. For each dataset, a set of regressors was generated with the size $M = 10$. The used learning algorithms are described in the previous section. Table 3 shows the performance of the regressors in the validation set $\mathcal{V}$. The results were calculated using 20 replications. The best results are in bold.

According to the literature, we observe in Table 3 that no learning algorithm is better than the others for all situations. The best learning algorithm is problem dependent. For each dataset in the next experiments, the best-performing learning algorithm is used to generate the homogeneous regressor ensemble.

The worst-performing regressors were those trained with the following learning algorithms: CART, LINEAR, and SVR with Linear kernel. These regressors did not perform as the best one in any dataset, so these algorithms are not selected for any dataset in the next phases.

4.4. MINE-S results

This section presents the results of the experiments performed using the MINE-S technique that selects the most suitable regressor per test pattern. Table 4 compares MINE-S with DS for different ensemble sizes $N = \{5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$. The results are the *MSE* arithmetic mean and the standard deviation of the 20 replications for each dataset.

Table 5 shows that MINE-S is better than DS, on average, in 15 out of 20 datasets. The biggest difference is obtained when the size of the ensemble is equal to 40. After performing a hypothesis tests (Wilcoxon signed rank test), MINE-S was significantly better than DS in 12 out of 20 datasets.

MINE-S has better results for any ensemble size when compared to the DS technique. For MINE-S and DS, increasing the size of the ensemble does not guarantee better results. In some datasets, the error increases when the size of the ensemble increases. A possible explanation to this fact is that selecting a suitable regressor among too many is a difficult task.

4.5. MINE-W and MINE-WS results

This section presents the results of the experiments performed using MINE-W and MINE-WS techniques, Tables 5 and 6, respectively. The results show the arithmetic mean and the standard deviation of the *MSE* computed for the 20 replications using different sizes of the ensemble $N = \{5, 10, 15, 20, 30, 40, 50, 60, 70, 80, 90, 100\}$.

According to Table 5, MINE-W performs better than DW for any ensemble size, and an increase in the ensemble size leaded to a decrease in the error rates MINE-W obtained superior performance on average in 13 out of 20 datasets, and reached smaller error rates when compared with MINE-S. The Wilcoxon signed rank test showed that MINE-W was significantly better than DW in 13 out of 20 datasets.

According to Table 6, MINE-WS performs better than DWS for any ensemble size. Similarly to MINE-W, in MINE-WS, increasing the ensemble size leaded to a decrease of the error rates. MINE-WS obtained superior performance on average in 11 out of 20 datasets. Also, MINE-WS reached smaller error rates when compared with MINE-S, but worse results when compared to MINE-W. The reduction in the variance achieved by weighted average of all regressors can explain why using all of them is better than the selection a subset of the regressors or just one of them. MINE-WS was significantly better than DWS in 10 out of 20 datasets based on the Wilcoxon signed rank test.

Table 4
Mean and standard deviation of the results calculated in 20 replications. For each dataset, the best result is in bold. Line "Win/Tie/Loss" shows the total of the results compared with MINE-S. The values marked with a • indicate that the null hypothesis must be rejected ($pValue \leqslant 0.05$), in other words, the result of MINE-S achieves superior performance. The values are in the scale 10^{-4}.

Dataset	5		10		15		20		30		40	
	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S
Abalone	58.83(1.26)•	**57.16(0.84)**	60.20(1.50)•	**58.06(1.03)**	60.68(1.75)•	**59.01(2.88)**	61.05(1.62)•	**58.42(0.91)**	62.11(1.62)•	**59.00(1.11)**	62.55(1.98)•	**58.95(0.88)**
Airfoil Self Noise	18.24(3.71)•	**17.23(3.81)**	15.61(3.82)•	**14.60(3.70)**	14.56(3.79)•	**13.13(1.07)**	13.43(0.92)•	**12.69(0.77)**	12.59(0.89)•	**12.05(0.79)**	12.21(0.81)•	**11.68(0.70)**
Bank32NH	109.73 (3.80)•	**106.42 (1.54)**	110.97 (3.00)•	**108.16 (1.81)**	113.20 (2.97)•	**109.00 (1.96)**	113.82 (2.45)•	**109.31 (1.72)**	115.23 (3.36)•	**109.73 (2.54)**	116.58 (3.41)•	**111.35 (3.27)**
Bank8FM	**13.05(0.15)**	13.07(0.14)	**13.12(0.18)**	13.13(0.17)	13.29(0.32)	**13.17(0.15)**	13.30(0.16)	**13.26(0.14)**	13.34(0.12)	**13.29(0.15)**	13.40(0.16)•	**13.32(0.18)**
Breast Cancer	716.15 (13.38)	**714.54 (14.81)**	**715.74 (14.92)**	718.17 (13.82)	**713.96 (12.80)**	716.78 (14.10)	**712.50 (12.42)**	715.17 (15.67)	**713.02 (14.21)**	713.24 (25.73)	712.96 (14.47)	**712.25 (20.84)**
CCPP	26.73(0.29)	**26.68(0.53)**	**26.84(0.28)**	27.27(0.67)	**26.94(0.28)**	27.23(0.54)	**27.16(0.24)**	27.31(0.61)	27.36(0.31)•	**26.97(0.75)**	27.48(0.30)•	**26.96(0.59)**
Comp Act	5.84(0.19)•	**5.65(0.08)**	5.89(0.73)•	**5.60(0.12)**	5.90(0.69)•	**5.65(0.16)**	5.97(0.71)•	**5.70(0.29)**	5.85(0.34)•	**5.65(0.14)**	5.88(0.34)•	**5.69(0.17)**
Comp Act Small	8.55(0.54)•	**8.31(0.11)**	8.57(1.02)•	**8.14(0.14)**	8.93(1.61)•	**8.04(0.14)**	8.62(1.18)•	**8.00(0.12)**	8.64(1.20)•	**7.96(0.12)**	8.64(1.19)•	**7.93(0.08)**
Concrete	47.93(36.21)	**39.40(7.20)**	43.89 (15.05)•	**36.31(3.46)**	41.24 (13.66)	**37.96(9.44)**	**41.30 (13.29)**	42.48 (21.48)	37.20(4.19)•	**35.47(3.07)**	35.46(3.39)	**35.33(4.38)**
Delta Ailerons	**14.79(0.04)**	14.80(0.06)	**14.71(0.04)**	14.75(0.05)	**14.68(0.04)**	14.72(0.05)	**14.66(0.04)**	14.69(0.05)	**14.64(0.04)**	14.67(0.04)	**14.63(0.04)**	14.65(0.06)
Delta Elevators	28.37(0.76)•	**28.02(0.12)**	28.61(0.69)•	**28.11(0.13)**	29.24(2.09)•	**28.22(0.14)**	29.38(1.85)•	**28.22(0.21)**	29.67(1.96)•	**28.20(0.13)**	29.83(1.94)•	**28.24(0.19)**
Housing	58.76(8.35)•	**52.56(5.44)**	55.03(6.24)•	**51.71(5.27)**	56.20(6.15)•	**50.07(6.02)**	56.31(7.32)•	**52.22(6.21)**	55.77(6.38)•	**49.36(6.03)**	54.83(5.30)•	**51.93(5.43)**
Kinematics	31.60(0.36)•	**31.42(0.39)**	30.23(0.25)•	**30.02(0.27)**	29.64(0.29)•	**29.52(0.26)**	29.32(0.29)•	**29.21(0.25)**	29.06(0.24)•	**28.93(0.25)**	28.89(0.27)•	**28.78(0.29)**
Machine	**57.06(11.97)**	58.66 (11.42)	**50.48(10.60)**	51.68 (10.26)	**49.18 (10.87)**	50.47 (10.37)	**48.33 (10.88)**	49.29 (10.54)	**48.00(11.06)**	49.67 (10.80)	**47.44(11.00)**	48.18 (10.98)
Puma32H	13.05(0.24)•	**12.92(0.27)**	13.01(0.16)•	**12.86(0.25)**	12.98(0.14)•	**12.83(0.25)**	13.05(0.14)•	**12.79(0.21)**	13.10(0.16)•	**12.85(0.24)**	13.17(0.12)•	**12.84(0.15)**
Puma8NH	173.52 (0.72)•	**173.01 (0.75)**	174.63 (0.92)•	**173.51 (0.90)**	175.31 (0.80)•	**173.30 (0.84)**	175.85 (0.88)•	**173.51 (0.77)**	176.81 (0.90)•	**173.22 (1.00)**	177.48 (0.97)•	**173.19 (0.98)**
Stock	5.95(0.32)•	**5.82(0.27)**	5.95(0.35)	**5.93(0.28)**	**5.90(0.30)**	**5.90(0.26)**	5.91(0.25)	**5.85(0.24)**	5.93(0.26)	**5.85(0.29)**	5.96(0.27)	**5.94(0.28)**
Triazines	**207.96(8.81)**	208.78 (8.31)	**206.84(7.69)**	208.86 (11.21)	**206.37 (7.79)**	208.95 (11.87)	**205.76 (6.36)**	208.38 (10.66)	**204.88(7.06)**	206.75 (9.82)	**202.31(7.34)**	206.47 (11.12)
Wine Q. Red	164.18(1.49)	**164.00 (1.53)**	**163.47(1.16)**	163.74 (1.36)	163.62 (1.20)	**163.61 (1.11)**	163.76 (1.13)	**163.66 (1.13)**	163.99(1.40)	**163.92 (1.27)**	164.23(1.07)	**164.18 (1.21)**
Wine Q. White	142.28 (4.21)•	**136.39 (1.44)**	141.83 (3.83)•	**136.74 (3.27)**	142.11 (3.61)•	**136.15 (1.24)**	142.61 (3.78)•	**136.21 (1.50)**	150.11 (21.26)•	**137.29 (2.18)**	153.72 (22.37)•	**136.43 (1.29)**
Win/Tie/Loss	4/0/16	**16/0/4**	7/0/13	**13/0/7**	5/1/14	**14/1/5**	6/0/14	**14/0/6**	4/0/16	**16/0/4**	3/0/17	**17/0/3**

Dataset	50		60		70		80		90		100	
	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S
Abalone	62.95(2.35)•	**58.85(1.15)**	64.00(3.43)•	**59.05(1.09)**	64.37(3.30)•	**59.20(1.00)**	64.87(3.31)•	**59.32(1.06)**	65.49(3.46)•	**59.53(1.04)**	65.64(3.33)•	**59.06(1.05)**
Airfoil Self Noise	11.91(0.76)•	**11.35(0.68)**	11.64(0.83)•	**11.21(0.66)**	11.57(0.84)•	**11.05(0.70)**	11.67(1.71)•	**11.18(1.75)**	11.55(1.66)•	**10.75(0.66)**	11.42(1.71)•	**10.84(0.88)**
Bank32NH	117.44 (3.60)•	**110.97 (2.65)**	118.70 (4.81)•	**111.49 (2.29)**	118.97 (4.83)•	**111.60 (2.60)**	119.55 (4.95)•	**111.08 (2.30)**	120.10 (5.14)•	**110.80 (3.31)**	120.60 (4.99)•	**110.87 (2.45)**
Bank8FM	13.51(0.22)•	**13.32(0.20)**	13.56(0.22)•	**13.40(0.30)**	13.67(0.29)•	**13.41(0.26)**	13.75(0.32)•	**13.46(0.23)**	13.81(0.34)•	**13.45(0.25)**	13.87(0.41)•	**13.53(0.43)**
Breast Cancer	712.68 (13.54)	**707.98 (17.93)**	713.75 (12.22)•	**705.98 (14.15)**	712.96 (12.53)	**710.66 (14.10)**	712.30 (12.55)	**707.22 (18.53)**	713.54 (12.46)	**711.42 (17.19)**	713.60 (10.68)	**713.56 (19.43)**
CCPP	27.58(0.43)•	**26.62(0.64)**	27.68(0.43)•	**26.64(0.54)**	27.78(0.46)•	**26.61(0.61)**	27.91(0.44)•	**26.41(0.66)**	28.01(0.50)•	**26.35(0.99)**	28.09(0.46)•	**26.22(0.63)**

Table 4 (*continued*)

Dataset	5		10		15		20		30		40	
	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S	DS	MINE-S
Comp Act	5.90(0.38)•	**5.69(0.15)**	5.96(0.53)•	**5.72(0.16)**	5.96(0.52)	**5.75(0.21)**	5.96(0.43)	**5.70(0.19)**	5.91(0.40)	**5.80(0.30)**	5.88(0.33)	**5.75(0.28)**
Comp Act Small	8.63(1.23)•	**7.95(0.10)**	8.63(1.21)•	**7.91(0.10)**	8.65(1.18)•	**7.90(0.10)**	8.67(1.24)•	**7.92(0.15)**	8.74(1.26)•	**7.89(0.11)**	8.64(1.20)•	**7.88(0.10)**
Concrete	36.05(4.50)	**35.33(4.01)**	35.59(4.38)	**35.52(4.04)**	35.23(4.25)	**35.18(4.72)**	35.30(3.83)	**34.85(4.39)**	36.01(4.85)	**34.89(3.71)**	36.07(4.60)	**34.42(3.33)**
Delta Ailerons	**14.62(0.04)**	14.64(0.04)	**14.61(0.05)**	14.63(0.06)	**14.60(0.05)**	14.63(0.06)	**14.60(0.04)**	14.62(0.07)	**14.59(0.04)**	14.62(0.06)	**14.59(0.04)**	14.62(0.06)
Delta Elevators	29.94(1.92)•	**28.30(0.16)**	30.11(1.89)•	**28.23(0.21)**	30.10(1.88)•	**28.27(0.14)**	30.21(1.86)•	**28.26(0.19)**	30.61(2.43)•	**28.27(0.17)**	30.69(2.45)•	**28.25(0.18)**
Housing	54.38(4.46)•	**51.92(5.41)**	55.24(6.27)•	**50.84(4.90)**	55.15(6.01)•	**50.45(5.41)**	55.32(5.91)•	**51.89(5.38)**	55.15(6.52)•	**51.86(8.07)**	54.68(6.47)	**53.22(6.95)**
Kinematics	28.71(0.23)•	**28.62(0.26)**	28.62(0.29)	**28.56(0.26)**	28.52(0.28)	**28.47(0.27)**	28.50(0.25)•	**28.42(0.26)**	28.45(0.22)•	**28.35(0.25)**	28.43(0.24)	**28.36(0.26)**
Machine	**47.55(10.97)**	48.73 (11.47)	**47.09(10.87)**	49.00 (10.54)	**46.99 (10.88)**	48.17 (10.80)	**46.93 (10.91)**	48.27 (11.12)	**46.85(10.82)**	48.81 (10.34)	**47.01(10.90)**	48.59 (10.50)
Puma32H	13.19(0.12)•	**12.88(0.21)**	13.20(0.15)•	**12.90(0.21)**	13.23(0.16)•	**12.87(0.18)**	13.20(0.17)•	**12.92(0.21)**	13.22(0.17)•	**12.89(0.23)**	13.27(0.18)•	**12.98(0.20)**
Puma8NH	177.99 (1.05)•	**173.28 (0.62)**	178.27 (1.06)•	**173.33 (0.90)**	178.79 (1.00)•	**173.25 (0.72)**	179.17 (1.24)•	**173.39 (0.80)**	179.45 (1.22)•	**173.22 (0.85)**	179.73 (1.22)•	**173.43 (0.79)**
Stock	5.89(0.27)	**5.87(0.31)**	**5.87(0.26)**	5.90(0.29)	5.85(0.28)	**5.81(0.27)**	**5.84(0.34)**	5.85(0.31)	**5.85(0.34)**	5.88(0.24)	5.89(0.37)	**5.88(0.32)**
Triazines	**201.48(7.83)**	207.24 (10.12)	**201.58(7.45)**	205.07 (10.62)	**202.65 (7.83)**	206.92 (13.75)	**201.60 (7.96)**	207.14 (11.10)	**201.72(7.83)**	210.19 (12.61)	**201.23(7.85)**	210.55 (11.21)
Wine Q. Red	**164.26(1.06)**	164.58 (1.08)	**164.00(1.33)**	164.06 (1.07)	**164.02 (1.28)**	164.36 (2.02)	**164.10 (1.14)**	164.24 (1.44)	**164.14(1.07)**	164.70 (1.39)	**164.20(1.03)**	164.38 (1.16)
Wine Q. White	152.05 (21.02)•	**137.08 (2.14)**	152.02 (21.08)•	**136.90 (1.31)**	146.27 (6.54)•	**137.56 (2.29)**	147.22 (6.37)•	**137.70 (2.10)**	149.48 (8.00)•	**138.09 (2.41)**	148.92 (7.32)•	**138.89 (5.78)**
Win/Tie/Loss	4/0/16	**16/0/4**	5/0/15	**15/0/5**	4/0/16	**16/0/4**	5/0/15	**15/0/5**	5/0/15	**15/0/5**	4/0/16	**16/0/4**

Table 5
Mean and standard deviation of the results calculated in 20 replications. For each dataset, the best result is in bold. Line "Win/Tie/Loss" shows the total of the results compared with MINE-W. The values marked with a • indicate that the null hypothesis must be rejected ($pValue \leqslant 0.05$), in other words, the result of MINE-W achieves superior performance. The values are in the scale 10^{-4}.

Dataset	5		10		15		20		30		40	
	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W
Abalone	55.32 (0.39)•	**55.25 (0.40)**	54.90 (0.28)	**54.87 (0.24)**	54.74 (0.21)	**54.72 (0.20)**	**54.68 (0.18)**	**54.68 (0.19)**	**54.62 (0.18)**	4.63 (0.22)	54.59 (0.19)	**54.58 (0.20)**
Airfoil Self Noise	20.45 (0.70)•	**16.81 (1.14)**	18.65 (0.61)•	**14.33 (1.02)**	18.15 (0.54)•	**13.46 (0.72)**	17.92 (0.50)•	**13.07 (0.67)**	17.69 (0.50)•	**12.56 (0.54)**	17.54 (0.42)•	**12.29 (0.46)**
Bank32NH	92.69 (0.68)•	**92.52 (0.64)**	**90.42 (0.43)**	90.44 (0.43)	**89.76 (0.46)**	89.84 (0.45)	**89.45 (0.34)**	89.55 (0.35)	**89.09 (0.29)**	89.23 (0.30)	**88.93 (0.28)**	89.06 (0.31)
Bank8FM	**12.45 (0.07)**	**12.45 (0.07)**	12.35 (0.05)•	**12.34 (0.05)**	12.31 (0.04)•	**12.30 (0.04)**	12.30 (0.04)•	**12.29 (0.04)**	12.27 (0.03)•	**12.26 (0.03)**	12.26 (0.03)•	**12.25 (0.03)**
Breast Cancer	**722.52 (11.52)**	724.68 (12.42)	**718.31 (9.00)**	721.26 (10.58)	**716.69 (7.31)**	719.39 (7.62)	**716.53 (7.12)**	720.59 (7.49)	**716.33 (7.17)**	719.81 (7.88)	**715.85 (6.95)**	719.12 (7.46)
CCPP	24.16 (0.17)•	**24.03 (0.16)**	23.61 (0.13)•	**23.48 (0.13)**	23.42 (0.12)•	**23.29 (0.12)**	23.32 (0.12)•	**23.18 (0.11)**	23.23 (0.12)•	**23.09 (0.11)**	23.18 (0.12)•	**23.04 (0.11)**
Comp Act	5.51 (0.06)•	**5.43 (0.05)**	5.41 (0.05)•	**5.34 (0.05)**	5.37 (0.03)•	**5.31 (0.04)**	5.36 (0.03)•	**5.30 (0.03)**	5.34 (0.02)•	**5.29 (0.03)**	5.33 (0.02)•	**5.28 (0.03)**
Comp Act Small	8.52 (0.08)•	**8.37 (0.05)**	8.41 (0.09)•	**8.24 (0.06)**	8.36 (0.06)•	**8.18 (0.04)**	8.33 (0.04)•	**8.15 (0.03)**	8.31 (0.04)•	**8.12 (0.03)**	8.30 (0.03)•	**8.10 (0.03)**
Concrete	39.35 (4.44)•	**35.71 (1.35)**	36.47 (1.71)•	**33.25 (1.30)**	35.73 (1.15)•	**32.52 (1.09)**	35.32 (1.04)•	**32.17 (1.19)**	34.90 (1.20)•	**32.32 (4.04)**	34.55 (1.06)•	**31.63 (2.63)**
Delta Ailerons	15.04 (0.04)•	**15.03 (0.04)**	15.03 (0.03)•	**15.02 (0.03)**	**15.02 (0.02)**	**15.02 (0.03)**	**15.02 (0.03)**	**15.02 (0.03)**	15.02 (0.02)•	**15.01 (0.03)**	15.02 (0.02)•	**15.01 (0.02)**
Delta Elevators	27.69 (0.77)•	**27.50 (0.08)**	27.45 (0.19)	**27.41 (0.05)**	27.40 (0.10)	**27.39 (0.04)**	**27.37 (0.06)**	27.38 (0.04)	**27.34 (0.03)**	27.35 (0.03)	**27.33 (0.03)**	27.34 (0.03)
Housing	54.98 (5.06)•	**50.74 (3.16)**	52.69 (2.93)•	**49.25 (2.64)**	51.41 (3.04)•	**48.20 (2.74)**	50.79 (3.21)•	**47.78 (2.95)**	50.36 (2.92)•	**47.38 (2.67)**	50.12 (2.97)•	**47.37 (2.96)**
Kinematics	33.28 (0.36)•	**32.14 (0.35)**	32.38 (0.33)•	**31.00 (0.29)**	32.13 (0.28)•	**30.63 (0.25)**	31.98 (0.23)•	**30.42 (0.21)**	31.83 (0.19)•	**30.20 (0.17)**	31.78 (0.17)•	**30.10 (0.16)**
Machine	**71.19 (9.13)**	74.52 (7.05)	**68.83 (8.60)**	71.32 (8.24)	**68.49 (8.06)**	71.17 (7.74)	**68.10 (7.96)**	71.55 (7.80)	**67.71 (7.91)**	72.15 (7.92)	**67.67 (7.89)**	70.45 (7.71)
Puma32H	11.26 (0.19)•	**11.25 (0.18)**	10.94 (0.15)•	**10.93 (0.14)**	10.82 (0.10)•	**10.81 (0.09)**	10.79 (0.08)•	**10.77 (0.07)**	10.73 (0.05)•	**10.72 (0.05)**	10.72 (0.04)•	**10.70 (0.04)**
Puma8NH	**168.08 (0.31)**	168.13 (0.31)	**167.51 (0.28)**	167.55 (0.27)	**167.30 (0.24)**	167.34 (0.22)	**167.19 (0.19)**	167.23 (0.19)	**167.11 (0.16)**	167.15 (0.15)	**167.05 (0.17)**	167.09 (0.16)
Stock	5.40 (0.21)•	**5.23 (0.22)**	5.18 (0.19)•	**5.02 (0.20)**	5.11 (0.17)•	**4.97 (0.17)**	5.07 (0.16)•	**4.94 (0.16)**	5.03 (0.16)•	**4.91 (0.17)**	5.02 (0.16)•	**4.90 (0.17)**
Triazines	207.73 (6.57)	**207.58 (6.65)**	**206.08 (5.43)**	206.77 (5.79)	**206.28 (5.49)**	206.51 (6.25)	**206.46 (5.28)**	207.71 (6.56)	**205.93 (5.24)**	207.53 (7.09)	**205.53 (4.95)**	206.28 (6.08)
Wine Q. Red	164.74 (1.23)•	**164.65 (1.34)**	164.31 (0.85)	**164.25 (0.94)**	164.25 (0.66)•	**164.16 (0.68)**	164.15 (0.63)•	**164.09 (0.66)**	164.09 (0.65)•	**164.03 (0.69)**	164.04 (0.64)•	**163.99 (0.69)**
Wine Q. White	135.45 (1.48)•	**133.82 (1.36)**	133.75 (1.06)•	**132.30 (0.97)**	133.17 (0.71)•	**131.69 (0.73)**	132.91 (0.66)•	**131.50 (0.79)**	132.67 (0.65)•	**131.14 (0.52)**	132.48 (0.59)•	**131.18 (0.76)**
Win/Tie/Loss	3/1/16	**16/1/3**	5/0/15	**15/0/5**	5/1/14	**14/1/5**	6/2/12	**12/2/6**	7/0/13	**13/0/7**	6/0/14	**14/0/6**

Dataset	50		60		70		80		90		100	
	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W
Abalone	**54.55 (0.20)**	**54.55 (0.21)**	54.55 (0.23)	**54.54 (0.23)**	**54.53 (0.20)**	**54.53 (0.21)**	54.51 (0.19)	**54.50 (0.19)**	**54.50 (0.18)**	**54.50 (0.18)**	**54.50 (0.18)**	**54.50 (0.17)**
Airfoil Self Noise	17.48 (0.41)•	**12.08 (0.40)**	17.41 (0.39)•	**11.93 (0.38)**	17.40 (0.38)•	**11.85 (0.35)**	17.35 (0.38)•	**11.75 (0.33)**	17.34 (0.36)•	**11.69 (0.31)**	17.33 (0.37)•	**11.65 (0.33)**
Bank32NH	**88.83 (0.28)**	88.95 (0.34)	**88.77 (0.31)**	88.91 (0.37)	**88.75 (0.32)**	88.86 (0.37)	**88.70 (0.29)**	88.83 (0.34)	**88.69 (0.29)**	88.78 (0.33)	**88.68 (0.27)**	88.77 (0.29)
Bank8FM	12.26 (0.03)•	**12.24 (0.03)**	12.25 (0.03)•	**12.24 (0.03)**	12.25 (0.02)•	**12.23 (0.03)**	12.24 (0.03)•	**12.23 (0.03)**	12.24 (0.02)•	**12.23 (0.03)**	12.24 (0.02)•	**12.23 (0.03)**
Breast Cancer	**715.39 (6.69)**	720.70 (7.30)	**715.11 (6.83)**	720.00 (7.64)	**714.94 (6.57)**	718.83 (7.45)	**714.75 (6.33)**	718.36 (7.94)	**714.82 (6.55)**	718.09 (8.42)	**714.91 (6.53)**	719.04 (8.18)
CCPP	23.14 (0.13)•	**23.00 (0.12)**	23.12 (0.12)•	**22.98 (0.12)**	23.10 (0.12)•	**22.96 (0.12)**	23.10 (0.13)•	**22.96 (0.12)**	23.09 (0.13)•	**22.95 (0.12)**	23.09 (0.12)•	**22.95 (0.12)**
Comp Act	5.32 (0.02)•	**5.28 (0.03)**	5.32 (0.02)•	**5.27 (0.02)**	5.32 (0.02)•	**5.27 (0.02)**	5.32 (0.02)•	**5.27 (0.02)**	5.31 (0.02)•	**5.26 (0.02)**	5.31 (0.02)•	**5.26 (0.02)**
Comp Act Small	8.29 (0.02)•	**8.10 (0.03)**	8.29 (0.03)•	**8.10 (0.05)**	8.28 (0.03)•	**8.09 (0.04)**	8.28 (0.02)•	**8.08 (0.03)**	8.28 (0.03)•	**8.08 (0.03)**	8.28 (0.03)•	**8.08 (0.03)**

Table 5 (*continued*)

Dataset	5		10		15		20		30		40	
	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W	DW	MINE-W
Concrete	34.37 (1.02)•	**31.23 (2.02)**	34.30 (1.03)•	**31.11 (1.91)**	34.21 (0.94)•	**30.90 (1.55)**	34.16 (0.93)•	**30.76 (1.36)**	34.13 (0.91)•	**30.69 (1.21)**	34.17 (0.88)•	**30.74 (1.18)**
Delta Ailerons	15.02 (0.02)•	**15.01 (0.02)**	15.02 (0.02)•	**15.01 (0.02)**	15.02 (0.02)•	**15.01 (0.02)**	15.02 (0.02)•	**15.01 (0.02)**	15.02 (0.02)•	**15.00 (0.02)**	15.02 (0.02)•	**15.00 (0.02)**
Delta Elevators	**27.32 (0.03)**	27.34 (0.04)	**27.32 (0.03)**	27.33 (0.04)	**27.31 (0.03)**	27.33 (0.04)	**27.31 (0.03)**	27.33 (0.04)	**27.31 (0.03)**	27.32 (0.03)	**27.31 (0.03)**	27.32 (0.03)
Housing	50.08 (2.89)•	**47.23 (3.05)**	49.91 (2.79)•	**47.14 (2.83)**	49.89 (2.80)•	**47.17 (2.80)**	49.86 (2.84)•	**47.12 (2.73)**	49.79 (2.85)•	**47.08 (2.80)**	49.77 (2.90)•	**47.06 (2.92)**
Kinematics	31.73 (0.16)•	**30.02 (0.15)**	31.70 (0.11)•	**29.97 (0.13)**	31.66 (0.11)•	**29.90 (0.11)**	31.65 (0.10)•	**29.88 (0.12)**	31.64 (0.09)•	**29.86 (0.10)**	31.63 (0.08)•	**29.83 (0.09)**
Machine	**67.48 (7.73)**	69.61 (7.74)	**67.04 (7.66)**	69.95 (7.56)	**66.84 (7.66)**	68.91 (7.09)	**66.62 (7.64)**	69.77 (6.77)	**66.70 (7.63)**	70.13 (6.39)	**66.79 (7.74)**	70.10 (6.62)
Puma32H	10.70 (0.04)•	**10.69 (0.04)**	10.68 (0.03)•	**10.67 (0.03)**	10.68 (0.03)•	**10.66 (0.03)**	10.67 (0.03)•	**10.65 (0.03)**	10.67 (0.03)•	**10.65 (0.03)**	10.66 (0.03)•	**10.65 (0.03)**
Puma8NH	**167.00 (0.18)**	167.05 (0.17)	**166.98 (0.17)**	167.04 (0.16)	**166.96 (0.17)**	167.01 (0.16)	**166.95 (0.16)**	167.00 (0.16)	**166.94 (0.16)**	167.00 (0.16)	**166.94 (0.15)**	166.99 (0.15)
Stock	4.99 (0.16)•	**4.88 (0.17)**	4.98 (0.15)•	**4.86 (0.16)**	4.97 (0.15)•	**4.86 (0.16)**	4.97 (0.15)•	**4.86 (0.16)**	4.97 (0.15)•	**4.85 (0.16)**	4.97 (0.15)•	**4.86 (0.16)**
Triazines	**205.39 (4.89)**	206.31 (6.01)	**205.31 (4.95)**	206.63 (6.21)	**205.33 (4.93)**	206.16 (5.66)	**205.25 (5.08)**	206.17 (5.62)	**205.23 (5.05)**	206.04 (5.00)	**205.12 (5.14)**	206.05 (5.44)
Wine Q. Red	164.00 (0.60)	**163.95 (0.67)**	163.98 (0.65)•	**163.92 (0.70)**	163.96 (0.66)•	**163.88 (0.70)**	163.97 (0.63)•	**163.90 (0.66)**	163.97 (0.61)•	**163.91 (0.65)**	163.96 (0.62)•	**163.91 (0.67)**
Wine Q. White	132.43 (0.52)•	**131.01 (0.75)**	132.37 (0.55)•	**130.89 (0.63)**	132.32 (0.50)•	**130.76 (0.45)**	132.30 (0.51)•	**130.69 (0.45)**	132.27 (0.46)•	**130.68 (0.42)**	132.23 (0.43)•	**130.61 (0.44)**
Win/Tie/Loss	6/1/13	**13/1/6**	6/0/14	**14/0/6**	6/1/13	**13/1/6**	6/0/14	**14/0/6**	6/1/13	**13/1/6**	6/1/13	**13/1/6**

In [44], an empirical study using time-series data concluded that the error rate decreases when the number of models in the combination increases. To the best of our knowledge, there is no such study for regression problems. To fulfill this gap, we present results varying the ensemble size, and it is possible to verify that the error rates decrease when the ensemble size increases, as shown in Tables 5 and 6.

4.6. Comparing MINE with static techniques

This section compares the three MINE techniques against Individual Regressor, and the combination of the regressors using Mean and Median as the combination rule (Table 7). The "Individual Regressor" column shows the results per dataset when only one regressor is applied. Each dataset is trained using the best performing regressor as listed in Table 3. In the previous section, the best results in Tables 5 and 6 were obtained with ensemble sizes of 80, 90, and 100 regressors. For the sake of simplicity, all the results in Table 7 use 90 regressors.

The "Individual Regressor" did not obtain the best performance in any of the used datasets, while Mean was better in 1 and Median in 2 out of 20 datasets. The MINE family obtained better results in 17 out of 20 datasets (MINE-S was better in 3, MINE-W in 7, and MINE-WS also in 7 out of 20 datasets). In general, combining the models has better performance than selecting just one. MINE-WS deserves a special highlight because it obtained similar performance when compared with MINE-W, however, it uses only a subset of the ensemble while MINE-W uses the whole ensemble.

4.7. Computational effort

Our proposal has four phases. The first phase (Learning Algorithm Selection) aims at choosing the best learning algorithm for each regression problem. The execution time of this phase is similar to any of the techniques presented in the paper. The second phase (Generation Phase) also has equal processing time for all techniques presented in the paper, since its task is only to generate the ensemble.

Phase 3 (Optimization Phase) is only performed by the MINE framework. It uses a genetic algorithm, which is a well-known time-consuming task. However, it is important to highlight that this task is performed offline, during the system's training. Once the genetic algorithm defines the best set of weights, this set is not modified in the next phase (Generalization Phase).

The computational effort of phase 4 is slightly different comparing the proposal with the literature DRS algorithms. While the algorithms in the literature extract only one error measure from the region of competence, the proposed algorithm extracts eight. However, the difference regarding the processing time is minimal.

Table 6
Mean and standard deviation of the results calculated in 20 replications. For each dataset, the best result is in bold. Line "Win/Tie/Loss" shows the total of the results compared with MINE-WS. The values marked with a • indicate that the null hypothesis must be rejected ($pValue \leqslant 0.05$), in other words, the result of MINE-WS achieves superior performance. The values are in the scale 10^{-4}.

Dataset	5		10		15		20		30		40	
	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS
Abalone	55.34(0.40)	**55.29(0.39)**	54.94(0.30)	**54.87(0.29)**	54.81(0.23)	**54.78(0.24)**	**54.76(0.20)**	54.78(0.21)	**54.69(0.18)**	54.75(0.35)	**54.77(0.28)**	54.80(0.30)
Airfoil Self Noise	19.45(0.83)•	**15.73(1.06)**	16.52(0.88)•	**13.34(0.79)**	15.19(0.79)•	**12.78(0.67)**	14.59(0.70)•	**12.54(0.55)**	13.81(0.59)•	**12.55(0.43)**	13.56(0.53)•	**12.35(0.51)**
Bank32NH	92.87(0.80)•	**92.65(0.67)**	90.93(0.68)•	**90.72(0.47)**	90.66(0.68)•	**90.22(0.42)**	90.56(0.63)•	**90.15(0.39)**	90.84(0.81)•	**90.28(0.88)**	91.17(1.02)•	**90.36(0.52)**
Bank8FM	**12.45(0.07)**	12.47(0.08)	**12.35(0.05)**	12.37(0.06)	**12.31(0.04)**	12.35(0.05)	**12.30(0.05)**	12.35(0.07)	**12.28(0.05)**	12.34(0.08)	**12.27(0.05)**	12.32(0.05)
Breast Cancer	**722.52 (11.52)**	724.90 (14.29)	**718.31 (9.00)**	722.44 (13.81)	**716.69 (7.31)**	720.95 (11.54)	**716.53 (7.12)**	720.81 (11.65)	**716.33 (7.17)**	722.07 (11.50)	**715.85 (6.95)**	718.61 (10.98)
CCPP	24.21(0.22)•	**24.06(0.20)**	23.74(0.21)•	**23.54(0.19)**	23.65(0.21)•	**23.32(0.15)**	23.62(0.19)•	**23.26(0.17)**	23.65(0.21)•	**23.19(0.19)**	23.67(0.20)•	**23.13(0.18)**
Comp Act	5.54(0.12)•	**5.47(0.06)**	5.45(0.08)•	**5.37(0.06)**	5.42(0.05)•	**5.35(0.09)**	5.41(0.05)•	**5.31(0.05)**	5.41(0.12)•	**5.26(0.04)**	5.38(0.03)•	**5.25(0.06)**
Comp Act Small	8.53(0.08)•	**8.37(0.05)**	8.51(0.40)•	**8.20(0.08)**	8.42(0.12)•	**8.08(0.06)**	8.38(0.12)•	**8.02(0.04)**	8.34(0.11)•	**7.96(0.06)**	8.31(0.08)•	**7.93(0.05)**
Concrete	43.92 (27.89)•	**35.87(1.61)**	35.89(3.11)•	**33.97(3.01)**	34.31(2.35)•	**33.55(5.73)**	33.74(2.09)•	**32.02(1.27)**	32.95(2.79)•	**31.33(1.59)**	31.94(1.61)•	**31.17(1.66)**
Delta Ailerons	**15.04(0.04)**	**15.04(0.04)**	**15.03(0.03)**	**15.03(0.03)**	**15.02(0.02)**	**15.02(0.03)**	**15.02(0.03)**	**15.02(0.04)**	**15.02(0.02)**	15.03(0.03)	**15.02(0.02)**	**15.02(0.03)**
Delta Elevators	27.70(0.77)	**27.52(0.09)**	27.46(0.19)	**27.43(0.06)**	**27.41(0.11)**	27.45(0.11)	**27.39(0.07)**	27.40(0.08)	**27.36(0.05)**	27.37(0.05)	**27.39(0.16)**	27.40(0.10)
Housing	55.95(5.11)•	**51.39(3.60)**	53.56(3.07)•	**49.74(3.06)**	52.54(3.33)•	**48.70(3.85)**	51.89(3.50)•	**47.80(5.33)**	50.90(3.18)•	**47.01(4.12)**	50.63(3.39)•	**46.59(3.42)**
Kinematics	33.25(0.36)•	**31.76(0.32)**	32.27(0.33)•	**30.10(0.29)**	31.93(0.30)•	**29.26(0.27)**	31.72(0.24)•	**28.85(0.24)**	31.42(0.21)•	**28.38(0.17)**	31.25(0.18)•	**28.12(0.21)**
Machine	**66.01(11.05)**	67.05(10.32)	58.78(10.42)	**58.27(10.23)**	**55.10 (10.51)**	56.43(10.16)	**54.43 (10.30)**	57.41(10.22)	**53.90 (10.30)**	58.23(10.40)	**53.50(9.99)**	55.57(10.86)
Puma32H	**11.26(0.19)**	11.27(0.19)	**10.95(0.15)**	**10.95(0.15)**	**10.83(0.10)**	10.84(0.10)	**10.80(0.08)•**	**10.80(0.08)**	10.76(0.06)•	**10.75(0.05)**	10.76(0.05)•	**10.73(0.05)**
Puma8NH	**168.08(0.31)**	168.14(0.31)	**167.51 (0.28)**	167.56(0.27)	**167.30 (0.24)**	167.35(0.24)	**167.19 (0.19)**	167.24(0.17)	**167.11 (0.16)**	167.28(0.51)	**167.05 (0.16)**	167.14(0.18)
Stock	5.44(0.24)•	**5.39(0.21)**	5.31(0.22)•	**5.23(0.22)**	5.25(0.20)•	**5.15(0.22)**	5.17(0.19)•	**5.07(0.20)**	5.07(0.17)•	**4.91(0.13)**	5.02(0.16)•	**4.82(0.15)**
Triazines	**208.31(6.72)**	208.44(7.68)	**207.33 (5.65)**	208.45(7.43)	**208.08 (5.42)**	208.44(7.17)	**208.16 (5.45)**	210.21(8.38)	**207.65 (5.66)**	208.58(8.04)	**207.00 (5.50)**	207.87(6.16)
Wine Q. Red	164.68 (1.25)•	**164.50(1.27)**	164.16(0.95)	**164.12(1.08)**	**164.01 (0.69)**	164.06(0.83)	**163.86 (0.67)**	163.87(0.81)	163.75(0.72)	**163.70(0.79)**	**163.66 (0.76)**	163.68(0.77)
Wine Q. White	135.71 (1.66)•	**133.80(1.08)**	134.05 (1.44)•	**132.56(1.12)**	133.71 (1.10)•	**131.86(0.81)**	133.68 (1.22)•	**131.71(0.68)**	133.51 (1.41)•	**131.04(0.54)**	133.70 (2.06)•	**131.01(0.92)**
Win/Tie/Loss	6/1/13	**13/1/6**	4/2/14	**14/2/4**	8/1/11	**11/1/8**	8/2/10	**10/2/8**	8/0/12	**12/0/8**	8/1/11	**11/1/8**

Dataset	50		60		70		80		90		100	
	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS
Abalone	54.74(0.24)	**54.71(0.35)**	**54.77(0.28)**	54.78(0.30)	**54.76(0.25)**	54.85(0.33)	**54.76(0.43)**	54.81(0.32)	**54.78(0.47)**	55.13(0.61)	**54.80(0.40)**	54.93(0.34)
Airfoil Self Noise	13.25(0.41)•	**12.25(0.39)**	13.17(0.39)•	**12.17(0.39)**	13.07(0.44)•	**12.11(0.51)**	13.11(0.46)•	**12.14(0.42)**	13.17(0.45)•	**11.92(0.48)**	13.23(0.48)•	**12.03(0.45)**
Bank32NH	91.60(1.01)•	**90.62(0.60)**	91.90(0.98)•	**90.59(0.54)**	92.41(1.51)•	**90.43(0.92)**	92.68(1.73)•	**90.08(0.57)**	93.06(1.73)•	**89.94(0.57)**	93.25(1.38)•	**89.99(0.73)**
Bank8FM	**12.27(0.05)**	12.32(0.07)	**12.26(0.05)**	12.31(0.06)	**12.26(0.05)**	12.32(0.06)	**12.26(0.04)**	12.34(0.13)	**12.27(0.07)**	12.36(0.16)	**12.27(0.07)**	12.33(0.12)
Breast Cancer	**715.39(6.69)**	721.03(8.87)	**715.11 (6.83)**	721.21(9.86)	**714.94 (6.57)**	720.55 (10.53)	**714.75 (6.33)**	720.80 (10.90)	**714.82 (6.55)**	718.71 (10.24)	**714.91 (6.53)**	721.39 (12.96)
CCPP	23.70(0.26)•	**23.07(0.18)**	23.70(0.28)•	**23.06(0.19)**	23.74(0.28)•	**23.01(0.15)**	23.80(0.26)•	**23.03(0.16)**	23.84(0.29)•	**23.02(0.18)**	23.90(0.29)•	**23.00(0.15)**
Comp Act	5.36(0.03)•	**5.23(0.06)**	5.35(0.03)•	**5.22(0.02)**	5.34(0.03)•	**5.21(0.03)**	5.35(0.04)•	**5.21(0.05)**	5.35(0.04)•	**5.20(0.03)**	5.35(0.03)•	**5.19(0.02)**
Comp Act Small	8.29(0.07)•	**7.92(0.05)**	8.28(0.08)•	**7.90(0.05)**	8.27(0.08)•	**7.89(0.07)**	8.25(0.07)•	**7.87(0.06)**	8.23(0.05)•	**7.87(0.09)**	8.22(0.04)•	**7.85(0.05)**

Table 6 (*continued*)

Dataset	5		10		15		20		30		40	
	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS	DWS	MINE-WS
Concrete	31.89(1.54)•	**30.87(1.35)**	31.41(1.56)	**31.06(1.43)**	**31.02(1.42)**	**31.02(1.29)**	**31.03(1.39)**	31.18(1.56)	31.11(1.38)	**30.88(1.13)**	30.95(1.25)	**30.70(1.04)**
Delta Ailerons	**15.02(0.02)**	**15.02(0.03)**	**15.02(0.02)**	**15.02(0.03)**	15.02(0.02)	**15.01(0.03)**	**15.01(0.02)**	**15.01(0.03)**	**15.02(0.02)**	**15.02(0.03)**	**15.02(0.02)**	**15.02(0.03)**
Delta Elevators	27.39(0.17)	**27.37(0.06)**	**27.40(0.17)**	27.79(1.85)	27.39(0.17)	**27.36(0.06)**	27.51(0.53)	**27.38(0.06)**	27.51(0.53)	**27.41(0.17)**	27.52(0.53)	**27.40(0.11)**
Housing	50.64(3.49)•	**45.98(3.55)**	50.38(3.46)•	**45.64(3.67)**	50.27(3.22)•	**45.16(3.70)**	50.10(3.16)•	**45.04(3.15)**	49.98(3.21)•	**45.28(3.46)**	49.74(3.29)•	**44.39(2.60)**
Kinematics	31.08(0.17)•	**27.97(0.20)**	30.97(0.17)•	**27.84(0.17)**	30.83(0.16)•	**27.74(0.16)**	30.73(0.13)•	**27.71(0.16)**	30.67(0.12)•	**27.68(0.15)**	30.61(0.12)•	**27.62(0.14)**
Machine	**53.16(10.15)**	55.96(10.05)	**52.67(9.82)**	55.31(10.06)	**52.50(9.81)**	56.60(10.42)	**52.46(9.88)**	55.84(10.37)	**52.34(9.91)**	56.24(10.73)	**52.34(9.97)**	56.60(10.35)
Puma32H	10.76(0.05)•	**10.72(0.04)**	10.76(0.04)•	**10.71(0.04)**	10.77(0.05)•	**10.70(0.04)**	10.77(0.04)•	**10.70(0.04)**	10.78(0.04)•	**10.69(0.03)**	10.78(0.04)•	**10.69(0.04)**
Puma8NH	**167.01(0.17)**	167.08(0.20)	**166.99 (0.16)**	167.10(0.19)	**166.96 (0.17)**	167.08(0.19)	**166.96 (0.16)**	167.04(0.18)	**166.95 (0.17)**	167.02(0.15)	**166.95 (0.16)**	167.01(0.15)
Stock	4.98(0.17)•	**4.77(0.16)**	4.92(0.19)•	**4.75(0.15)**	4.87(0.18)•	**4.74(0.17)**	4.85(0.18)•	**4.75(0.16)**	4.84(0.16)•	**4.74(0.17)**	4.84(0.16)•	**4.74(0.15)**
Triazines	**206.74(5.54)**	206.86(6.49)	**206.49 (5.87)**	208.91(7.75)	**206.39 (5.74)**	209.36(6.57)	**206.78 (5.74)**	209.59(7.32)	**206.68 (5.82)**	210.11(7.24)	**206.69 (5.74)**	208.41(6.78)
Wine Q. Red	**163.63(0.73)**	163.68(0.78)	**163.59 (0.76)**	163.65(0.79)	**163.53 (0.75)**	163.69(0.67)	**163.54 (0.72)**	163.75(0.69)	**163.58 (0.74)**	163.73(0.69)	**163.55 (0.75)**	163.77(0.71)
Wine Q. White	133.65 (1.63)•	**130.69(0.70)**	133.50 (1.44)•	**130.79(1.58)**	133.43 (1.38)•	**130.70(1.44)**	133.30 (1.26)•	**130.64(1.58)**	133.19 (1.13)•	**130.11(0.76)**	133.11 (1.11)•	**130.54(1.28)**
Win/Tie/Loss	7/1/13	**13/1/7**	8/1/11	**11/1/8**	7/1/12	**12/1/7**	8/1/11	**11/1/8**	7/1/12	**12/1/7**	7/1/12	**12/1/7**

Table 7
Mean and standard deviation of the results calculated in 20 replications. For each dataset, the best result is in bold. The values are in the scale 10^{-4}. Ensemble Size = 90.

Dataset	Individual Regressor	Mean	Median	MINE-S	MINE-W	MINE-WS
Abalone	56.58(0.68)	54.51(0.19)	**54.46(0.17)**	59.53(1.04)	54.50(0.18)	55.13(0.61)
Airfoil Self Noise	32.00(4.54)	21.48(0.43)	20.89(0.4	10.75(0.66)	**11.69(0.31)**	11.92(0.48)
Bank32NH	98.04(0.96)	88.79(0.29)	89.15(0.25)	110.80(3.31)	**88.78(0.33)**	89.94(0.57)
Bank8FM	12.79(0.08)	12.27(0.02)	12.32(0.02)	13.45(0.25)	**12.23(0.03)**	12.36(0.16)
Breast Cancer	730.74(9.40)	715.72(6.52)	716.95(7.10)	**711.42(17.19)**	718.09(8.42)	718.71(10.24)
CCPP	24.46(0.15)	23.37(0.12)	23.42(0.12)	26.35(0.99)	**22.95(0.12)**	23.02(0.18)
Comp Act	6.03(0.16)	5.37(0.02)	5.38(0.02)	5.80(0.30)	5.26(0.02)	**5.20(0.03)**
Comp Act Small	9.28(0.20)	8.42(0.02)	8.45(0.02)	7.89(0.11)	8.08(0.03)	**7.87(0.09)**
Concrete	52.83(2.70)	39.36(0.89)	38.08(0.76)	34.89(3.71)	**30.69(1.21)**	30.88(1.13)
Delta Ailerons	15.05(0.02)	15.03(0.02)	15.03(0.02)	**14.62(0.06)**	15.00(0.02)	15.02(0.03)
Delta Elevators	27.76(0.10)	**27.32(0.03)**	27.35(0.03)	28.27(0.17)	**27.32(0.03)**	27.41(0.17)
Housing	55.79(4.32)	51.75(2.62)	51.01(2.25)	51.86(8.07)	47.08(2.80)	**45.28(3.46)**
Kinematics	39.61(1.03)	33.01(0.10)	33.04(0.12)	28.35(0.25)	29.86(0.10)	**27.68(0.15)**
Machine	82.03(5.37)	78.79(6.21)	81.56(5.44)	**48.81(10.34)**	70.13(6.39)	56.24(10.73)
Puma32H	12.29(0.43)	10.71(0.03)	**10.64(0.02)**	12.89(0.23)	10.65(0.03)	10.69(0.03)
Puma8NH	169.48(0.64)	**166.93(0.16)**	166.95(0.17)	173.22(0.85)	167.00(0.16)	167.02(0.15)
Stock	5.52(0.22)	5.26(0.14)	5.23(0.16)	5.88(0.24)	4.85(0.16)	**4.74(0.17)**
Triazines	211.96(4.99)	206.27(4.89)	209.16(5.19)	210.19(12.61)	**206.04(5.00)**	210.11(7.24)
Wine Q. Red	164.81(0.55)	164.30(0.59)	164.67(0.62)	164.70(1.39)	163.91(0.65)	**163.73(0.69)**
Wine Q. White	143.12(3.19)	133.35(0.39)	133.56(0.34)	138.09(2.41)	130.68(0.42)	**130.11(0.76)**

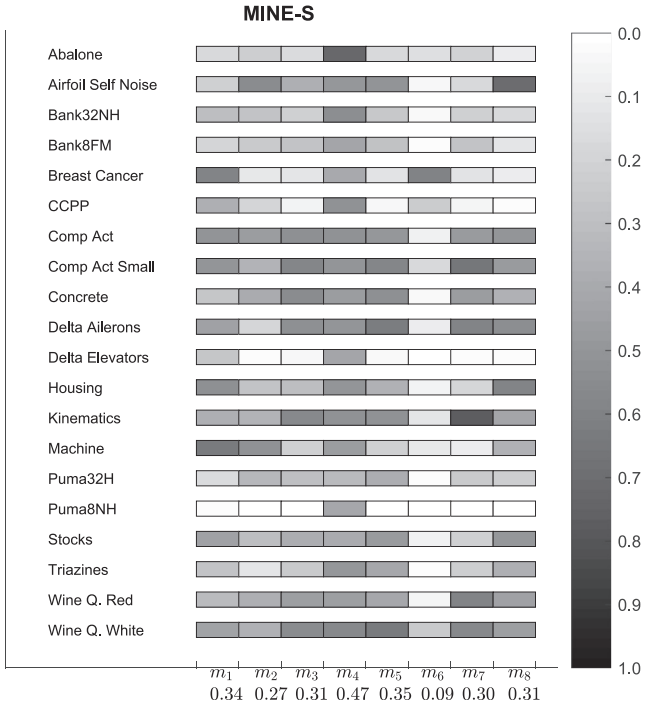

Fig. 2. Mean of the weights of the measures calculated for MINE-S.

All the experiments we performed on an AMD Ryzen 5 1600 6-core processor, 8 Gb DDR 4 2666Mhz and MatLab R2017a. The Generalization phase requires, on average, 2.29 ± 7.63 seconds per dataset. While the Optimization phase needs $85.26 \pm 96.50, 89.02 \pm 97.08$ and 95.65 ± 101.69 on average, per dataset, for MINE-S, MINE-W, and MINE-WS, respectively.

4.8. Evaluating the measures

As explained in the previous sections, all the eight measures presented in Section 3.3.1 were combined using a vector of weights $\mathcal{W}$ calculated in the Optimization Phase of the MINE framework. For each test pattern, the combination of the mea-

sures generates a new vector of weights $\mathcal{A}$ that was used to select the most competent regressor in MINE-S, to select and combine the regressors in MINE-WS, and to combine all the regressors in MINE-W.

Figs. 2–4 show the arithmetic mean of the weights over 20 replications per datasets for the MINE-S, MINE-W, and MINE-WS, respectively. These tables also show the mean of the weights per measure (these values are at the bottom of each figure).

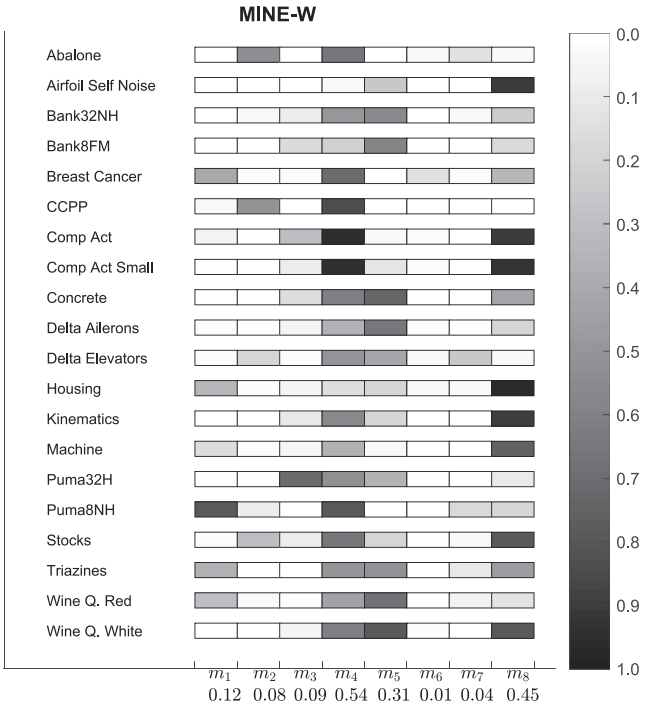

Fig. 3. Mean of the weights of the measures calculated for MINE-W.

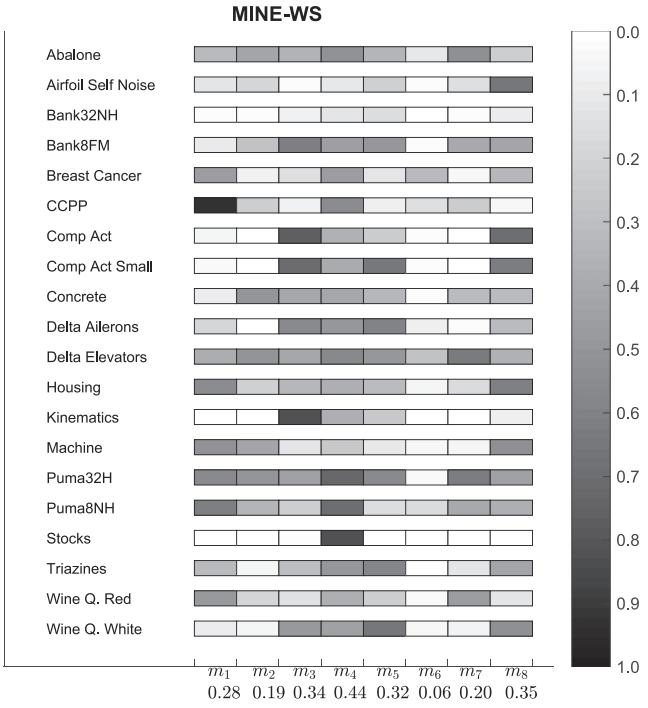

Fig. 4. Mean of the weights of the measures calculated for MINE-WS.

We can observe that the weights of the measures vary depending on the technique under analysis. In MINE-S, the range of the weights is wider than in MINE-W and MINE-WS. For instance, in MINE-W, some weights are zero or close to it. This means that this measure has little or no influence in the decision process, observe m_6, for instance. MINE-WS uses more measures, on average, when compared with MINE-W, but the values of the weights are not as high as in MINE-S.

This analysis shows that the importance of the measures varies from dataset to dataset and also from technique to technique indicating that their combination is more advantageous than using only one.

5. Conclusion

This paper proposed the MINE framework for dynamic regressor selection that aims to select and combine the best regressors per query pattern from a homogeneous ensemble. MINE uses information extracted from the region of competence as a criterion to select the competent regressors. Instead of using only one measure from the region of competence, knowing that no single measure is the best for any task, the proposal combines a set of measures to better select the competent regressors.

Three algorithms were presented, and their difference resides in how many regressors are selected from the ensemble. MINE-S selects only the most competent regressor while MINE-W combines all the regressors. MINE-WS, in turn, selects a subset of the regressors. Experiments showed that the MINE techniques presented in this work perform better compared to state-of-the-art DRS techniques, and classical combination techniques, such as Mean and Median. Among the MINE family, a highlight to MINE-WS because it performed similarly to MINE-W but required fewer regressors in the combination phase.

The results showed that the combination of multiple measures extracted from the region of competence generates more accurate results than using only a single measure. We also observed that some measures received zero-weight for some datasets. In other words, the set of measures is problem-dependent and can be selected instead of using all of them. The proposed framework is modular and can be evaluated using more significant set measures. Also, as presented in [20,27], the size of the region of competence is problem-dependent and for better error rates a study must be done to find the ideal neighborhood size for each dataset.

For future work, we intend to evaluate different optimization algorithms in the Optimization Phase, such as PSO (Particle Swarm Optimization) [34], and Differential Evolution [34]. We also intend to analyze some parameters of the framework, such as the size of the region of competence.

CRediT authorship contribution statement

Thiago J.M. Moura: Conceptualization, Methodology, Software, Validation, Writing - review & editing. **George D.C. Cavalcanti:** Conceptualization, Methodology, Writing - review & editing. **Luiz S. Oliveira:** Conceptualization, Methodology, Writing - review & editing.

Declaration of Competing 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.

Acknowledgments

This research has been supported by the following Brazilian agencies: CNPq (Conselho Nacional de Desenvolvimento Cientfico e Tecnológico), CAPES (Coordenação de Aperfeiçoamento de Pessoal de Nível Superior), and FACEPE (Fundação de Amparo à Ciência e Tecnologia de Pernambuco).

References

[1] T.K. Ho, The random subspace method for constructing decision forests, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (8) (1998) 832–844.
[2] C. Domeniconi, B. Yan, Nearest neighbor ensemble, in: International Conference, Pattern Recognition, vol. 1, 2004, pp. 228–231.
[3] S. Singh, M. Singh, A dynamic classifier selection and combination approach to image region labelling, Signal Processing: Image Communication 20 (3) (2005) 219–231.
[4] H.D. Monmouth, Improving regressors using boosting techniques, in: 14th International Conference on Machine Learning, 1997, pp. 107–115.
[5] D.L. Shrestha, D.P. Solomatine, Experiments with adaboost.rt, an improved boosting scheme for regression, Neural Computation 18 (7) (2006) 1678–1710.
[6] C.-X. Zhang, J.-S. Zhang, G.-W. Wang, An empirical study of using rotation forest to improve regressors, Applied Mathematics and Computation 195 (2) (2008) 618–629.
[7] R.M. Cruz, R. Sabourin, G.D. Cavalcanti, Dynamic classifier selection: Recent advances and perspectives, Information Fusion 41 (2018) 195–216.
[8] J.J. Rodríguez, L.I. Kuncheva, C.J. Alonso, Rotation forest: A new classifier ensemble method, IEEE Transactions on Pattern Analysis and Machine Intelligence 28 (10) (2006) 1619–1630.
[9] G. Giacinto, F. Roli, Methods for dynamic classifier selection, in: Proceedings of International Conference on Image Analysis and Processing, 1999, pp. 659–664.

[10] G. Giacinto, F. Roli, Dynamic classifier selection based on multiple classifier behaviour, Pattern Recognition 34 (9) (2001) 1879–1881.
[11] M.P. Perrone, L.N. Cooper, When networks disagree: Ensemble methods for hybrid neural networks, in: Neural Networks for Speech and Image Processing, 1993, pp. 123–140.
[12] D. Ortiz-Boyer, C. HervásMartínez, N. García-Pedrajas, Cixl2: A crossover operator for evolutionary algorithms based on population features, Journal of Artificial Intelligence Research 24 (1) (2005) 1–48.
[13] I. Partalas, G. Tsoumakas, E.V. Hatzikos, I. Vlahavas, Greedy regression ensemble selection: Theory and an application to water quality prediction, Information Sciences 178 (20) (2008) 3867–3879.
[14] A.H. Ko, R. Sabourin, A.S. Britto, From dynamic classifier selection to dynamic ensemble selection, Pattern Recognition 41 (5) (2008) 1718–1731.
[15] A.S. Britto, R. Sabourin, L.S.O. Oliveira, Dynamic selection of classifiers – a comprehensive review, Pattern Recognition 47 (11) (2014) 3665–3680.
[16] J. Mendes-Moreira, C. Soares, A.M. Jorge, J.F. D. Sousa, Ensemble approaches for regression: A survey, ACM Computing Surveys 45 (1) (2012) 10:1–10:40.
[17] A. Verikas, A. Lipnickas, K. Malmqvist, M. Bacauskiene, A. Gelzinis, Soft combination of neural classifiers: A comparative study, Pattern Recogn. Lett. 20 (4) (1999) 429–444.
[18] C.J. Merz, Dynamical selection of learning algorithms, in: International Workshop on Artificial Intelligence and Statistics, Springer, 1996.
[19] N. Rooney, D. Patterson, S. Anand, A. Tsymbal, Dynamic integration of regression models, Proceedings of the International Workshop on Multiple Classifier Systems 3077 (2004) 164–173.
[20] J. Mendes-Moreira, A.M. Jorge, C. Soares, J.F.D. Sousa, Ensemble learning: A study on different variants of the dynamic selection approach, in: Machine Learning and Data Mining in Pattern Recognition, vol. 5632, 2009, pp. 191–205.
[21] R.M.C. Cruz, R. Sabourin, G.D. Cavalcanti, T.I. Ren, Meta-des: A dynamic ensemble selection framework using meta-learning, Pattern Recognition 48 (5) (2015) 1925–1935.
[22] R.M. Cruz, R. Sabourin, G.D. Cavalcanti, Meta-des.h: A dynamic ensemble selection technique using meta-learning and a dynamic weighting approach, in, in: Proceedings of the International Joint Conference on Neural Networks, 2016, pp. 216–221.
[23] R.M. Cruz, R. Sabourin, G.D. Cavalcanti, Meta-des.oracle: Meta-learning and feature selection for dynamic ensemble selection, Information Fusion 38 (2017) 84–103.
[24] T.J.M. Moura, G.D.C. Cavalcanti, L.S. Oliveira, Evaluating competence measures for dynamic regressor selection, in: International Joint Conference on Neural Networks – IJCNN, 2019.
[25] N. Rooney, D. Patterson, A weighted combination of stacking and dynamic integration, Pattern Recognition 40 (4) (2007) 1385–1388.
[26] A.T. Sergio, T.P.F. de Lima, T.B. Ludermir, Dynamic selection of forecast combiners, Neurocomputing 218 (2016) 37–50.
[27] J. Mendes-Moreira, A.M. Jorge, J.F. de Sousa, C. Soares, Improving the accuracy of long-term travel time prediction using heterogeneous ensembles, Neurocomputing 150 (2015) 428–439.
[28] D.H. Wolpert, Stacked generalization, Neural Networks 5 (2) (1992) 241–259.
[29] C.M. Bishop, Pattern Recognition and Machine Learning, vol. 19, Springer, 2006.
[30] L. Breiman, J.H. Friedman, R.A. Olshen, C.J. Stone, Classification and Regression Trees, vol. 19, 1984.
[31] L. Breiman, Bagging predictors, Machine Learning 24 (2) (1996) 123–140.
[32] V. Tresp, M. Taniguchi, Combining estimators using non-constant weighting functions, in, in: Proceedings of the 7th International Conference on Neural Information Processing Systems, 1995, pp. 419–426.
[33] R. Adhikari, A neural network based linear ensemble framework for time series forecasting, Neurocomputing 157 (2015) 231–242.
[34] A.E. Eiben, J.E. Smith, Introduction to Evolutionary Computing, Springer, 2003.
[35] H. Kaya, P.Tüfekci, S.F.Gürgen, Local and global learning methods for predicting power of a combined gas and steam turbine, International Conference on Emerging Trends in Computer and Electronics Engineering – ICETCEE 60 (2012) 13–18.
[36] P.Tüfekci, Prediction of full load electrical power output of a base load operated combined cycle power plant using machine learning methods, International Journal of Electrical Power and Energy Systems 60 (2014) 126–140.
[37] I.-C. Yeh, Modeling of strength of high-performance concrete using artificial neural networks, Cement and Concrete Research 28 (12) (1998) 1797–1808.
[38] J.D. Hirst, R.D. King, M.J.E. Sternberg, Comparison of artificial intelligence methods for modeling pharmaceutical qsars, Applied Artificial Intelligence 9 (2) (1995) 213–233.
[39] J.D. Hirst, R.D. King, M.J.E. Sternberg, Quantitative structure-activity relationships by neural networks and inductive logic programming. II. The inhibition of dihydrofolate reductase by triazines, Journal of Computer-Aided Molecular Design 8 (4) (1994) 421–432.
[40] P. Cortez, A. Cerdeira, F. Almeida, T. Matos, J. Reis, Modeling wine preferences by data mining from physicochemical properties, Decision Support Systems 47 (4) (2009) 547–553.
[41] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[42] S.K. Smit, A.E. Eiben, Parameter tuning of evolutionary algorithms: Generalist vs. specialist, in: Applications of Evolutionary Computation, Springer Berlin Heidelberg, Berlin, Heidelberg, 2010, pp. 542–551.
[43] M. Angelova, T. Pencheva, Tuning genetic algorithm parameters to improve convergence time, International Journal of Chemical Engineering.
[44] S. Makridakis, R.L. Winkler, Averages of forecasts: Some empirical results, Management Science 29 (9) (1983) 987–996.