Received October 4, 2021, accepted October 21, 2021, date of publication October 26, 2021, date of current version November 2, 2021.

Digital Object Identifier 10.1109/ACCESS.2021.3123151

On the Regressand Noise Problem: Model Robustness and Synergy With Regression-Adapted Noise Filters

JUAN MARTÍN[1], **JOSÉ A. SÁEZ**[2], **AND EMILIO CORCHADO**[1]
[1]Department of Computer Science and Automatics, University of Salamanca, 37008 Salamanca, Spain
[2]Department of Statistics and Operational Research, University of Granada, 18071 Granada, Spain

Corresponding author: Juan Martín (juanmartin@usal.es)

ABSTRACT This research focuses on analyzing the robustness of different regression paradigms under regressand noise, which has not been examined in depth in the specialized literature. Furthermore, their synergy with fourteen noise preprocessing techniques adapted from the field of classification, known as noise filters, is studied. In order to do this, several noise levels are injected into the output variable of 20 real-world datasets. They are used to evaluate the performance of each regression algorithm with and without the employment of noise filters. The results obtained allow building a robustness ranking of the regression methods to regressand noise. This provides interesting findings, such as some learning paradigms change their well-know behavior with noise in classification problems when they are applied to regression data. On the other hand, the usage of noise filters improves the performance of regression methods, showing different synergies depending on the regression paradigm and filter employed.

INDEX TERMS Filtering, noisy data, regressand noise, regression, robustness.

I. INTRODUCTION

Real-world data often present errors or imperfections caused by device deficiencies, measurement tool limitations or data transcription failures [1], [2]. Fighting these issues can represent more than half the time required to develop a data mining project [3], [4]. In fact, one of the main problems in data analysis is *noise* [5], [6], which is defined as corruptions altering the values in a dataset. Noise can affect both classification [7], [8] and regression [9], [10] data causing numerous problems, such as reduction of performance in the models created and increase in their building time and complexity [5], [11].

In classification, in which the output variable is a discrete class label, noise refers to any disturbance that obscures the relationship between the attributes of a sample and its class [12]. Thus, two types of noise are distinguished: attribute noise [13], which refers to corrupted values in one or more attributes, and class noise [14], which is produced when the samples are incorrectly labeled. Among them, noise in the output variable is usually the most harmful to build models [5], [15]. Significative research efforts have been made to reduce the negative impacts of class noise, motivating the analysis of *robustness* of classification methods with noisy data [16], [17]. For example, it is well-known that `C4.5` [18] is robust due to its pruning scheme, whereas *Nearest Neighbor* (`NN`) [19] is considered sensitive to noise. In addition, plenty of works have been proposed to improve data quality by applying noise preprocessing techniques [12]. One of the main alternatives is *noise filtering* [20], which aims at removing samples with noise in the output variable.

Similar to class noise in classification, *regressand noise* in regression can be defined as imperfections affecting the output variable. However, the identification of regressand noise is more complex than that of class noise. In classification, boundaries among classes must be determined to detect potential noisy samples [21], which are usually identified as those samples falling outside the decision limits of their class. Furthermore, since the amount of labels for a given sample are limited, errors can be identified easier. Because of this, contrary to classification [12], the problem of noisy data has not been addressed in depth in the field of regression. Research on the robustness of different regression methods with several regressand noise levels is scarce in the specialized literature. Furthermore, there are few noise filtering alternatives in regression. The main proposal is that of Kordos and Blachnik [22], which adapted two well-known filters

The associate editor coordinating the review of this manuscript and approving it for publication was Yiqi Liu.

from the field of classification (*Edited Nearest Neighbors* (ENN) and *Condensed Nearest Neighbors* (CNN) [23]) to regression problems.

This paper aims to strengthen the research in this area, analyzing the robustness of different regression paradigms under regressand noise. The experimentation carried out considers 20 real-world datasets, into which several regressand noise levels have been injected (from 5% to 30%, by increments of 5%). These datasets have been used to create regression models with 5 algorithms of a different nature, such as *Recursive Partitioning and Regression Trees* (RPART) [24], NN [19], *Support Vector Machine* (SVM) [25], *Extreme Learning Machine* (ELM) [26] and *Extreme Gradient Boosting* (XGBoost) [27]. Based on the work of [22], 14 noise filters (most of which have not been previously tested in regression problems) are designed to evaluate their synergy with these regression methods. The performance of the regression algorithms with or without noise filtering has been analyzed using the appropriate statistical tests [28]. Full results and details of the experiments are available in the webpage associated with this research at https://juanmartinsantos.github.io/regressandnoise/.

Thus, the main contributions of this work are:

1) *Analysis of robustness to regressand noise of different regression paradigms.* The impact of several noise levels in the prediction performance of regression methods of a different noise sensitiveness is analyzed.
2) *Adaptation of noise filters taken from the classification literature to regression problems.* A wide variety of filtering techniques, most of which have not been previously considered in regression problems, are designed to deal with noisy regression data.
3) *Examination of the suitability of noise filters in regression problems.* The improvement in performance of noise filters with respect to not preprocessing in problems with regressand noise is examined.
4) *Study of synergies between noise filters and regression methods.* The most and less recommended noise filters depending on the regression method used are analyzed in a thorough experimental study.

The remaining of this research is organized as follows. Section II focuses on the main alternatives to deal with noisy data in supervised learning. Section III details the foundations for noise filtering in regression, as well as how noise filters are adapted from the field of classification. Section IV describes the experimental framework. Section V focuses on analyzing the robustness of regression techniques to noise in absence of preprocessing. Section VI analyzes the results of regression noise filters. Finally, Section VII concludes this work and offers ideas about future research.

II. ADDRESSING NOISE IN SUPERVISED LEARNING

In the specialized literature, there exist two main alternatives to deal with the problems caused by noise [12]:

- *Algorithm level approaches* [18], [29]. These methods, known as robust learners, are characterized by being less influenced by noise.
- *Data level approaches* [23], [30]. They preprocess the dataset using noise filters to remove noisy samples.

This section introduces both approaches. Section II-A focuses on the robustness of learning algorithms, whereas Section II-B describes previous research on noise filtering.

A. ROBUSTNESS OF LEARNING ALGORITHMS

Robustness refers to the capability of algorithms to build models that are tolerant to noisy data [31]. This fact implies that their performance is less affected by errors. In classification, there are a large number of works studying the robustness of algorithms [12], [16], [17]. Thus, C4.5 [18] is considered robust due to its pruning strategies to reduce the overfitting to noisy samples [32]. *Random Forest* (RF) [33] counteracts the presence of noise using a random selection of features to split each node. Other techniques, such as XGBoost [27], have shown a better performance than other traditional learning algorithms dealing with classification problems with noisy data [34], [35].

On the other hand, there are classification methods that are considered sensitive to noise [19], [25]. For example, the robustness of k-NN [19] depends on the value of k: the decision model can be easily altered by individual samples if one nearest neighbor is considered, whereas higher values of k make the model more robust [16]. SVM [25] is usually accurate enough when data are relatively clean, whereas its performance is quite deteriorated when considering increased noise levels since the hyperplane found may be affected by samples with errors [17], [36].

In the regression literature, there are fewer studies on the robustness of learning algorithms. Most of them are related to SVM [37]–[39]. Cherkassky and Ma [37] studied the selection of parameters for SVM dealing with data with Gaussian noise. Suykens *et al.* [38] analyzed the improvement in regression performance with noisy data by applying a weighted version of *Least Squares Support Vector Machine* (LS-SVM) [40], whereas Yang *et al.* [41] proposed a robust alternative to LS-SVM to reduce the effect of noise.

This research aims to deepen the understanding of the behavior of different regression methods dealing with data with regressand noise, as well as checking if the findings known in the literature on noisy classification data are maintained in regression problems. Note that noise sensitive techniques, either classification or regression methods, need the usage of preprocessing approaches to reduce the negative impacts of noise. These are briefly described in the next section.

B. NOISE FILTERING TECHNIQUES

In classification, noise filtering consists of removing samples with class noise from the training dataset [21], [30]. Noise filters can be used with datasets regardless of the characteristics of noisy data distributions and, thus, they can deal with both symmetric and asymmetric noise [42]. The separation of noise detection and learning has the advantage that noisy samples do not influence the model building [11]. The removal of such samples reduces the size of the original dataset by selecting relevant data, which improves the performance of the models learned later [11], [43].

There are two main types of noise filters [21], [23]: *similarity*-based filters (which are those based on k-`NN` and distances among samples) and *ensemble*-based filters (which are those based on the predictions of several classification models).

One of the most well-know *similarity* filters is `ENN` [23]. It removes a sample if its class label is different from that of the majority of its nearest neighbors. Based on `ENN`, *All-k Edited Nearest Neighbors* (`AENN`) [44] applies `ENN` from 1 to k and removes those noisy samples considered by any `ENN`. A different strategy is followed by *Blame Based Noise Reduction* (`BBNR`) [45], which removes a sample if it participates in the misclassification of another sample and if its removal does not produce the misclassification on another correctly classified sample. Other *similarity* noise filters, such as `CNN` [23], perform a first classification with `NN` and store the samples that are misclassified. Then, these samples are taken as the training set. The process stops when all the unstored samples are correctly classified. *Reduced Nearest Neighbors* (`RNN`) [46] is an enhancement of `CNN` that includes one more step, which removes samples in the training set that do not affect the performance of k-`NN`.

Ensemble-based filters build several classifiers from the training data and then, based on a voting scheme, decide which samples should be removed. There exist two main voting schemes: *i) consensus* (a sample is removed if it is misclassified by all the classifiers) and, *ii) majority* (a sample is removed if it is misclassified by more than a half of the classifiers). The main representative within this field is *Ensemble Filter* (`EF`) [21]. It divides the training set into μ subsets. Then, a prediction is obtained for each one of the classifiers `C4.5`, `NN` and `LDA`. Finally, noisy samples are removed using a voting scheme. Similarly to `EF`, *Cross-Validation Commitees Filter* (`CVCF`) [47] divides the training set into μ folds and builds a decision tree with `C4.5` on each fold. Using each classifier, a prediction of the whole dataset is obtained. Finally, a sample is considered as noisy using one of the two aforementioned voting schemes. *Iterative-Partitioning Filter* (`IPF`) [30] builds a classifier with `C4.5` on each fold to evaluate the whole dataset. In contrast to `CVCF`, `IPF` integrates an iterative process that removes noisy samples until the number of removed samples is below a threshold.

Along with the aforementioned filters, there are other numerous proposals for noise filtering in classification [48], [49]. Nevertheless, the number of research studies on noise filtering in regression problems, which is discussed in the next section, is considerably smaller than that in classification.

III. DESIGN OF NOISE FILTERS FOR REGRESSION

This section focuses on how noise filtering can be addressed in regression problems. Section III-A explains the basics of regression noise filtering. Then, Section III-B details the output similarity function to build the regression noise filters from those traditionally used in classification. Finally, Section III-C describes specific replacements for classification techniques to make traditional noise filters appropriate for regression tasks.

A. BASICS OF REGRESSION NOISE FILTERING

The first issue to be addressed on noise filtering in regression is defining what this process consists of. Establishing a parallelism with noise filtering in classification, regression noise filtering will be in charge of detecting and eliminating samples with regressand noise in a dataset. However, different from classification (in which class noise is usually identified as misclassifications), regressand noise can occur to varying degrees: the error in the output variable of a sample can be small or more pronounced. For this reason, it must be determined which type of noise should be addressed. In order to do this, it is necessary to define a *threshold* [22] indicating the minimum error allowed to consider that a sample with regressand noise must be treated.

Second, it is important to analyze the design of most of existing classification noise filters. Regardless the filtering mechanism, most of them are based on the comparison of a prediction versus the actual output for each sample to detect the presence of noise. Based on such comparison, the samples are identified as clean ones (if the predicted and the actual outputs are equal) or noisy ones (if they are different). However, this comparison provides an inequality in most of the regression predictions due to the continuous nature of the output variable. In this context, the concept of threshold helps to determine whether two outputs should be considered equal (similar enough) or not.

Under the aforementioned premises, there is one main approach that proposed the adaptation of `ENN` and `CNN` from classification to regression problems [22]. In order to determine if a sample is noisy with these noise filters, the following testing (Equation 1) is performed for each sample φ_j in the dataset once its nearest neighbors $\varphi_{\eta_1}, \ldots, \varphi_{\eta_k}$ are computed:

$$\begin{aligned} &\textit{if} \ \ |\varphi_{0,j} - \textit{mean}(\varphi_{0,\eta_1}, \ldots, \varphi_{0,\eta_k})| > \\ &\alpha \cdot \textit{sd}(\varphi_{0,\eta_1}, \ldots, \varphi_{0,\eta_k}) \ \Rightarrow x \ \ \textit{is noisy}, \end{aligned} \tag{1}$$

where $\varphi_{0,j}$ is the output of the sample φ_j, $\varphi_{0,\eta_1}, \ldots, \varphi_{0,\eta_k}$ are the outputs of the k nearest neighbors of φ_j and α is a user parameter. Thus, there are two factors affecting the decision on if φ_j is noisy: the α parameter and the dispersion of the outputs of the nearest neighbors of φ_j. Both compose the threshold to determine the samples with regressand noise to filter.

Even though this approach provides good results [22], it presents the limitation that the threshold could be hard to interpret. α can take any positive real value, which adds complexity when determining an appropriate value for it.

TABLE 1. Parameter setup for the noise filters.

Similarity filters		
Filter	**Ref.**	**Parameters**
AENN	[44]	$k = 5$; distance: *Euclidean*
BBNR	[45]	$k = 3$; distance: *Euclidean*
CNDC	[49]	$k = 10$; distance: *Euclidean*
CNN	[23]	$k = 1$; distance: *Euclidean*
ENN	[23]	$k = 3$; distance: *Euclidean*
GE	[50]	$k = 5$; distance: *Euclidean*
RNN	[46]	$k = 1$; distance: *Euclidean*

Ensemble filters		
Filter	**Ref.**	**Parameters**
CVCF	[47]	Voting: *majority*; $\mu = 10$
DF	[43]	Voting: *majority*; $\mu = 10$; models = 3
EF	[21]	Voting: *majority*; $\mu = 4$
FMF	[48]	Voting: *majority*; models = 3
HRRF	[51]	Voting: *majority*; models = 4
IPF	[30]	Voting: *majority*; $\mu = 5$; $p = 0.01$; $s = 3$; $y = 0.5$
IRF	[52]	-

In addition, it weights the dispersion of the outputs of the nearest neighbors of the sample, acting as the final threshold. Since the dispersion of the neighborhood is unpredictable, this chained relationship may obscure the consequences of choosing an α value or another. Finally, the work of [22] focused on the adaptation of two similarity filters (ENN and CNN). Section III-B proposes a way to simplify the threshold function, increasing its interpretability and adapting it for being used with most of the classification noise filters.

B. OUTPUT SIMILARITY FUNCTION FOR REGRESSION NOISE FILTERS

This research proposes to adapt the classification filters described in Section II-B, along with other representative ones, up to a total of 14. They are shown in Table 1 along with their main parameters, which are the default ones recommended by the authors of the original noise filters. Note that, even though the CNDC filter [49] has characteristics of both *similarity* and *ensemble* filters, it has been included within *similarity* filters in Table 1 for simplicity.

In order to adapt these noise filters, the approach of Kordos and Blachnik [22] will be applied introducing some modifications. In order to equalize the importance of each variable when computing distances among samples in *similarity* filters and ease the interpretability of the threshold, all the attribute values $\varphi_{i,j}$ (input and output variables) are normalized to the interval [0, 1]. Then, the testing proposed in Equation 1 to check if a sample is noisy is replaced by an *output similarity function* (Equation 2) that allows comparing two regression outputs to determine whether they are similar:

$$similar(\varphi_0^r, \varphi_0^p) = \begin{cases} true, & \text{if } |\varphi_0^r - \varphi_0^p| \le \tau \\ false, & \text{otherwise} \end{cases} \quad (2)$$

where φ_0^r and φ_0^p are two continuous output values and $\tau \in [0, 1]$ is the *noise filtering threshold*, which defines the maximum difference between the outputs to consider them as similar. Note that this function must be incorporated in a regression noise filter when its associated classification filter compares the predicted label against the actual label of a sample.

Since the output values are normalized to [0, 1], $|\varphi_0^r - \varphi_0^p|$ is always in [0, 1]. This fact allows establishing the domain of the threshold $\tau \in [0, 1]$. A value of $\tau = 0$ implies that the actual output $\varphi_0^r = \varphi_{0,j}$ of a sample δ_j and its prediction φ_0^p by the regression noise filter must be exactly equal to be considered as similar and, thus, the corresponding sample δ_j is treated as clean. A value of $\tau = 1$ means that the actual output φ_0^r and the predicted output φ_0^p are always considered as similar and, therefore, all the samples in the training dataset are treated as clean and not removed by the noise filter. Other values of τ imply that the difference between the actual φ_0^r and the predicted φ_0^p outputs of the sample δ_j must be below the τ percent of the domain of the output variable φ_0 to define δ_j as clean, otherwise δ_j is treated as noisy. Therefore, lower values of τ produce a more aggressive filtering (removing more noisy samples), whereas higher values of τ lead to a more conservative filtering (maintaining more samples in the dataset). In this research, all the filters consider a value $\tau = 0.2$, which is that generally providing the best performance results in our experiments –Section VI-E can be consulted for a deeper analysis on the impact of the threshold in the performance.

C. REPLACEMENTS OF CLASSIFICATION ALGORITHMS IN REGRESSION NOISE FILTERS

In order to adapt classification noise filters to regression problems, modifications in some filters are necessary because their original design is based on the usage of classification algorithms. For example, *ensemble*-based filters use a wide variety of classifiers [21], [43], which must be replaced by regression methods. The replacements proposed in this research have been chosen with the aim of not altering the nature of the original noise filters excessively. Thus, the main changes of algorithms to be considered are as follows:

- *Regression k-NN instead of classification k-NN*. All the *similarity* filters (ENN, AENN, GE, BBNR, CNDC, CNN and RNN) and *ensemble* filters (DF, EF, FMF and HRRF) are adapted to use the k-NN version for regression problems. The main difference with respect to the classification version of k-NN is that, instead of considering the majority class from the nearest neighbors to determine the label of a sample, the average of the outputs of its nearest neighbors is considered.
- *Regression versions of* SVM, RF *and Multiple-Layer Perceptron Neural Network* (MLPNN) [53] *instead of classification versions.* The noise filters that use SVM and MLPNN (DF, FMF and HRRF) and the noise filter that uses RF (DF) are adapted to employ the regression

TABLE 2. Datasets used in the experimentation.

Datasets	#at	#sa	Datasets	#at	#sa
abalone	9	4177	forest	12	517
anacalt	7	4052	heart	14	3656
built	108	372	house	17	22784
california	9	20640	mortgage	16	1049
compactiv	22	8192	pole	27	14998
concrete	9	1030	puma32h	33	8192
deltaail	6	7129	stock	10	950
deltaelv	7	9517	treasury	16	1049
elevetors	17	16599	wankara	10	1609
energy	23	17515	wizmir	10	1461

version of such algorithms that is commonly used in the literature.

- `RPART` *instead of* `C4.5`. Those filtering methods based on `C4.5` (`EF`, `CVCF`, `IPF`, `IRF` and `DF`) replace it with the regression method `RPART`, which is also based on decision trees.
- *Linear Regression* (`LR`) *instead of* `LDA` *and Naive Bayes* (`NB`) [54]. `LDA` (used in `EF`) and `NB` (used in `DF`) are replaced by `LR`. `LR` is a simple yet effective regression method which helps to complement the rest of regression paradigms used in these ensemble filters [21], [43].

IV. EXPERIMENTAL FRAMEWORK

This section presents the details of the experimental framework. Section IV-A describes the regression datasets used, whereas Section IV-B explains the methodology for the analysis of the results. Then, Section IV-C presents the computational efficiency associated to the experiments carried out.

A. REAL-WORLD DATASETS

The experimentation considers 20 regression datasets of different cardinalities taken from the *UCI machine learning* and *KEEL-dataset* repositories.[1] They are shown in Table 2, along with the number of attributes (*#at*) and samples (*#sa*) for each one. Constant attributes and samples with missing values are removed from these datasets before their usage. Both the input attributes and the output variable are normalized to the interval [0, 1]. The experiments are executed under macOS Big Sur 11.6 using a dual-core machine (Intel Core i7, 3.5 GHz, 16 GB RAM).

In order to induce regressand noise in each dataset, a scheme based on a well-known approach to inject class noise in classification problems is used [16]. To introduce a noise level $x\%$ into a regression dataset, $x\%$ of the samples are randomly chosen and their output values are replaced by other random ones within the corresponding domain. The noise levels range from $x = 5\%$ to $x = 30\%$, by increments of 5%. As a consequence, 120 noisy datasets with regressand noise are created as follows:

1) In a copy of the original dataset, the desired noise level $x\%$ is injected into the output variable.
2) The original dataset and the noisy copy are partitioned into 5 equivalent folds, that is, each fold contains the same samples in both datasets.
3) The training partitions are built from the noisy copy, whereas the test partitions are built with the samples from the original dataset.

In order to estimate the performance of each regression method, 5 runs of a 5-fold *cross-validation* are considered. This validation scheme is used in other related works [32], [55] and enables to obtain robust performance results. The 120 noisy datasets are processed with 14 noise filters adapted to regression, resulting in a total of 1800 datasets. Each dataset is then trained with one of 5 different regression methods: `RPART` [24], `NN` [19], `SVM` [25], `ELM` [26] and `XGBoost` [27]. All the results obtained from these executions can be found in the webpage associated with this research.[2]

TABLE 3. Parameter setup for the regression methods.

Method	Ref.	Parameters
RPART	[24]	complexity = 0.01; samples per node = 20; method: anova
NN	[19]	distance: *Euclidean*
SVM	[25]	kernel: *radial basis*; tolerance = 0.001; ε = 0.1
ELM	[26]	hidden neurons = 50; activation: *satlins*; weights: $U(-1, 1)$
XGBoost	[27]	trees = 150; learning rate = 0.3; depth = 6; min. weight = 1

B. METHODOLOGY OF ANALYSIS

Both the robustness of the `RPART`, `NN`, `SVM`, `ELM` and `XGBoost` regression methods and the efficacy of the 14 filters in Table 1 are analyzed based on the regression accuracy of the models in a thorough experimental study. Table 3 shows the parameter setup for these regression algorithms. Due to the large amount of results obtained, just the averaged results of each method for the 20 datasets, noise level and noise filter are shown in this paper, but it must be noted that the conclusions are based on the appropriate statistical tests [28], which consider all the results.

The analysis of results is divided into 6 different parts, each one described in a separate section:

1) *Analysis of robustness of regression methods to regressand noise* (Section V). This section analyzes how several regression methods behave with different noise levels in the datasets and compares the results obtained to related findings found in the field of classification.
2) *Comparison of approaches to build regression filters from classification ones* (Section VI-A). This section compares the main approaches to create regression noise filters and remove erroneous samples in the datasets.
3) *Advantages of noise filtering against not preprocessing in regression problems* (Section VI-B). This section analyzes whether the application of noise filtering techniques implies an improvement with respect to the absence of preprocessing.

[1] http://archive.ics.uci.edu/ml; http://www.keel.es/

[2] https://juanmartinsantos.github.io/regressandnoise/

TABLE 4. Computational costs associated to experiments.

Similarity filters		Ensemble filters	
Method	Complexity	Method	Complexity
AENN	$\mathcal{O}(n^2)$	CVCF	$\mathcal{O}(n \cdot log_2\ n)$
BBNR	$\mathcal{O}(n^3)$	DF	$\mathcal{O}(n^4)$
CNDC	$\mathcal{O}(n^4)$	EF	$\mathcal{O}(n^2)$
CNN	$\mathcal{O}(n^3)$	FMF	$\mathcal{O}(n^4)$
ENN	$\mathcal{O}(n^2)$	HRRF	$\mathcal{O}(n^4)$
GE	$\mathcal{O}(n^2)$	IPF	$\mathcal{O}(n \cdot log_2\ n)$
RNN	$\mathcal{O}(n^3)$	IRF	$\mathcal{O}(n \cdot log_2\ n)$

4) *Synergies of regression methods and noise filters under regressand noise* (Section VI-C). This section examines which are the best filters to preprocess noisy data with respect to the regression method considered.
5) *Relationship between the properties of datasets and regression noise filters* (Section VI-D). In this section, the behavior of the regression noise filters has been analyzed according to the size of the datasets, considering independent studies for the number of samples and the number of attributes.
6) *Influence of the noise filtering threshold on the regression performance* (Section VI-E). The motivation of this section is to identify the most suitable thresholds to improve the performance of noise filtering techniques in regression problems.

The performance of the methods is evaluated using the RMSE metric. The *Aligned Friedman* test [28] is applied for multiple comparisons (Sections V, VI-C, VI-D and VI-E). This test is used to compute the set of ranks that represent the effectiveness associated with each method and the p-value (p_{AF}) of significance of the differences found. In addition, the adjusted p-value (p_{Fin}) with the *Finner* procedure is computed. *Wilcoxon*'s test [28] is applied for pairwise comparisons (Sections VI-A and VI-B). The associated p-values p_{Wil} are obtained for these comparisons. This research considers a difference to be significant if the p-value obtained is lower than 0.05.

C. COMPUTATIONAL EFFICIENCY OF EXPERIMENTS

Table 4 shows an estimation of the computational cost for each of the noise filters. The efficiency of similarity filters is mainly determined by the times that distances between samples are computed [56], whereas the efficiency of ensemble filters is mainly influenced by the highest cost of the algorithms used within the ensemble.

The computational cost associated with the whole experiment carried out is determined by the number of noise levels l, the number of datasets d, the number of folds f, the number of noise filters p and the number of regression methods r, as well as the highest cost among noise filters $\mathcal{O}(P)$ and regression methods $\mathcal{O}(R)$. This implies a total cost for the experimentation of $\mathcal{O}(l \cdot d \cdot f \cdot p \cdot r \cdot P \cdot R)$.

V. ANALYSIS OF ROBUSTNESS OF REGRESSION METHODS TO REGRESSAND NOISE

This section focuses on the analysis of the results obtained by different regression techniques with regressand noise in absence of preprocessing. Table 5 presents the RMSE performance results for each noise level and regression method considered. In addition, the number of datasets in which each technique obtains the best result is shown. The analysis of results of this table reveals the following observations:

A. ROBUSTNESS OF REGRESSION METHODS TO NOISE

The behavior of all the regression methods, regardless of their nature, is affected by regressand noise. SVM and RPART are the most robust methods, with very similar results in all the noise levels. They are followed by ELM and XGBoost, with slightly more affected results. Finally, NN obtains the worst RMSE values in all the noise levels.

The good performance results of RPART can be explained by the form of building the model, which is based on decision trees including a pruning mechanism. This approach is able to better avoid the overfitting to noise since those noisy samples that do not follow the general distribution of the dataset are usually not modelled. The results of SVM are equally good. Even though noisy samples may affect the optimization process of SVM and deteriorate its performance, the results in Table 5 show that noise is not affecting the performance of SVM in a high degree. This fact may be related to the usage of its ε-insensitive loss function, which allows dealing with undesired samples, such as noisy samples.

The average performance results of ELM are not as good as those of RPART and SVM. However, it achieves the best result on a remarkable number of datasets. This fact implies that ELM achieves excellent results on some datasets, while it is more deteriorated in other ones. Note that conventional ELM algorithms were designed for the noiseless situation. These techniques are generally better capturing the nuances hidden in the data, providing excellent performance results for datasets in which noise is not deteriorating important parts of the problem domain. Nevertheless, when these nuances are noisy and impact crucial areas of the domain, the performance may be impaired. Something similar occurs with XGBoost, which performs better at low noise levels and deteriorates as the number of noisy samples increases.

Finally, NN provides the worst results since the inclusion of some noisy samples negatively affects the prediction of many others, reducing its performance.

B. PERFORMANCE DETERIORATION AS THE NOISE LEVEL INCREASES

This analysis helps to complete the understanding of the robustness of each regression method. The increase of the noise level impacts the behavior of the methods differently. XGBoost, which obtains relatively good results, presents the largest drop in performance from 5% to 30% of noise, with

TABLE 5. Comparison of robustness to regressand noise of regression methods: RMSE and number of datasets with the best result (the best results are remarked in bold).

Metric	RMSE						Best (out of 20)					
Noise	5%	10%	15%	20%	25%	30%	5%	10%	15%	20%	25%	30%
RPART	0.1251	0.1573	0.1785	0.1989	**0.2145**	**0.2269**	3	3	3	3	5	6
NN	0.1567	0.2043	0.2404	0.2697	0.2889	0.3046	0	0	0	0	0	0
SVM	**0.1180**	**0.1528**	**0.1762**	**0.1977**	0.2163	0.2298	**8**	**7**	**7**	7	4	5
ELM	0.1526	0.1826	0.2011	0.2198	0.2322	0.2550	3	5	**7**	**8**	**9**	**8**
XGBoost	0.1195	0.1592	0.1863	0.2077	0.2277	0.2392	6	5	3	2	2	1

a deterioration of 100.17%. RPART (whose performance is deteriorated by a 81.37%) and ELM (with a deterioration of 67.1%) are the least affected methods by increasing the noise level. It is important to note that NN, which obtains the worst results, and SVM, which obtains the best results, show a very similar drop in performance (around 94.5%) from the lowest to highest noise level.

Generally, the higher the performance of a learning method at the lowest noise level, the greater the margin for this performance to deteriorate as the noise level increases. However, this is not always fulfilled in the results of Table 5, which leads to interesting observations. Methods with good results, such as RPART, are able to maintain the performance through the different noise levels better than other techniques with worse performance results, such as NN. This fact reflects a good initial performance of these methods and a high robustness to increase the noise level in the data.

C. BEST REGRESSION METHOD IN INDIVIDUAL DATASETS

SVM achieves the best performance in 7-8 datasets at 5-10% of noise. Then, SVM and ELM obtain the best results in 7 datasets at 15% of noise. From that noise level onwards, ELM is the best in 8-9 datasets in each noise level. These results show that SVM provides a good prediction accuracy with noisy regression data and it also highlights in the results of individual datasets. On the other hand, even though ELM offers the best performance in some datasets at concrete noise levels, its average RMSE is not as good as that of SVM. As it was previously mentioned, this fact indicates that ELM offers less stable results among the different datasets.

XGBoost obtains good performances in all the noise levels but, as the noise level increases, the number of datasets in which it obtains the best result decreases. This fact, along with its aforementioned performance deterioration, indicates a certain sensibility of XGBoost to increase the noise level. It is also important to note that RPART increases the number of datasets with the best result along with the amount of noise, which supports the idea of its robustness to high noise levels. Finally, NN is not the best in any dataset and noise level, as it was expected for its worse performance results.

D. COMPARISON BETWEEN NOISE ROBUSTNESS OF REGRESSION AND CLASSIFICATION METHODS

The comparison of the above results for noisy regression datasets with other ones found in the classification literature [12], [17], [35] leads to interesting observations.

C4.5 [18] is known to be a robust method (due to its pruning mechanism), which usually provides good performance results when working with noisy data. In experiments with regressand noise, RPART (which is based on building decision trees as C4.5) is one of the methods providing the best results, as it also integrates pruning strategies that increase its robustness against noise. On the other hand, NN is usually considered as a very noise sensitive learner in the field of classification [16], [19]. This statement is also valid in our experiments, since NN performs worse than the other methods.

Difference from the previous cases, SVM offers a distinct behavior with noisy regression data than that of its classic well-known behavior with noisy classification data. In classification, SVM is usually viewed as a noise sensitive method [16], [17]. The experiments of this research with regressand noise show that SVM is able to be more noise tolerant with independence of the noise level maintaining, even at the highest noise levels, one of the best performances among all the regression methods. This fact is mainly due to the different working of SVM dealing with classification and regression problems. In classification, the decision boundary between classes is determined by a hyperplane. It is obtained maximizing its separation to the closest samples of each class (*support vectors*) [25]. If any of the support vectors is affected by noise, the position and orientation of the hyperplane can change, reducing the performance of SVM. In regression, the hyperplane is obtained adjusting the data with a specific function. Then, the decision boundaries, which are placed at a distance ε of the hyperplane, are defined. The decision boundaries carry, in this case, a noise robustness mechanism, since ε can be viewed as the maximum error allowed to penalize samples (errors lower than ε are not relevant). Thus, if noise affects a sample (within the decision boundaries) with an error lower than ε or if a sample (outside the decision boundaries) is corrupted and enters within the decision boundaries, the hyperplane is little affected. On the other hand, samples that go outside the decision boundaries are penalized to build the model. These mechanisms explain the good performance of SVM with noisy regression data.

Finally, it should be noted that XGBoost and ELM, despite being the most advanced techniques tested, do not offer the best results with noisy regression data.

In order to complete the above analysis, Table 6 shows the ranks obtained by the *Aligned Friedman* test for all the regression methods in each noise level. In addition, it shows

TABLE 6. Comparison of robustness to regressand noise of regression methods: results of the *Aligned Friedman* (*ranks* and *p*-values p_{AF}) and *Finner* test (*p*-values p_{Fin}). The best results are remarked in bold.

Noise	5%	10%	15%	20%	25%	30%
			Ranks			
RPART	46.20	42.30	37.83	36.25	**33.10**	**31.75**
NN	79.95	83.60	88.25	88.50	88.50	87.50
SVM	**28.25**	**25.88**	**27.20**	**29.05**	34.00	35.50
ELM	55.60	50.43	46.23	44.73	41.10	44.55
XGBoost	42.50	50.30	53.00	53.98	55.80	53.20
p_{AF}	**2.39E-06**	**1.74E-07**	**9.65E-09**	**1.19E-08**	**1.23E-08**	**5.26E-08**
			p_{Fin}			
RPART	6.66E-02	7.34E-02	2.47E-01	4.33E-01	-	-
NN	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
SVM	-	-	-	-	9.22E-01	6.83E-01
ELM	**5.74E-03**	**1.48E-02**	5.05E-02	1.15E-01	4.75E-01	2.11E-01
XGBoost	1.20E-01	**1.48E-02**	**9.82E-03**	**1.31E-02**	**2.65E-02**	3.84E-01

the *p*-values corresponding to the *Aligned Friedman* (p_{AF}) and *Finner* (p_{Fin}) tests. The following remarks can be made after the analysis of Table 6:

- *Statistical ranking of the regression methods with noisy data.* SVM obtains the best ranks for the noise levels ranging from 5% to 20%. For the highest noise levels (25% and 30%) the best method is RPART. ELM and XGBoost are placed in the positions 3-4 for all the noise levels. The worst position is obtained by NN with independence of the noise level. The low *p*-values p_{AF} indicate that there are significant differences in all the comparisons performed.
- *Statistical differences among the regression methods.* According to the *Aligned Friedman* test, the control method is either SVM or RPART with independence of the noise level. They are always statistically better against NN, which obtains the highest RMSE values in Table 5. Apart from these differences, the *Finner* test also reveals that SVM statistically improves the performance of ELM at 5-15% of noise and XGBoost at 10-20%. Similar results are obtained for RPART at 25% of noise, achieving significant differences against XGBoost.

These statistical results confirm the conclusions derived from the analysis of Table 5. SVM and RPART are positioned as the best alternatives to work with noisy regression data, whereas NN is clearly the most noise sensitive method and its usage is not recommended with regression datasets suffering from regressand noise.

VI. EFFICACY OF REGRESSION-ADAPTED NOISE FILTERS WITH REGRESSAND NOISE

This section analyzes the results obtained once the datasets are preprocessed with noise filtering methods adapted to regression tasks. First, Section VI-A compares the original approach to create regression noise filters (Section III-A) to the proposal of this research. Section VI-B focuses on analyzing the advantages of applying regression noise filtering against not preprocessing. Section VI-C studies the synergy of noise filters with each regression method, whereas Section VI-D evaluates the relationship between regression-adapted noise filters and the size properties of the datasets. Finally, Section VI-E examines the influence of the noise filtering threshold in the regression performance.

A. COMPARISON OF APPROACHES TO BUILD REGRESSION FILTERS FROM CLASSIFICATION ONES

The RMSE results and *Wilcoxon*'s test *p*-values for all the regression methods considering the noise filters developed with the modified approach of this research (denoted as MOD) and those with the original approach of [22] (denoted as ORI) are shown in the Table 7. The ENN and CNN filters are considered in this comparison since they are the methods originally adapted in [22]. From the analysis of results in Table 7, the following remarks can be made:

1) COMPARISON OF APPROACHES TO CREATE REGRESSION NOISE FILTERS

For all the regression methods using both ENN and CNN, MOD is the best in all the noise levels. Its greatest advantages with respect to ORI are generally in the medium-high noise levels (20-30%). From these results, it is important to highlight the good behavior of NN. This fact is particularly interesting, since NN is the most noise sensitive regression method employed and it is able to better detect the efficacy of noise filtering [11].

MOD is able to better maintain the initial performance of all the regression methods along the different noise levels. For example, the performance of NN using ENN is deteriorated a 51.91% from 5% to 30% with MOD, whereas this performance is reduced a 126.29% with ORI. A similar situation occurs considering CNN. Thus, SVM is affected a 13.24% with MOD, whereas the usage of ORI implies a deterioration of 69.90%. Another interesting case is that of XGBoost, which is affected a 6.37% between the extreme levels with MOD and a 55.33% with ORI.

2) STATISTICAL SIGNIFICANCE OF THE CONCLUSIONS REACHED

The *p*-values p_{Wil} obtained confirm the above results. For ENN, *Wilcoxon*'s test shows statistical differences in favor of MOD in all the noise levels, except in 5% for RPART and 5-10% for ELM, in which no statistical differences are found among the methods. For CNN, *Wilcoxon*'s test confirms MOD as the best approach in all the noise levels and regression methods. These results show that implementation of the *similar* function used in MOD to consider a sample as noisy (Equation 2) offers several advantages over the original function used in ORI (Equation 1). First, as discussed in Section III.B, the similarity function in MOD is more interpretable and easier to employ for the user. On the other hand, as seen in the results in Table 7, it also offers better prediction precision results. This fact indicates that the margin around the regressand value of a sample to be considered safe (that is, the sample does not contain regressand noise) must be established according to the size of the domain, as it is done in MOD. This implies a more stable (the same for all samples) and precise way of determining regressand noise in the samples than if a

TABLE 7. Comparison of the approach of [22] to create regression noise filters (ORI) with that incorporating some modifications to improve its interpretability (MOD) applying the ENN and CNN noise filters (the best results are remarked in bold).

Metric		RMSE						p_{Wil}					
Noise		5%	10%	15%	20%	25%	30%	5%	10%	15%	20%	25%	30%
Approach		ENN											
RPART	MOD	**0.0856**	**0.0868**	**0.0877**	**0.0892**	**0.0910**	**0.0944**	7.79E-01	**4.00E-02**	**7.08E-04**	**1.19E-04**	**4.77E-05**	**2.67E-05**
	ORI	0.0857	0.0901	0.0970	0.1067	0.1166	0.1283						
NN	MOD	**0.0888**	**0.0950**	**0.1037**	**0.1122**	**0.1245**	**0.1349**	**1.05E-04**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
	ORI	0.0989	0.1295	0.1610	0.1856	0.2063	0.2238						
SVM	MOD	**0.0739**	**0.0748**	**0.0755**	**0.0771**	**0.0777**	**0.0795**	**1.91E-06**	**1.34E-05**	**7.08E-04**	**3.22E-04**	**3.95E-04**	**1.68E-04**
	ORI	0.0801	0.0806	0.0810	0.0817	0.0838	0.0873						
ELM	MOD	**0.1117**	**0.1125**	**0.1195**	**0.1168**	**0.1215**	**0.1226**	6.42E-01	1.04E-01	**1.07E-02**	**3.81E-06**	**7.08E-04**	**3.81E-06**
	ORI	0.1131	0.1159	0.1255	0.1362	0.1406	0.1641						
XGBoost	MOD	**0.0591**	**0.0628**	**0.0687**	**0.0746**	**0.0824**	**0.0905**	**4.22E-03**	**9.54E-06**	**3.62E-05**	**5.72E-06**	**5.72E-06**	**3.81E-06**
	ORI	0.0646	0.0819	0.0991	0.1147	0.1309	0.1427						
Approach		CNN											
RPART	MOD	**0.1273**	**0.1384**	**0.1520**	**0.1595**	**0.1695**	**0.1769**	**4.83E-04**	**1.91E-06**	**1.34E-05**	**3.81E-06**	**1.91E-06**	**1.34E-04**
	ORI	0.1604	0.1938	0.2122	0.2283	0.2372	0.2406						
NN	MOD	**0.1591**	**0.1950**	**0.2260**	**0.2472**	**0.2591**	**0.2705**	**1.91E-06**	**3.81E-06**	**1.34E-04**	**1.91E-06**	**3.81E-06**	**2.67E-05**
	ORI	0.2414	0.2883	0.3165	0.3483	0.3577	0.3576						
SVM	MOD	**0.0846**	**0.0836**	**0.0879**	**0.0872**	**0.0898**	**0.0958**	**5.86E-04**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**8.20E-05**
	ORI	0.0990	0.1160	0.1318	0.1468	0.1615	0.1682						
ELM	MOD	**0.1672**	**0.1717**	**0.1784**	**0.1842**	**0.1913**	**0.1992**	**3.81E-06**	**3.81E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
	ORI	0.3183	0.3013	0.5833	0.4908	0.4670	0.5217						
XGBoost	MOD	**0.1372**	**0.1544**	**0.1702**	**0.1804**	**0.1930**	**0.2009**	**5.86E-04**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-05**
	ORI	0.1800	0.2225	0.2448	0.2674	0.2802	0.2796						

margin depending on the output values in the neighborhood of each sample is considered, as it is performed in ORI.

Due to the good results obtained by MOD in these comparisons, the next sections focus on analyzing the characteristics and performance of the regression filters created using it.

B. ADVANTAGES OF NOISE FILTERING AGAINST NOT PREPROCESSING IN REGRESSION PROBLEMS

Table 8 shows the RMSE results of RPART, NN and SVM in datasets with different regressand noise levels without preprocessing (None) and once they are preprocessed with the 14 noise filters of Table 1, which are adapted for regression problems following the procedure described in Sections III-B and III-C. In addition, the p-values (p_{Wil}) corresponding to *Wilcoxon*'s test after comparing each noise filter against None are shown for each noise level. The results of ELM and XGBoost can be found in the webpage associated with this research and provide similar conclusions to those presented here. The analysis of this table of results leads to the following observations:

1) SUITABILITY OF USING NOISE FILTERS WITH NOISY REGRESSION DATA

All the regression methods improve their performance using noise filters against not preprocessing at all the noise levels -with the exceptions of RPART and NN with the CNN and RNN *similarity*-based filters at the lowest noise level. These results show the high potential of all the filtering methods adapted to regression data to overcome the problems produced by errors in the output variable.

2) CAPABILITY TO MAINTAIN THE PERFORMANCE REGARDLESS THE NOISE LEVEL

One of the most important observations from the results in Table 8 is that noise filters are able to better maintain the initial performance of the methods as the noise level increases. For RPART, most of the filters are able to closely maintain the performance of the lowest noise level until reaching the highest noise level. For NN, even though it is considered one of the algorithms most sensitive to noise, some noise filters are also able to maintain the model performance regardless of the injected noise level. For example, IPF and AENN obtain close RMSE values (around 0.09 and 0.1 for 5% and 30% of noise, respectively). Finally, SVM is the method showing the most stable prediction accuracy for all the filters at all the noise levels. These results show the capacity to detect noisy samples of the proposed approach, since noise filters are able to maintain the performance at the different noise levels.

3) ON THE GOOD PERFORMANCE OF SVM WITH REGRESSION FILTERS

Using SVM, CNDC obtains the lowest RMSE (0.0716) at 5% of noise, whereas EF presents the best performance at the highest noise level with an RMSE of 0.0758. The maximum RMSE value is 0.0958 for CNN at 30% of noise. These low and close RMSE values suggest the great performance of all the adapted regression noise filters with SVM. Thus, SVM is the method, among those tested in this experimentation, most robust against noise and that benefiting the most from the usage of regression noise filters.

4) STATISTICAL SIGNIFICANCE OF THE CONCLUSIONS REACHED

For RPART, all the noise filters are statistically better than None (except CNN at 5% and RNN for 5-10% of noise). For NN, noise filters show significant differences in most of the cases as the low p-values p_{Wil} reflect. There are few

TABLE 8. RMSE results for RPART, NN and SVM applying regression noise filters and without preprocessing (the best results are remarked in bold).

Metric	RMSE						p_{Wil}					
Noise	5%	10%	15%	20%	25%	30%	5%	10%	15%	20%	25%	30%
							RPART					
None	0.1251	0.1573	0.1785	0.1989	0.2145	0.2269	-	-	-	-	-	-
AENN	0.0888	0.0912	0.0914	0.0921	0.0959	0.0942	**3.81E-06**	**3.81E-06**	**1.91E-06**	**1.91E-06**	**3.81E-06**	**1.91E-06**
BBNR	0.0859	0.0900	0.0944	0.1006	0.1072	0.1151	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNDC	0.0860	0.0907	0.0985	0.1048	0.1117	0.1202	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNN	0.1273	0.1384	0.1520	0.1595	0.1695	0.1769	6.41E-01	**7.08E-04**	**7.08E-04**	**3.81E-06**	**1.91E-06**	**9.54E-06**
ENN	0.0856	0.0868	0.0877	0.0892	0.0910	0.0944	**3.81E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
GE	0.0885	0.0967	0.1064	0.1162	0.1251	0.1356	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
RNN	0.1276	0.1516	0.1636	0.1708	0.1780	0.1812	1.00E+00	1.89E-01	**1.36E-02**	**1.34E-04**	**1.91E-06**	**1.91E-06**
CVCF	0.0864	0.0867	0.0882	0.0888	0.0908	0.0954	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
DF	0.0852	0.0851	0.0853	0.0865	0.0865	0.0884	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
EF	0.0849	0.0857	0.0860	0.0887	0.0925	0.0945	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
FMF	0.0849	0.0850	0.0862	0.0860	0.0869	0.0881	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
HRRF	0.0888	0.0979	0.1068	0.1171	0.1306	0.1428	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**3.81E-06**	**3.81E-06**
IPF	0.0900	0.0909	0.0898	0.0913	0.0910	0.0927	**2.67E-05**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
IRF	0.0879	0.0881	0.0889	0.0892	0.0906	0.0922	**5.72E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
							NN					
None	0.1567	0.2043	0.2404	0.2697	0.2889	0.3046	-	-	-	-	-	-
AENN	0.0896	0.0933	0.0947	0.0975	0.1045	0.1046	**3.81E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
BBNR	0.1031	0.1222	0.1409	0.1544	0.1767	0.1893	**3.81E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNDC	0.1200	0.1468	0.1722	0.1905	0.2048	0.2175	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNN	0.1591	0.1950	0.2260	0.2472	0.2591	0.2705	1.00E+00	**2.96E-02**	**1.05E-04**	**3.81E-06**	**1.91E-06**	**1.91E-06**
ENN	0.0888	0.0950	0.1037	0.1122	0.1245	0.1349	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
GE	0.1288	0.1597	0.1851	0.2097	0.2265	0.2402	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
RNN	0.1636	0.2038	0.2343	0.2553	0.2669	0.2720	1.00E+00	8.67E-01	1.98E-01	**1.21E-02**	**5.72E-06**	**1.91E-06**
CVCF	0.0884	0.0921	0.0972	0.1019	0.1100	0.1180	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
DF	0.0856	0.0889	0.0928	0.0975	0.1017	0.1095	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
EF	0.0876	0.0969	0.1046	0.1156	0.1279	0.1379	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
FMF	0.0869	0.0902	0.0936	0.0979	0.1039	0.1093	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
HRRF	0.1291	0.1604	0.1881	0.2110	0.2270	0.2411	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
IPF	0.0913	0.0949	0.0964	0.0992	0.1026	0.1063	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
IRF	0.0885	0.0919	0.0943	0.0977	0.1015	0.1055	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
							SVM					
None	0.1180	0.1528	0.1762	0.1977	0.2163	0.2298	-	-	-	-	-	-
AENN	0.0774	0.0783	0.0790	0.0802	0.0841	0.0827	**1.91E-05**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
BBNR	0.0780	0.0777	0.0778	0.0789	0.0788	0.0806	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNDC	0.0716	0.0722	0.0728	0.0744	0.0750	0.0770	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CNN	0.0846	0.0836	0.0879	0.0872	0.0898	0.0958	**1.34E-04**	**3.81E-06**	**5.72E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
ENN	0.0739	0.0748	0.0755	0.0771	0.0777	0.0795	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
GE	0.0719	0.0728	0.0741	0.0762	0.0776	0.0805	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
RNN	0.0831	0.0869	0.0911	0.0909	0.0939	0.0953	**3.62E-05**	**3.81E-06**	**5.72E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
CVCF	0.0739	0.0746	0.0750	0.0762	0.0768	0.0801	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
DF	0.0724	0.0733	0.0736	0.0749	0.0749	0.0766	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
EF	0.0722	0.0728	0.0728	0.0741	0.0743	0.0758	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
FMF	0.0732	0.0743	0.0748	0.0762	0.0759	0.0775	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
HRRF	0.0729	0.0744	0.0754	0.0780	0.0818	0.0873	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
IPF	0.0772	0.0780	0.0781	0.0786	0.0792	0.0797	**3.81E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**
IRF	0.0749	0.0756	0.0761	0.0776	0.0777	0.0809	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**	**1.91E-06**

exceptions, such as CNN and RNN, in which significant differences are not observed up to 5% and 15% of noise, respectively. For SVM, *Wilcoxon*'s test shows significant differences in favor of the noise filters with very low *p*-values in all the comparisons performed. These results allow concluding that the elimination of samples with regressand noise, in a similar way as occurs in the classification when eliminating samples with class noise, is beneficial for the creation of models in terms of their performance.

C. SYNERGIES OF REGRESSION METHODS AND NOISE FILTERS UNDER REGRESSAND NOISE

The left side of Table 9 shows the results of the *Aligned Friedman* test (*ranks*) and the *p*-values of the *Finner* test (p_{Fin}) in the form of *ranks*/p_{Fin}, considering the 14 noise filters for RPART, NN and SVM at all the noise levels. The right side of Table 9 shows the number of datasets in which each noise filter achieves the best result. The results of ELM and

TABLE 9. Comparison of noise filters with each regression method: results of *Aligned Friedman* (*Ranks*), *Finner* (p_{Fin}) and number of datasets in which each method obtains the best results (the best results are remarked in bold).

Metric	Ranks/p_{Fin}						Best (out of 20)					
Noise	5%	10%	15%	20%	25%	30%	5%	10%	15%	20%	25%	30%
				RPART								
AENN	132/3.43E-01	122/2.73E-01	110/3.46E-01	107/3.89E-01	99/5.21E-01	95/5.55E-01	2	3	2	3	**8**	**6**
BBNR	113/6.39E-01	120/2.86E-01	130/8.30E-02	143/**1.94E-02**	154/**3.81E-03**	163/**6.52E-04**	1	1	0	0	0	0
CNDC	118/5.96E-01	139/9.65E-02	166/**1.45E-03**	174/**3.65E-04**	176/**1.81E-04**	180/**4.90E-05**	0	1	0	1	0	0
CNN	258/**5.45E-09**	256/**2.07E-10**	255/**2.62E-11**	253/**3.63E-11**	249/**5.00E-11**	248/**4.01E-11**	0	0	0	0	0	0
ENN	110/7.07E-01	99/6.72E-01	93/6.04E-01	88/7.13E-01	88/6.38E-01	90/5.55E-01	1	2	3	2	3	2
GE	147/2.18E-01	178/**1.12E-03**	200/**5.67E-06**	208/**9.99E-07**	210/**3.43E-07**	212/**1.20E-07**	1	1	1	0	0	0
RNN	257/**5.45E-09**	260/**1.27E-10**	260/**1.60E-11**	256/**2.46E-11**	253/**3.56E-11**	249/**4.01E-11**	0	0	0	0	0	0
CVCF	116/6.04E-01	100/6.72E-01	95/6.04E-01	89/7.10E-01	90/6.38E-01	95/5.55E-01	0	0	1	1	0	0
DF	102/8.90E-01	**86**	**78**	79/9.25E-01	**74**	72/9.64E-01	2	**5**	**6**	**6**	4	4
EF	101/9.04E-01	90/8.94E-01	82/8.63E-01	87/7.13E-01	92/6.38E-01	92/5.55E-01	2	1	3	1	0	0
FMF	**98**	86/9.93E-01	83/8.63E-01	**76**	75/9.60E-01	**71**	**7**	3	1	5	3	4
HRRF	144/2.18E-01	189/**2.27E-04**	205/**2.62E-06**	212/**4.50E-07**	223/**2.88E-08**	223/**1.26E-08**	1	0	0	0	0	0
IPF	140/2.37E-01	131/1.56E-01	107/3.79E-01	102/4.52E-01	91/6.38E-01	89/5.55E-01	1	1	1	1	1	4
IRF	131/3.43E-01	111/4.33E-01	102/4.43E-01	95/5.98E-01	93/6.38E-01	88/5.55E-01	0	2	2	0	1	0
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	-	-	-	-	-	-
				NN								
AENN	88/6.94E-01	78/7.50E-01	75/8.64E-01	72/9.44E-01	70/8.79E-01	**62**	**8**	**9**	**10**	**10**	**10**	**8**
BBNR	153/**3.72E-03**	165/**3.73E-04**	173/**1.07E-04**	175/**8.10E-05**	176/**2.43E-05**	179/**1.06E-05**	0	0	0	0	0	0
CNDC	193/**6.35E-06**	197/**1.34E-06**	199/**9.97E-07**	196/**1.96E-06**	198/**3.88E-07**	201/**1.45E-07**	0	0	0	0	0	0
CNN	249/**4.23E-11**	249/**1.15E-11**	250/**1.18E-11**	249/**1.51E-11**	245/**8.77E-12**	245/**6.34E-12**	0	0	0	0	0	0
ENN	89/6.94E-01	92/5.32E-01	101/3.36E-01	107/2.13E-01	114/7.93E-02	117/**4.92E-02**	1	0	0	0	0	0
GE	216/**6.43E-08**	218/**1.80E-08**	215/**4.37E-08**	217/**2.20E-08**	220/**2.96E-09**	222/**1.29E-09**	0	0	0	0	0	0
RNN	251/**4.23E-11**	254/**5.57E-12**	256/**4.02E-12**	254/**6.90E-12**	251/**3.14E-12**	247/**6.34E-12**	0	0	0	0	0	0
CVCF	89/6.94E-01	84/6.54E-01	82/7.34E-01	80/7.99E-01	82/6.09E-01	83/5.32E-01	1	0	1	1	0	0
DF	**72**	**69**	**69**	**69**	**63**	66/9.04E-01	7	6	4	4	4	4
EF	84/6.94E-01	100/3.83E-01	102/3.36E-01	114/1.41E-01	123/**3.51E-02**	127/**1.96E-02**	0	0	0	0	0	0
FMF	80/7.61E-01	73/8.66E-01	73/8.93E-01	70/9.59E-01	68/8.85E-01	67/8.99E-01	0	1	1	0	1	1
HRRF	219/**4.61E-08**	219/**1.80E-08**	218/**2.36E-08**	220/**1.65E-08**	221/**2.66E-09**	223/**1.29E-09**	0	0	0	0	0	0
IPF	98/5.09E-01	89/5.52E-01	80/7.52E-01	74/9.08E-01	70/8.79E-01	66/9.04E-01	1	3	0	2	1	0
IRF	87/6.94E-01	81/7.06E-01	75/8.64E-01	71/9.49E-01	67/8.87E-01	63/9.56E-01	2	1	4	3	4	7
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	-	-	-	-	-	-
				SVM								
AENN	162/**7.44E-03**	169/**2.71E-03**	155/**1.01E-02**	163/**1.96E-03**	156/**2.56E-03**	143/**1.52E-02**	0	0	1	0	2	3
BBNR	194/**8.75E-05**	186/**1.98E-04**	180/**3.38E-04**	170/**9.47E-04**	158/**2.27E-03**	151/**7.76E-03**	2	1	0	1	2	1
CNDC	**85**	**82**	90/7.18E-01	90/6.15E-01	97/3.67E-01	103/2.78E-01	**5**	**6**	5	**5**	**5**	4
CNN	225/**5.68E-07**	222/**2.82E-07**	229/**3.19E-08**	229/**1.51E-08**	236/**8.13E-10**	239/**5.63E-10**	1	0	0	0	0	0
ENN	130/1.29E-01	127/1.21E-01	125/1.15E-01	131/5.63E-02	129/**3.90E-02**	125/6.56E-02	1	1	0	0	0	0
GE	88/8.99-01	96/6.43E-01	111/2.43E-01	122/1.01E-01	134/**2.70E-02**	143/**1.52E-02**	2	2	1	1	0	0
RNN	217/**1.62E-06**	239/**9.53E-09**	244/**1.81E-09**	245/**4.53E-10**	248/**6.50E-11**	245/**2.54E-10**	1	0	0	0	0	0
CVCF	132/1.20E-01	126/1.21E-01	122/1.33E-01	116/1.43E-01	114/1.28E-01	119/9.39E-02	0	1	1	2	2	1
DF	100/6.31E-01	91/7.55E-01	89/7.18E-01	82/8.19E-01	76/8.50E-01	81/7.49E-01	2	2	2	3	0	1
EF	97/6.60E-01	89/7.78E-01	**79**	**76**	**71**	**73**	4	**6**	**8**	3	**5**	**5**
FMF	114/3.40E-01	116/2.22E-01	115/2.04E-01	109/2.25E-01	95/3.74E-01	97/3.78E-01	1	0	0	1	1	1
HRRF	115/3.40E-01	130/1.11E-01	143/**2.64E-02**	161/**2.36E-03**	185/**4.07E-05**	195/**7.82E-06**	0	0	1	0	0	0
IPF	164/**6.95E-03**	162/**4.27E-03**	151/**1.28E-02**	144/**1.72E-02**	142/**1.21E-02**	130/**4.93E-02**	1	1	0	3	1	3
IRF	143/**4.91E-02**	133/9.46E-02	132/6.94E-02	131/5.63E-02	125/5.28E-02	125/6.56E-02	0	0	1	1	2	1
p_{AF}	**1.09E-14**	**5.55E-16**	**5.55E-16**	**6.66E-16**	**0.00E+00**	**0.00E+00**	-	-	-	-	-	-

XGBoost are available on the website associated with this research.

A first view of Table 9 reveals that regression methods change their behavior with respect to the noise filter used since the ranks of the *Friedman Aligned* test and the p-values of *Finner* vary from one regression method to another. This fact suggests that there are different synergies between the type of regression method and noise filter employed. In the analysis below, note that the *Aligned Friedman* test always provides very low p-values showing significant differences among the noise filters and thus, the *Finner* post-hoc procedure can be safely applied in all the cases. Analyzing the results of Table 9, the following points must be remarked:

1) SYNERGIES BETWEEN RPART AND REGRESSION NOISE FILTERS

- *Principal noise filters.* The results obtained recommend the usage of the FMF and DF *ensemble*-filters with RPART, since both filters alternate good results at different noise levels. The best ranks are obtained by FMF at 5%, 20% and 30% of noise and by DF for the rest of noise levels. The *Finner* test reveals that FMF is statistically better than RNN and CNN at 5% of noise. DF presents significant differences with RNN, CNN, GE and HRRF at 10% of noise, whereas CNDC is included at 15%. Finally, FMF and DF present statistical differences with BBNR, CNDC, CNN, GE, RNN and HRRF for the rest of noise levels.

- *Inferior noise filters.* The less recommendable noise filters for RPART are CNN and RNN (both *similarity*-based). They obtain the worst ranks in all the noise levels. In fact, RNN presents the highest ranks in all the noise levels except at 5%, in which CNN is the worst method.

2) SYNERGIES BETWEEN NN AND REGRESSION NOISE FILTERS

- *Principal noise filters.* In this case, DF is the most recommended to use with NN when the noise level is low-medium, whereas AENN is recommendable when the noise level is high. DF obtains the lowest ranks in all the noise levels except at 30%, in which the best method is AENN. The *Finner* test shows that DF presents statistical differences against six other filters (BBNR, CNDC, CNN, GE, RNN and HRRF) at noise levels 5-20%. On the other hand, AENN shows significant differences against all the aforementioned filters plus ENN at 30% of noise.
- *Inferior noise filters.* In the case of NN, the amount of noise filters to avoid is larger than that in the case of RPART, being less advisable to use *similarity*-based filters. A large amount of noise filters are statistically worse than the best filters. This is the case of BBNR, CNDC, CNN, GE, RNN and HRRF.

3) SYNERGIES BETWEEN SVM AND REGRESSION NOISE FILTERS

- *Principal noise filters.* According to the *Aligned Friedman* test, CNDC is one of the best methods with low amounts of noise, whereas EF is the general recommendation when working with noisy regression data. The *Finner* test shows that both methods obtain statistical differences against a remarkable number of filters.
- *Inferior noise filters.* There are some noise filters that, although providing good results, are not as the same level as the best noise filters for SVM. This is the case of the AENN, BBNR, CNN and RNN *similarity*-filters, which are less recommended for use with SVM.

As can be seen from the previous analysis, most of the filters that obtain better synergies with each of the regression methods are based on ensembles, with the exception of AENN with NN when the noise level is high. This fact shows that the combination of regression models can improve noise detection in regression datasets in a similar way as classifier ensembles improve class noise detection in classification problems. On the other hand, *similarity* filters, although they tend to have a lower computational cost, they also tend to have less synergy with regression algorithms due to their less elaborate mechanisms for detecting noise.

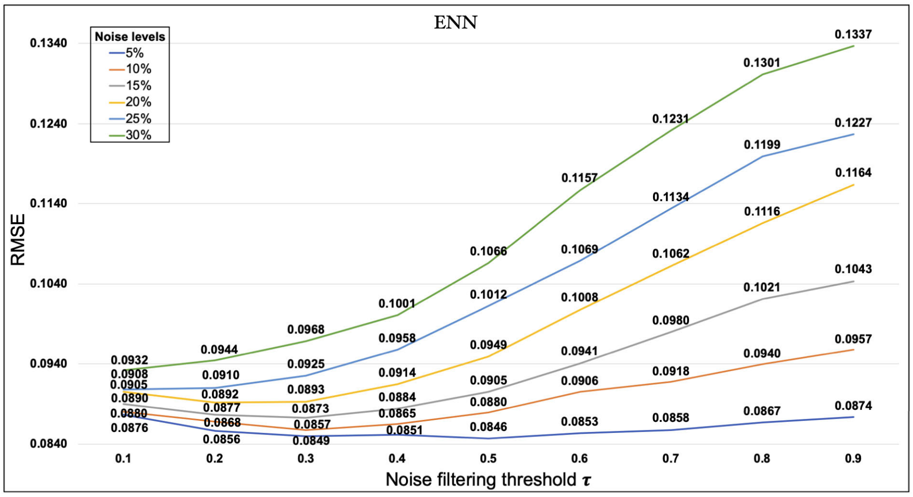
Regarding the best filters in individual datasets, those with lower ranks for RPART are not always the best in the largest number of datasets. This fact may be due to the variability in the nature of the datasets, which suggests that some filters process certain datasets better than others but they obtain similar results in average performance (as Table 8 shows). This observation is confirmed by the *Finner* test, which does not show statistical differences between these filters. Therefore, if noise filters are used with RPART in a large number of datasets, the most recommended ones are FMF and DF. However, if only a few datasets are considered, it is recommendable to test all the available noise filters and choose that offering the best results. The results of NN are similar to those obtained by RPART, where the filter with the lowest ranks does not always obtain the best performance in the largest number of datasets. In this case, AENN is the best one in 8-10 datasets for all the noise levels. Finally, for SVM, CNDC and EF are the filters with the best performance in the largest number of datasets in almost all the noise levels. Difference from the cases of RPART and NN, the best filters using SVM are also the best ones in the largest number of datasets.

D. RELATIONSHIP BETWEEN THE PROPERTIES OF DATASETS AND REGRESSION NOISE FILTERS

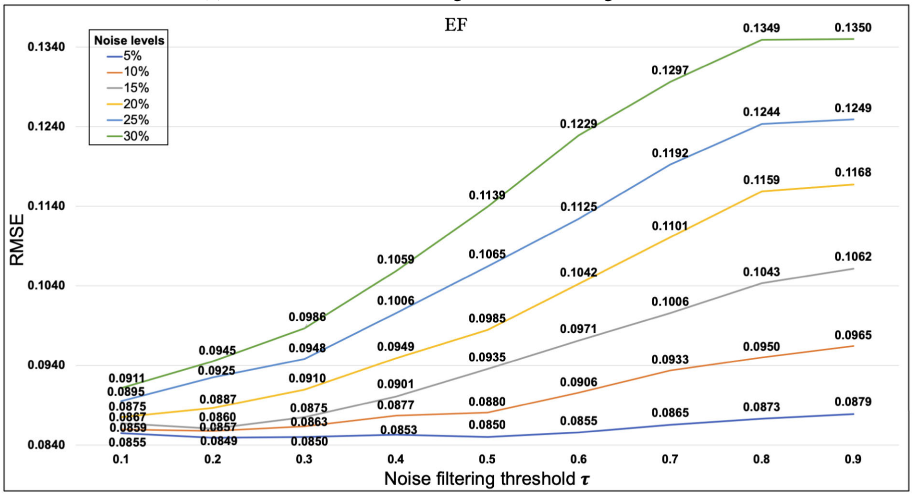
This research considers two main properties of the datasets with respect to their size: the number of samples and attributes (see Table 2). These characteristics allow delving into the relationship between the size of the datasets and the behavior of the regression noise filters implemented. This section focuses on such analysis, considering two different studies: one on the number of samples in each dataset and another on the number of attributes.

In each study, the results of the regression algorithms (RPART, NN, SVM, ELM and XGBoost) with each one of the regression filters are analyzed. In order to check the effectiveness of the regression noise filters according to the number of samples, the datasets are divided into two equal-sized groups: *small size* datasets (which contain less than 4100 samples) and *large size* datasets (which contain more than 4100 samples). On the other hand, to study the behavior of the regression noise filters with respect to the number of attributes, the datasets are also divided into two groups of equal size: *low dimensionality* datasets (which contain less than 12 attributes) and *high dimensionality* datasets (which contain more than 12 attributes). Table 10 shows the results obtained by each filter in each of the aforementioned groups of datasets, which are compared using the *Aligned Friedman* and *Finner* tests in the form of $ranks/p_{Fin}$, in a similar way to Table 9.

The results obtained show that the DF noise filter is usually that, among those adapted to regression in this research, providing the best results (although it is sometimes surpassed by FMF). This fact occurs regardless of the properties of the datasets considered (number of samples and attributes). However, it should be noted that, in general, DF does not obtain statistical differences with respect to the AENN and ENN *similarity* filters, nor with most of *ensemble* filters (with the exception of HRRF). This indicates that any of them could be recommended as they obtain results that are competitive with each other. Thus, these findings suggest



(a) Results of ENN considering different filtering thresholds.



(b) Results of EF considering different filtering thresholds.

FIGURE 1. RMSE **results of** RPART **using the noise filters** ENN **(a) and** EF **(b) with different thresholds** τ**.**

that most of *ensemble*-based filters are usually better adapted to any type of data, with independence of the number of samples and attributes: employing complementary models can improve the flexibility of the system, exploiting the strengths of each method and, finally, improving noise detection. On the contrary, these methods tend to have a higher computational cost and a greater number of parameters to configure.

It is important to note that these results (which are focused on the performance of noise filters with respect to the size of the datasets) are more stable than those shown in Section VI-C (in which the best regression noise filter usually depends on the regression algorithm used and the noise level in the data). Thus, the analysis of results in Table 10 shows that the size of the dataset (number of samples and attributes) does not appear to influence the effectiveness of regression noise filters, that is, noise filters maintain their better or worse results regardless of whether they deal with larger or smaller datasets.

E. INFLUENCE OF THE NOISE FILTERING THRESHOLD IN THE REGRESSION PERFORMANCE

The results of RPART with the ENN and EF noise filters using different thresholds τ (from 0.1 to 0.9, by increments of 0.1) are show in Figure 1. ENN and EF are considered in this section because they are the most representative noise filters within their corresponding groups (*similarity* and *ensemble* filters, respectively). Additionally, Table 11 shows the results of the *Aligned Friedman* test (*ranks* and *p*-values p_{AF}) and the *Finner* test (*p*-values p_{Fin}) in the form *ranks*/p_{Fin} after comparing different thresholds with the noise filters.

1) RELATIONSHIP BETWEEN THE FILTERING THRESHOLD AND THE NOISE LEVEL

The results in Figure 1a show that the performance of ENN for each noise level changes with the different threshold values. As the noise level increases, the lower threshold values achieve the best results. This fact is due to more noisy

TABLE 10. Results of *Aligned Friedman* (*Ranks*) and *Finner* (p_{Fin}) tests on the relationship between the properties of the datasets (number of samples and attributes) and the behavior of the noise filters (the best results are remarked in bold).

Noise filters & number of samples						
Noise Filter	5%	10%	15%	20%	25%	30%
	Small size datasets					
AENN	302/8.80E-02	286/6.68E-02	253/2.87E-01	245/3.72E-01	251/1.44E-01	230/3.06E-01
BBNR	351/**3.09E-03**	390/**6.61E-06**	420/**1.09E-07**	423/**6.03E-08**	439/**6.26E-10**	460/**1.31E-11**
CNDC	363/**1.39E-03**	393/**5.58E-06**	425/**6.72E-08**	427/**4.30E-08**	419/**1.05E-08**	426/**2.86E-09**
CNN	638/**0.00E+00**	624/**0.00E+00**	621/**0.00E+00**	616/**0.00E+00**	604/**0.00E+00**	609/**0.00E+00**
ENN	270/3.30E-01	246/3.57E-01	259/2.46E-01	257/2.54E-01	267/6.87E-02	272/**4.55E-02**
GE	445/**1.19E-07**	488/**4.35E-12**	496/**8.04E-13**	520/**5.77E-15**	517/**6.66E-16**	523/**0.00E+00**
RNN	622/**0.00E+00**	638/**0.00E+00**	635/**0.00E+00**	630/**0.00E+00**	614/**0.00E+00**	608/**0.00E+00**
CVCF	229/8.84E-01	211/8.03E-01	213/7.80E-01	209/8.49E-01	215/5.13E-01	217/4.60E-01
DF	**222**	**201**	**200**	199/9.90E-01	**183**	**181**
EF	226/9.34E-01	225/6.48E-01	216/7.48E-01	230/5.55E-01	248/1.51E-01	246/1.67E-01
FMF	235/8.17E-01	213/7.98E-01	219/7.35E-01	**198**	195/7.63E-01	194/7.81E-01
HRRF	457/**3.11E-08**	508/**1.29E-13**	515/**2.89E-14**	531/**9.99E-16**	553/**0.00E+00**	559/**0.00E+00**
IPF	300/8.80E-02	267/1.60E-01	228/6.13E-01	219/7.07E-01	206/6.34E-01	195/7.81E-01
IRF	247/6.38E-01	216/7.81E-01	207/8.63E-01	204/9.09E-01	197/7.50E-01	186/9.12E-01
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	Large size datasets					
AENN	274/2.82E-01	260/3.27E-01	253/3.64E-01	239/3.96E-01	227/4.56E01	219/5.44E-01
BBNR	331/**1.84E-02**	347/**2.51E-03**	336/**3.33E-03**	359/**1.71E-04**	377/**9.79E-06**	390/**1.67E-06**
CNDC	383/**2.43E-04**	412/**2.78E-06**	450/**5.30E-09**	468/**7.95E-11**	481/**2.15E-12**	485/**8.37E-13**
CNN	602/**0.00E+00**	596/**0.00E+00**	598/**0.00E+00**	595/**0.00+00**	596/**0.00E+00**	594/**0.00E+00**
ENN	240/7.56E-01	236/6.69E-01	228/6.84E-01	225/5.87E-01	219/5.14E-01	218/5.44E-01
GE	411/**1.31E-05**	447/**3.11E-08**	494/**5.34E-12**	510/**5.05E-14**	525/**6.66E-16**	526/**0.00E+00**
RNN	610/**0.00E+00**	613/**0.00E+00**	614/**0.00E+00**	604/**0.00E+00**	609/**0.00E+00**	604/**0.00E+00**
CVCF	306/6.23E-02	280/1.52E-01	252/3.64E-01	247/3.28E-01	240/3.47E-01	246/2.56E-01
DF	**225**	**215**	**208**	**199**	**191**	**190**
EF	236/8.12E-01	224/8.52E-01	216/8.64E-01	223/5.91E-01	221/5.12E-01	225/4.72E-01
FMF	234/8.12E-01	222/8.59E-01	222/7.65E-01	206/8.67E-01	196/9.10E-01	192/9.63E-01
HRRF	420/**5.76E-06**	467/**2.02E-09**	496/**4.78E-12**	521/**6.77E-15**	534/**0.00E+00**	538/**0.00E+00**
IPF	314/**4.33E-02**	290/1.05E-01	267/2.25E-01	255/2.67E-01	238/3.47E-01	228/4.66E-01
IRF	322/**2.86E-02**	297/7.77E-02	274/1.87E-01	257/2.67E-01	254/2.12E-01	251/2.32E-01
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

Noise filters & number of attributes						
	Low dimensionality datasets					
AENN	299/1.33E-01	283/1.35E-01	246/4.92E-01	250/4.11E-01	259/1.56E-01	244/2.48E-01
BBNR	332/**2.26E-02**	379/**7.59E-05**	386/**2.22E-05**	382/**2.08E-05**	409/**1.37E-07**	435/**1.34E-09**
CNDC	372/**1.03E-03**	400/**8.30E-06**	437/**3.99E-08**	448/**3.79E-09**	442/**1.28E-09**	445/**3.28E-10**
CNN	626/**0.00E+00**	614/**0.00E+00**	612/**0.00E+00**	609/**0.00E+00**	591/**0.00E+00**	601/**0.00E+00**
ENN	269/3.90E-01	247/4.57E-01	253/4.28E-01	241/4.99E-01	244/2.77E-01	241/2.49E-01
GE	425/**4.18E-06**	457/**4.15E-09**	477/**9.05E-11**	502/**4.70E-13**	505/**2.39E-14**	508/**4.33E-15**
RNN	608/**0.00E+00**	625/**0.00+00**	623/**0.00E+00**	617/**0.00E+00**	600/**0.00E+00**	598/**0.00E+00**
CVCF	249/6.81E-01	239/5.46E-01	237/5.28E-01	235/5.21E-01	235/3.70E-01	228/3.52E-01
DF	**229**	**211**	**208**	206/9.50E-01	**190**	**185**
EF	235/8.79E-01	221/8.28E-01	218/7.99E-01	219/7.21E-01	227/4.16E-01	230/3.52E-01
FMF	243/7.62E-01	220/8.28E-01	230/6.14E-01	**203**	205/7.09E-01	204/6.27E-01
HRRF	431/**2.61E-06**	479/**1.45E-10**	497/**4.06E-12**	518/**3.28E-14**	549/**0.00E+00**	554/**0.00E+00**
IPF	308/9.33E-02	282/1.35E-01	242/5.28E-01	235/5.21E-01	218/5.28E-01	215/4.77E-01
IRF	282/2.59E-01	250/4.52E-01	240/5.28E-01	240/4.99E-01	232/3.70E-01	218/4.67E-01
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**
	High dimensionality datasets					
AENN	290/1.25E-01	271/1.74E-01	258/2.63E-01	239/4.36E-01	225/4.49E-01	212/6.07E-01
BBNR	353/**1.92E-03**	366/**1.57E-04**	387/**8.84E-06**	406/**3.42E-07**	421/**1.36E-08**	430/**3.39E-09**
CNDC	374/**3.10E-04**	406/**1.96E-06**	438/**1.04E-08**	446/**1.26E-09**	454/**8.23E-11**	461/**2.43E-11**
CNN	617/**0.00E+00**	610/**0.00E+00**	609/**0.00E+00**	603/**0.00E+00**	605/**0.00E+00**	602/**0.00E+00**
ENN	245/5.74E-01	235/5.27E-01	233/5.48E-01	238/4.36E-01	241/2.94E-01	251/1.85E-01
GE	433/**3.80E-07**	478/**5.01E-11**	512/**4.41E-14**	527/**6.66E-16**	533/**0.00E+00**	535/**0.00E+00**
RNN	623/**0.00E+00**	628/**0.00E+00**	625/**0.00E+00**	620/**0.00E+00**	621/**0.00E+00**	614/**0.00E+00**
CVCF	272/2.33E-01	241/4.66E-01	220/6.85E-01	215/6.38E-01	214/5.53E-01	225/4.41E-01
DF	**218**	**206**	**200**	**194**	**186**	187/9.72E-01
EF	224/8.83E-01	225/6.63E-01	214/7.58E-01	231/4.44E-01	241/2.94E-01	244/2.36E-01
FMF	228/8.41E-01	216/8.08E-01	210/8.00E-01	202/8.54E-01	189/9.35E-01	**186**
HRRF	447/**7.13E-08**	501/**1.33E-12**	522/**8.66E-15**	536/**0.00E+00**	543/**0.00E+00**	546/**0.00E+00**
IPF	306/5.65E-02	273/1.74E-01	248/3.62E-01	234/4.39E-01	223/4.49E-01	206/6.47E-01
IRF	277/2.10E-01	251/3.53E-01	231/5.48E-01	217/6.31E-01	211/5.66E-01	208/6.47E-01
p_{AF}	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**	**0.00E+00**

samples are need to be removed, implying that lower values of τ usually achieve better performance results with a more aggressive filtering. EF (Figure 1b) presents similar results regarding the efficiency of the threshold when varying the

TABLE 11. Results of the *Aligned Friedman* (*Ranks* and p_{AF}) and *Finner tests* (p_{Fin}) after comparing the noise filters ENN and EF with different thresholds (the best results are remarked in bold).

Metric	Ranks/p_{Fin}					
Noise	**5%**	**10%**	**15%**	**20%**	**25%**	**30%**
τ	**ENN**					
0.1	94/1.62E-01	66/4.28E-01	66/4.46E-01	56/6.36E-01	47/8.92E-01	**43**
0.2	76/6.32E-01	58/6.82E-01	52/9.55E-01	**46**	**44**	44/9.59E-01
0.3	71/7.13E-01	**50**	**51**	47/9.60E-01	47/8.92E-01	47/8.35E-01
0.4	75/6.32E-01	57/6.88E-01	53/9.47E-01	57/6.36E-01	55/6.05E-01	54/6.03E-01
0.5	**65**	76/1.89E-01	70/3.84E-01	71/2.06E-01	80/**4.57E-02**	77/6.69E-01
0.6	89/2.36E-01	108/**9.35E-04**	105/**2.15E-03**	112/**1.52E-04**	114/**4.50E-05**	119/**0.00E+00**
0.7	100/9.52E-02	124/**2.40E-05**	130/**5.00E-06**	134/**0.00E+00**	135/**0.00E+00**	135/**0.00E+00**
0.8	120/**3.56E-03**	134/**1.00E-06**	141/**0.00E+00**	144/**0.00E+00**	144/**0.00E+00**	146/**0.00E+00**
0.9	125/**2.55E-03**	141/**0.00E+00**	147/**0.00E+00**	148/**0.00E+00**	149/**0.00E+00**	149/**0.00E+00**
p_{AF}	**1.64E-03**	**6.40E-11**	**8.10E-11**	**6.10E-11**	**6.70E-11**	**5.90E-11**
τ	**EF**					
0.1	**69**	53/8.09E-01	45/7.96E-01	40/9.52E-01	**35**	**33**
0.2	**69**/9.91E-01	**48**	**40**	**39**	**35**/9.81E-01	37/8.08E-01
0.3	**69**/9.95E-01	50/9.36E-01	43/8.39E-01	43/8.52E-01	41/7.75E-01	41/6.82E-01
0.4	74/8.92E-01	65/4.01E-01	55/4.72E-01	58/3.12E-01	58/2.00E-01	57/1.89E-01
0.5	**69**/9.92E-01	75/1.70E-01	81/**2.06E-02**	78/**2.77E-02**	86/**3.19E-03**	88/**1.34E-03**
0.6	92/3.03E-01	104/**1.35E-03**	118/**4.00E-06**	119/**2.00E-06**	124/**0.00E+00**	126/**0.00E+00**
0.7	115/**1.23E-02**	133/**1.00E-06**	137/**0.00E+00**	140/**0.00E+00**	140/**0.00E+00**	140/**0.00E+00**
0.8	127/**1.51E-03**	141/**0.00E+00**	146/**0.00E+00**	147/**0.00E+00**	148/**0.00E+00**	146/**0.00E+00**
0.9	130/**1.47E-03**	145/**0.00E+00**	149/**0.00E+00**	149/**0.00E+00**	148/**0.00E+00**	146/**0.00E+00**
p_{AF}	**6.71E-06**	**6.80E-11**	**8.70E-11**	**6.50E-11**	**9.40E-11**	**8.00E-11**

noise level, that is, lower threshold values obtain better results when the noise level increases.

2) VARIABILITY OF RESULTS DEPENDING ON THE FILTERING THRESHOLD

The best threshold for ENN at 5% of noise is 0.5. However, the performance does not show large variations among the different threshold values at 5% of noise, with only a 3.55% of deterioration. For 30% of noise, the performance between the best threshold (0.1) and the worst threshold (0.9) represents a 43.45% of loss of performance. For EF the best threshold is 0.2 up to 15% of noise and 0.1 in the highest noise levels. The performance variation at each noise level with respect to the threshold presents a similar behavior in both filters. The performance variation of EF at 5% is 3.53%, whereas at 30% of noise the performance is variated a 48.18%. This fact indicates that the threshold works in a similar way regardless of the filter. The influence of the threshold is not so high when working with low noise levels, whereas the filters improve their performance when increasing the noise level if the right threshold is chosen.

3) STATISTICAL SIGNIFICANCE OF THE CONCLUSIONS REACHED

The ranks obtained by the *Aligned Friedman* test support the aforementioned conclusions. Table 11 shows that the best thresholds for ENN change with respect to the noise level. For 5% of noise level, the *Finner* test shows significant differences only with the thresholds 0.8 and 0.9, whereas it shows significant differences with thresholds higher than 0.5 for the rest of noise levels. EF presents a similar behavior than that of ENN. These results are complemented with the *Finner* test, which shows statistical differences for all the thresholds higher than 0.4 in most of the comparisons. These facts validate the good working of low threshold values for both filters in the experiments performed.

VII. CONCLUSION AND FUTURE WORK

The results obtained in this research have shown that the presence of regressand noise in the data negatively affects the performance of the different types of regression methods. However, the impact of noise on the models created with each algorithm is very different. Thus, NN is the most sensitive method to regressand noise, since the inclusion of a single noisy sample alters its predictions, whereas RPART provides more robust results due to its pruning mechanisms. These conclusions are similar to those obtained in the field of classification when dealing with class noise, which can be considered equivalent to regressand noise in regression. The most notable difference from the well-known robustness results in classification occurs with SVM. In classification, it is often considered a noise-sensitive algorithm, as noisy samples easily alter the decision hyperplane. However, regression-SVM can be seen as a robust algorithm against regressand noise since, in this case, its ϵ-insensitive loss function allows dealing with noisy samples to reduce their impact when obtaining the hyperplane. This fact indicates that, contrary to classification, the usage of SVM and other methods based on it can be interesting with in this scenario.

On the other hand, a new way of adapting the noise filters used in classification to regression problems has been proposed. This adaptation has shown to improve other previous classification filtering adaptation proposals. The application of noise filters adapted to regression has also shown to be useful with respect to not applying any data preprocessing, so the usage of these techniques, despite not being very widespread, should be considered when addressing noisy regression problems. Depending on the regression algorithm used, it is advisable to use some noise filters adapted to regression or others. Thus, for `RPART`, the filters from which a better performance can be expected are `DF` and `FMF`, whereas `CNN` and `RNN` are less recommended. For `NN` it is also recommended to use `DF` when the noise level is low-medium and `AENN` when the noise level is high. The most recommended filter for `SVM` is `CNDC` with low noise levels, while `EF` is the recommendation when working with higher noise levels. In general, a worse performance can be expected from *similarity*-based filters with any of the regression methods considered regardless of the noise level. An additional study has shown that the size of the dataset (number of samples and attributes) does not appear to influence the effectiveness of regression noise filters, that is, noise filters maintain their better or worse results regardless of whether they deal with larger or smaller datasets. The analysis of these results has shown that the `DF` filter and others based on ensembles provided the most robust results and are, therefore, recommended in general terms when dealing with noisy regression data. Note that ensemble filters generally provide a better noise detection at the cost of increasing their overall computational complexity in most of the cases. Finally, the effect of different values of the threshold parameter in the noise filters has been also studied, showing the importance of performing an aggressive filtering (that is, using lower values in the threshold) when the noise level increases.

Note that, despite the good performance of the regression noise filters designed, they require setting the threshold τ. In addition, this research has considered regressand noise to study the behavior of both regression techniques and noise filters. In future works we aim to find an automatic way to set the optimal threshold in each dataset. We also plan to analyze the influence of attribute noise on the performance of different regression methods, as well as studying different alternatives to improve the performance of the methods with this type of data. Finally, since a direct relationship between the size of the dataset and the performance of the regression-adapted noise filters has not been observed, it is necessary to study other properties of the data, in a similar way as it has been performed in classification using *data-complexity* measures [11] or using the known as *quality indices* [57], which allow measuring certain properties of the data with independence of the type of output variable.

ACKNOWLEDGMENT

The authors thank the anonymous reviewers for the time spent on this research work, as well as all their interesting corrections and suggestions, which provided important and valuable feedback to identify essential issues and helped to improve the quality of the manuscript.

REFERENCES

[1] Z. Kang, H. Pan, S. C. H. Hoi, and Z. Xu, "Robust graph learning from noisy data," *IEEE Trans. Cybern.*, vol. 50, no. 5, pp. 1833–1843, May 2020.

[2] J. A. Saez and E. Corchado, "KSUFS: A novel unsupervised feature selection method based on statistical tests for standard and big data problems," *IEEE Access*, vol. 7, pp. 99754–99770, 2019.

[3] S.-A.-N. Alexandropoulos, S. B. Kotsiantis, and M. N. Vrahatis, "Data preprocessing in predictive data mining," *Knowl. Eng. Rev.*, vol. 34, no. e1, pp. 1–33, 2019.

[4] Y. Yin, L. Long, and X. Deng, "Dynamic data mining of sensor data," *IEEE Access*, vol. 8, pp. 41637–41648, 2020.

[5] X. Zhu and X. Wu, "Class noise vs. attribute noise: A quantitative study," *Artif. Intell. Rev.*, vol. 22, no. 3, pp. 177–210, Nov. 2004.

[6] T. Nguyen-Van, "A discrete-time state estimation for nonlinear systems with noises," *IEEE Access*, vol. 8, pp. 147089–147096, 2020.

[7] P. Borah, D. Gupta, and M. Prasad, "Improved 2-norm based fuzzy least squares twin support vector machine," in *Proc. IEEE Symp. Ser. Comput. Intell. (SSCI)*, Nov. 2018, pp. 412–419.

[8] S. Balasundaram and D. Gupta, "On optimization based extreme learning machine in primal for regression and classification by functional iterative method," *Int. J. Mach. Learn. Cybern.*, vol. 7, no. 5, pp. 707–728, Oct. 2016.

[9] X. Xu and S.-L. Huang, "Maximal correlation regression," *IEEE Access*, vol. 8, pp. 26591–26601, 2020.

[10] U. Gupta and D. Gupta, "Least squares large margin distribution machine for regression," *Int. J. Speech Technol.*, vol. 51, no. 10, pp. 7058–7093, Oct. 2021.

[11] J. A. Sáez, J. Luengo, and F. Herrera, "Predicting noise filtering efficacy with data complexity measures for nearest neighbor classification," *Pattern Recognit.*, vol. 46, no. 1, pp. 355–364, Jan. 2013.

[12] B. Frenay and M. Verleysen, "Classification in the presence of label noise: A survey," *IEEE Trans. Neural Netw. Learn. Syst.*, vol. 25, no. 5, pp. 845–869, May 2014.

[13] J. A. Sáez and E. Corchado, "ANCES: A novel method to repair attribute noise in classification problems," *Pattern Recognit.*, vol. 121, Jan. 2022, Art. no. 108198.

[14] W. Yuan, D. Guan, T. Ma, and A. M. Khattak, "Classification with class noises through probabilistic sampling," *Inf. Fusion*, vol. 41, pp. 57–67, May 2018.

[15] K. Nikolaidis, T. Plagemann, S. Kristiansen, V. Goebel, and M. Kankanhalli, "Using under-trained deep ensembles to learn under extreme label noise: A case study for sleep apnea detection," *IEEE Access*, vol. 9, pp. 45919–45934, 2021.

[16] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Tackling the problem of classification with noisy data using multiple classifier systems: Analysis of the performance and robustness," *Inf. Sci.*, vol. 247, pp. 1–20, Oct. 2013.

[17] D. F. Nettleton, A. Orriols-Puig, and A. Fornells, "A study of the effect of different types of noise on the precision of supervised learning techniques," *Artif. Intell. Rev.*, vol. 33, no. 4, pp. 275–306, Apr. 2010.

[18] J. Quinlan, *C4.5: Programs for Machine Learning*. Amsterdam, The Netherlands: Elsevier, 2014.

[19] G. Biau and L. Devroye, *Lectures on the Nearest Neighbor Method* (Springer Series in the Data Sciences). Cham, Switzerland: Springer, 2015.

[20] J. A. Sáez, B. Krawczyk, and M. Wozniak, "On the influence of class noise in medical data classification: Treatment using noise filtering methods," *Appl. Artif. Intell.*, vol. 30, no. 6, pp. 590–609, Jul. 2016.

[21] C. E. Brodley and M. A. Friedl, "Identifying mislabeled training data," *J. Artif. Intell. Res.*, vol. 11, pp. 131–167, Aug. 1999.

[22] M. Kordos and M. Blachnik, "Instance selection with neural networks for regression problems," in *Proc. Int. Conf. Artif. Neural Netw.*, 2012, pp. 263–270.

[23] L. Devroye, L. Györfi, and G. Lugosi, "Condensed and edited nearest neighbor rules," in *A Probabilistic Theory of Pattern Recognition*. New York, NY, USA: Springer, 1996, pp. 303–313.

[24] L. Breiman, J. Friedman, C. Stone, and R. Olshen, *Classification Regression Trees*. Boca Raton, FL, USA: CRC Press, 1984.

[25] B. Schlkopf, A. Smola, and F. Bach, *Learning With Kernels: Support Vector Machines, Regularization, Optimization, and Beyond*. Cambridge, MA, USA: MIT Press, 2018.

[26] S. Ding, H. Zhao, Y. Zhang, X. Xu, and R. Nie, "Extreme learning machine: Algorithm, theory and applications," *Artif. Intell. Rev.*, vol. 44, no. 1, pp. 103–115, Jun. 2015.

[27] T. Chen and C. Guestrin, "XGBoost: A scalable tree boosting system," in *Proc. 22nd ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*, Aug. 2016, pp. 785–794.

[28] J. Derrac, S. García, D. Molina, and F. Herrera, "A practical tutorial on the use of nonparametric statistical tests as a methodology for comparing evolutionary and swarm intelligence algorithms," *Swarm Evol. Comput.*, vol. 1, no. 1, pp. 3–18, Mar. 2011.

[29] S. Chaudhury and T. Yamasaki, "Robustness of adaptive neural network optimization under training noise," *IEEE Access*, vol. 9, pp. 37039–37053, 2021.

[30] T. M. Khoshgoftaar and P. Rebours, "Improving software quality prediction by noise filtering techniques," *J. Comput. Sci. Technol.*, vol. 22, no. 3, pp. 387–396, May 2007.

[31] W. Zhang, D. Wang, and X. Tan, "Robust class-specific autoencoder for data cleaning and classification in the presence of label noise," *Neural Process. Lett.*, vol. 50, no. 2, pp. 1845–1860, Oct. 2019.

[32] J. A. Sáez, M. Galar, J. Luengo, and F. Herrera, "Analyzing the presence of noise in multi-class problems: Alleviating its influence with the one-vs-one decomposition," *Knowl. Inf. Syst.*, vol. 38, no. 1, pp. 179–206, Jan. 2014.

[33] R. Hansch, *Handbook of Random Forests: Theory and Applications for Remote Sensing*. Singapore: World Scientific, 2018.

[34] A. Gómez-Ríos, J. Luengo, and F. Herrera, "A study on the noise label influence in boosting algorithms: AdaBoost, GBM and XGBoost," in *Hybrid Artificial Intelligent Systems*, F. Martínez, R. Urraca, H. Quintián, and E. Corchado, Eds. Cham, Switzerland: Springer, 2017, pp. 268–280.

[35] R. McDonald, D. Hand, and I. Eckley, "An empirical comparison of three boosting algorithms on real data sets with artificial class noise," in *Multiple Classifier Systems*, T. Windeatt and F. Roli, Eds. Berlin, Germany: Springer, 2003, pp. 35–44.

[36] P. Borah and D. Gupta, "A two-norm squared fuzzy-based least squares twin parametric-margin support vector machine," in *Machine Intelligence and Signal Analysis*, M. Tanveer and R. Pachori, Eds. Singapore: Springer, 2019, pp. 119–134.

[37] V. Cherkassky and Y. Ma, "Practical selection of SVM parameters and noise estimation for SVM regression," *Neural Netw.*, vol. 17, no. 1, pp. 113–126, Jan. 2004.

[38] J. A. K. Suykens, J. De Brabanter, L. Lukas, and J. Vandewalle, "Weighted least squares support vector machines: Robustness and sparse approximation," *Neurocomputing*, vol. 48, nos. 1–4, pp. 85–105, Oct. 2002.

[39] S. Balasundaram and D. Gupta, "Training Lagrangian twin support vector regression via unconstrained convex minimization," *Knowl.-Based Syst.*, vol. 59, pp. 85–96, Mar. 2014.

[40] J. Suykens and J. Vandewalle, "Least squares support vector machine classifiers," *Neural Process. Lett.*, vol. 9, no. 3, pp. 293–300, 1999.

[41] X. Yang, L. Tan, and L. He, "A robust least squares support vector machine for regression and classification with noise," *Neurocomputing*, vol. 140, pp. 41–52, Sep. 2014.

[42] Y. Sun, Y. Tian, Y. Xu, and J. Li, "Limited gradient descent: Learning with noisy labels," *IEEE Access*, vol. 7, pp. 168296–168306, 2019.

[43] L. P. F. Garcia, A. C. Lorena, and A. C. P. L. F. Carvalho, "A study on class noise detection and elimination," in *Proc. Brazilian Symp. Neural Netw.*, Oct. 2012, pp. 13–18.

[44] I. Tomek, "An experiment with the edited nearest-neighbor rule," *IEEE Trans. Syst., Man, Cybern.*, vol. SMC-6, no. 6, pp. 448–452, Jun. 1976.

[45] S. Delany and P. Cunningham, "An analysis of case-base editing in a spam filtering system," in *Proc. Eur. Conf. Case-Based Reasoning*, 2004, pp. 128–141.

[46] G. Gates, "The reduced nearest neighbor rule (Corresp.)," *IEEE Trans. Inf. Theory*, vol. IT-18, no. 3, pp. 431–433, May 1972.

[47] S. Verbaeten and A. Van, "Ensemble methods for noise elimination in classification problems," in *Proc. Int. Workshop Multiple Classifier Syst.*, Jun. 2003, pp. 317–325.

[48] L. P. F. Garcia, A. C. Lorena, S. Matwin, and A. C. P. L. F. de Carvalho, "Ensembles of label noise filters: A ranking approach," *Data Mining Knowl. Discovery*, vol. 30, no. 5, pp. 1192–1216, Sep. 2016.

[49] Z. Nematzadeh, R. Ibrahim, and A. Selamat, "Improving class noise detection and classification performance: A new two-filter CNDC model," *Appl. Soft Comput.*, vol. 94, Sep. 2020, Art. no. 106428.

[50] J. Koplowitz and T. A. Brown, "On the relation of performance to editing in nearest neighbor rules," *Pattern Recognit.*, vol. 13, no. 3, pp. 251–255, Jan. 1981.

[51] A. Miranda, L. Garcia, A. Carvalho, and A. Lorena, "Use of classification algorithms in noise detection and elimination," in *Hybrid Artificial Intelligence Systems*, E. Corchado, X. Wu, E. Oja, A. Herrero, and B. Baruque, Eds. Berlin, Germany: Springer, 2009, pp. 417–424.

[52] S. Verbaeten, "Identifying mislabeled training examples in ILP classification problems," in *Proc. 12th Belgian-Dutch Conf. Mach. Learn.*, 2002, pp. 1–8.

[53] M. Popescu, V. Balas, L. Perescu-Popescu, and N. Mastorakis, "Multilayer perceptron and neural networks," *World Sci. Eng. Academy Soc.*, vol. 8, no. 7, pp. 579–588, 2009.

[54] S. Xu, "Bayesian Naïve Bayes classifiers to text classification," *J. Inf. Sci.*, vol. 44, no. 1, pp. 48–59, Feb. 2018.

[55] Z. Yu, D. Wang, Z. Zhao, C. L. P. Chen, J. You, H.-S. Wong, and J. Zhang, "Hybrid incremental ensemble learning for noisy real-world data classification," *IEEE Trans. Cybern.*, vol. 49, no. 2, pp. 403–416, Feb. 2019.

[56] N. Jankowski and M. Grochowski, "Comparison of instances seletion algorithms I. Algorithms survey," in *Artificial Intelligence and Soft Computing*, L. Rutkowski, J. Siekmann, R. Tadeusiewicz, and L. Zadeh, Eds. Berlin, Germany: Springer, 2004, pp. 598–603.

[57] J. A. Saez and E. Corchado, "A meta-learning recommendation system for characterizing unsupervised problems: On using quality indices to describe data conformations," *IEEE Access*, vol. 7, pp. 63247–63263, 2019.

JUAN MARTÍN is currently pursuing the Ph.D. degree in marine energy and propulsion. He is currently working with the University of Salamanca, Spain, holding a 'Retención de jóvenes talentos'' predoctoral grant from Castile and Leon Regional Government. His main research interests include smart agriculture, data preprocessing, ensemble learning, and noisy data in regression.

JOSÉ A. SÁEZ received the Ph.D. degree in computer science and computing technology from the University of Granada. He is currently working as a Professor with the University of Granada, Spain. His main research interests include data mining, data preprocessing and data transformation tasks in knowledge discovery in databases, including noisy data in classification, discretization methods, imbalanced learning, performance evaluation methods, non-parametrical statistical tests, and unsupervised learning. His main research line is that of noisy data in classification tasks, counting with more than 20 publications on this topic.

EMILIO CORCHADO received the Ph.D. degree in computer science from the University of Salamanca.

He is currently a Full Professor in computer and automatic science with the University of Salamanca, Spain. He has published over 100 peer-reviewed articles in a range of topics from knowledge management and risk analysis, intrusion detection systems, food industry, artificial vision, and modeling of industrial processes. He has patented software models and he owns the IP of more than ten ICT tools and models. He has actively contributed to several current projects in the EU, including SOFTCOMP, IT4Innovation, ICT Action COST IC1303, and IntelliCIS NISIS. His research interests include neural networks, with a particular focus on exploratory projection pursuit, maximum likelihood hebbian learning, self-organising maps, multiple classifier systems, and hybrid systems. He has been the Organizing Chair, the Program Committee Chair, the Session Chair, and the General Chair of a number of conferences, such as International Conference on Hybrid Artificial Intelligence Systems (HAIS), International Conference on Intelligent Data Engineering and Automated Learning (IDEAL), and International Conference on Knowledge-Based Intelligent Information and Engineering Systems (KES). He is a RTD Expert hired by international organizations, such as the European Commission, the Grant Agency of Czech Republic, the Spanish National Agency for Assessment and Forecasting, and has collaborated with SMEs and new companies in the innovation field in about 40 projects. He was the Chair of the IEEE Spanish Section, from 2014 to 2015.