Contents lists available at SciVerse ScienceDirect

Information Sciences

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

Tackling the problem of classification with noisy data using Multiple Classifier Systems: Analysis of the performance and robustness

José A. Sáez [a,*], Mikel Galar [b], Julián Luengo [c], Francisco Herrera [a]

[a] *Department of Computer Science and Artificial Intelligence, University of Granada, CITIC-UGR, Granada 18071, Spain*
[b] *Department of Automática y Computación, Universidad Pública de Navarra, Pamplona 31006, Spain*
[c] *Department of Civil Engineering, LSI, University of Burgos, Burgos 09006, Spain*

A R T I C L E I N F O

Article history:
Received 25 May 2012
Received in revised form 13 May 2013
Accepted 2 June 2013
Available online 13 June 2013

Keywords:
Noisy data
Class noise
Attribute noise
Multiple Classifier System
Classification

A B S T R A C T

Traditional classifier learning algorithms build a unique classifier from the training data. Noisy data may deteriorate the performance of this classifier depending on the degree of sensitiveness to data corruptions of the learning method. In the literature, it is widely claimed that building several classifiers from noisy training data and combining their predictions is an interesting method of overcoming the individual problems produced by noise in each classifier. This statement is usually not supported by thorough empirical studies considering problems with different types and levels of noise. Furthermore, in noisy environments, the noise robustness of the methods can be more important than the performance results themselves and, therefore, it must be carefully studied. This paper aims to reach conclusions on such aspects focusing on the analysis of the behavior, in terms of performance and robustness, of several Multiple Classifier Systems against their individual classifiers when these are trained with noisy data. In order to accomplish this study, several classification algorithms, of varying noise robustness, will be chosen and compared with respect to their combination on a large collection of noisy datasets. The results obtained show that the success of the Multiple Classifier Systems trained with noisy data depends on the individual classifiers chosen, the decisions combination method and the type and level of noise present in the dataset, but also on the way of creating diversity to build the final system. In most of the cases, they are able to outperform all their single classification algorithms in terms of global performance, even though their robustness results will depend on the way of introducing diversity into the Multiple Classifier System.

© 2013 Elsevier Inc. All rights reserved.

1. Introduction

Classifier learning algorithms aim to extract the knowledge from a problem from the available set of labeled examples (training set) in order to predict the class for new, previously unobserved, examples [8]. Classic learning algorithms [36,4] build a unique model, called a classifier, which attempts to generalize the peculiarities of the training set. Therefore, the success of these methods, that is, their ability to classify new examples, highly depends on the usage of a concrete feature descriptor and a particular inference procedure, and directly on the training data.

* Corresponding author. Tel.: +34 958 240598; fax: +34 958 243317.
E-mail addresses: smja@decsai.ugr.es (J.A. Sáez), mikel.galar@unavarra.es (M. Galar), jluengo@ubu.es (J. Luengo), herrera@decsai.ugr.es (F. Herrera).

0020-0255/$ - see front matter © 2013 Elsevier Inc. All rights reserved.
http://dx.doi.org/10.1016/j.ins.2013.06.002

Real-world data, which is the input of the classifier learning algorithms, are affected by several components [42,52,37]; among them, the presence of noise is a key factor. Noise is an unavoidable problem, which affects the data collection and data preparation processes in Data Mining applications, where errors commonly occur [48,50]. The performance of the classifiers built under such circumstances will heavily depend on the quality of the training data, but also on the robustness against noise of the classifier itself. Hence, classification problems containing noise are complex problems and accurate solutions are often difficult to achieve with a unique classifier system – particularly if this classifier is noise-sensitive.

Several works have claimed that simultaneously using classifiers of different types, complementing each other, improves classification performance on difficult problems, such as satellite image classification [27], fingerprint recognition [30] and foreign exchange market prediction [33]. Multiple Classifier Systems (MCSs) [15,14,32,45] are presented as a powerful solution to these difficult classification problems, because they build several classifiers from the same training data and therefore allow the simultaneous usage of several feature descriptors and inference procedures. An important issue when using MCSs is the way of creating *diversity* among the classifiers [24], which is necessary to create discrepancies among their decisions and hence, to take advantage from their combination.

MCSs have been traditionally associated with the capability of working accurately with problems involving noisy data [15]. The main reason supporting this hypothesis could be the same as one of the main motivations for combining classifiers: the improvement of the generalization capability (due to the complementarity of each classifier), which is a key question in noisy environments, since it might allow one to avoid the overfitting of the new characteristics introduced by the noisy examples [39]. Most of the works studying MCSs and noisy data are focused on techniques like bagging and boosting [7,25,20], which introduce diversity considering different samples of the set of training examples and use only one baseline classifier. For example, in [7] the suitability of randomization, bagging and boosting to improve the performance of C4.5 was studied. The authors reached the conclusion that with a low noise level, boosting is usually more accurate than bagging and randomization. However, bagging outperforms the other methods when the noise level increases. Similar conclusions were obtained in the paper of Maclin and Opitz [25]. Other works [20] compare the performance of boosting and bagging techniques dealing with imbalanced and noisy data, reaching also the conclusion that bagging methods generally outperforms boosting ones. Nevertheless, explicit studies about the adequacy of MCSs (different from bagging and boosting, that is, those introducing diversity using different base classifiers) to deal with noisy data have not been carried out yet. Furthermore, most of the existing works are focused on a concrete type of noise and on a concrete combination rule. On the other hand, when data are suffering from noise, a proper study on how the robustness of each single method influences the robustness of the MCS is necessary, but this fact is usually overlooked in the literature.

This paper aims to develop a thorough analysis of the behavior of several MCSs with noisy training data with respect to their individual components (classifiers), studying to what extent the behavior of these MCSs depends on that of the individual classifiers. The classic hypothesis about the good behavior of MCSs with noisy data will be checked in detail and the conditions under which the MCSs studied work well with noisy data will be analyzed. In order to reach meaningful conclusions based on the characteristics of the noise, a large collection of real-world datasets will be considered and different types of noise, present in real-world data, and several noise levels will be introduced into them, since these are usually unknown in real-world data. Two different types of noise, class and attribute noise, and four different schemes to introduce them will be considered. The experimentation will consist of a total of 1640 datasets. The results taken from these datasets will be analyzed taking into account two different factors: (i) the *performance*, and (ii) the *robustness*, i.e., the capability of the classifier to be insensitive to the increments in the noise level, of each method in each noisy dataset. The results obtained will also be contrasted using the proper statistical tests, as recommended in the specialized literature [16,17,10,11].

The choice of the single classification algorithms used to build the MCSs is based on their behavior with noisy data. In such a way, one is able to extract meaningful conclusions from the point of view of noise. Three algorithms have been selected, among the top ten algorithms in Data Mining [47], each one belonging to a different learning paradigm, and having a well-known differentiated robustness to noise: a *Support Vector Machine* (SVM) [5], C4.5 [36] and *k-Nearest Neighbors* (*k*-NN) [29].

All these classification algorithms will be combined using different decisions combination methods [28,38,41,18] to create MCSs of different sizes and characteristics. Two forms to create diversity will be considered: (i) considering different individual classifiers trained with the whole training data (heterogeneous base classifiers) and (ii) considering only one baseline classifier trained with different random samples of the training data of equal size as the original training data (using bagging). In this way, whether the performance and noise-robustness of the individual components is related to that of the corresponding MCS will be verified. All the conclusions and lessons learned from the analysis of the empirical results will be included in a specific section at the end of this paper.

A web-page with all the complementary material associated with this paper is available at http://www.sci2s.ugr.es/mcs_noise, including the basic information of this paper, all the datasets created and the complete results obtained for each classification algorithm, in such a way that this work becomes easily reproducible by other researchers.

The rest of this paper is organized as follows. Section 2 presents an introduction to classification with noisy data. Section 3 gives the motivations for the usage of MCSs. Next, Section 4 describes the experimental framework. Section 5 includes the experimental results and their analysis. Section 6 studies the results of different decisions combination methods. Section 7 presents the lessons learned and, finally, Section 8 presents some concluding remarks.

2. Noisy data in classification problems

First, this section introduces the problem of noisy data in classification (Section 2.1) and then, it describes how to simulate different types of noise present in real-world data (Section 2.2).

2.1. Introduction to noisy data

Data gathered from real-world problems are never perfect and often suffer from corruptions that may hinder the performance of the system in terms of the classification accuracy, building time, size and interpretability of the classifier [51]. Noise mainly affects the data acquisition and preprocessing phases, having two main sources [52]: implicit errors introduced by measurement tools, such as different types of sensors; and random errors introduced by batch processes or experts when the data are gathered, such as in a document digitalization process.

A large number of components determine the quality of a dataset [42]. Among them, the class labels and the attribute values directly influence the quality of a classification dataset. The quality of the class labels refers to whether the class of each example is correctly assigned; otherwise, the quality of the attributes refers to their capability of properly characterizing the examples for classification purposes – obviously, if noise affects attribute values, this capability of characterization and therefore, the quality of the attributes, is reduced. Based on these two information sources, two types of noise can be distinguished in a given dataset [46,3]:

1. **Class noise**. This occurs when an example is incorrectly labeled. Class noise can be attributed to several causes, such as subjectivity during the labeling process, data entry errors, or inadequacy of the information used to label each example. Two types of class noise can be distinguished: (i) *contradictory examples* [13] – duplicate examples have different class labels –, and (ii) *misclassifications* [53] – examples that are labeled as a class different from the real one.
2. **Attribute noise**. This refers to corruptions in the values of one or more attributes. Examples of attribute noise are: erroneous attribute values, missing or unknown attribute values, and incomplete attributes or "do not care" values.

In this paper, class noise refers to misclassifications, whereas attribute noise refers to erroneous attribute values, because they are the most common in real-world data [52]. Furthermore, erroneous attribute values, unlike other types of attribute noise, such as missing values (which are easily detectable), have received less attention in the literature.

Noise hinders the knowledge extraction from the data and spoils the models obtained using that noisy data when they are compared to the models learned from clean data from the same problem, which represent the real implicit knowledge of the problem [52]. In this sense, *robustness* [19] is the capability of an algorithm to build models that are insensitive to data corruptions and suffer less from the impact of noise; that is, the more robust an algorithm is, the more similar the models built from clean and noisy data are. Thus, a classification algorithm is said to be more robust than another if the former builds classifiers which are less influenced by noise than the latter. Robustness is considered more important than performance results when dealing with noisy data, because it allows one to know *a priori* the expected behavior of a learning method against noise in those cases where the characteristics of noise are unknown.

In order to analyze the degree of robustness of the classifiers in the presence of noise, we will compare the performance of the classifiers learned with the original (without induced noise) dataset with the performance of the classifiers learned using the noisy dataset. Therefore, those classifiers learned from noisy datasets that are more similar (in terms of results) to the noise-free[1] classifiers will be the most robust ones. An example of a robust learner is the C4.5 decision tree learning algorithm [36] considered in this paper, which uses pruning strategies to reduce the chances of trees being influenced by the noise in the training data [34,35]. One of the objectives of this paper is to check whether the performance and robustness of C4.5 trained with noisy data can be improved by incorporating it into an MCS.

2.2. Simulating the noise of real-world datasets

Checking the effect of noisy data on the performance of classifier learning algorithms is necessary to improve their reliability and has motivated the study of how to generate and introduce noise into the data. Noise generation can be characterized by three main characteristics [52]:

1. **The place where the noise is introduced.** Noise may affect the input attributes or the output class, impairing the learning process and the resulting model.
2. **The noise distribution.** The way in which the noise is present can be, for example, uniform [39,54] or Gaussian [53,52].
3. **The magnitude of generated noise values.** The extent to which the noise affects the dataset can be relative to each data value of each attribute, or relative to the minimum, maximum and standard deviation for each attribute [53,54,52].

[1] Noise-free refers to the original datasets without induced noise, which might contain noise that is not quantifiable.

In contrast to other studies in the literature, this paper aims to clearly explain how noise is defined and generated (see Section 4.2 for more details), and also to properly justify the choice of the noise introduction schemes. Furthermore, the noise generation software has been incorporated into the KEEL tool [2] for its free usage. The two types of noise considered in this work, class and attribute noise, have been modeled using four different noise schemes; in such a way, the presence of those types of noise will allow one to simulate the behavior of the classifiers in these two scenarios:

1. **Class noise** usually occurs on the boundaries of the classes, where the examples may have similar characteristics – although it can occur in any other area of the domain. In this paper, class noise is introduced using an *uniform class noise scheme* [39] (randomly corrupting the class labels of the examples) and a *pairwise class noise scheme* [53,52] (labeling examples of the majority class with the second majority class). Considering these two schemes, noise affecting any pair of classes and only the two majority classes are simulated, respectively.
2. **Attribute noise** can proceed from several sources, such as transmission constraints, faults in sensor devices, irregularities in sampling and transcription errors [40]. The erroneous attribute values can be totally unpredictable, i.e., random, or imply a low variation with respect to the correct value. We use the *uniform attribute noise scheme* [54,52] and the *Gaussian attribute noise scheme* in order to simulate each one of the possibilities, respectively. We introduce attribute noise in accordance with the hypothesis that interactions between attributes are weak [52]; as a consequence, the noise introduced into each attribute has a low correlation with the noise introduced into the rest.

3. Multiple Classifier Systems for classification tasks

This section focuses on describing the usage of MCSs for classification tasks. Section 3.1 presents the motivations for the usage of MCSs, whereas Section 3.2 describes the different ways usually considered to build an MCS. Next, Section 3.3 shows different methods to combine the outputs of the classifiers. Finally, Section 3.4 explains how the MCSs of this paper are built.

3.1. Single classifiers against their combination

Given a set of problems, finding the best overall classification algorithm is sometimes difficult because some classifiers may excel in some cases and perform poorly in others. Moreover, even though the optimal match between a learning method and a problem is usually searched, this match is generally difficult to achieve and perfect solutions are rarely found for complex problems [14,15]. This is a reason for using MCSs [15,14,32], since it is not necessary to choose a specific learning method. All of them might be used, taking advantage of the strengths of each method, while avoiding its weaknesses. Furthermore, there are other motivations to combine several classifiers [14]:

- To avoid the choice of some arbitrary but important initial condition, e.g., those involving the parameters of the learning method.
- To introduce some randomness to the training process in order to obtain different alternatives that can be combined to improve the results obtained by the individual classifiers.
- To use complementary classification methods improves dynamic adaptation and flexibility.

3.2. Using several classifiers for a classification problem

There are several strategies to use more than one classifier for a single classification task [15]:

- *Dynamic classifier selection*. This is based on the fact that one classifier may outperform all others using a global performance measure but it may not be the best in all parts of the domain. Therefore, this type of methods divide the input domain into several parts and aim to select the classifier with the best performance in that part.
- *Multi-stage organization*. This builds the classifiers iteratively. At each iteration, a group of classifiers operates in parallel and their decisions are then combined. A dynamic selector decides which classifiers are to be activated at each stage based on the classification performances of each classifier in previous stages.
- *Sequential approach*. A classifier is used first and the other ones are used only if the first does not yield a decision with sufficient confidence.
- *Parallel approach*. All available classifiers are used for the same input example in parallel. The outputs from each classifier are then combined to obtain the final prediction.

Although the first three approaches have been explored to a certain extent, the majority of classifier combination research focuses on the fourth approach, due to its simplicity and the fact that it enables one to take advantage of the factors presented in the previous section. For these reasons, this paper focus on the fourth approach.

3.3. Decisions combination in multiple classifiers systems

As has been previously mentioned, parallel approaches need a posterior phase of combination after the evaluation of a given example by all the classifiers. Many decisions combination proposals can be found in the literature, such as the intersection of decision regions [12], voting methods [28], prediction by top choice combinations [43], use of the Dempster–Shafer theory [26,49] or ranking methods [15]. In concrete, we will study the following four combination methods for the MCSs built with heterogeneous classifiers:

1. **Majority vote (MAJ)** [28]. This is a simple but powerful approach, where each classifier gives a vote to the predicted class and the most voted one is chosen as the output.
2. **Weighted majority vote (W-MAJ)** [38]. Similarly to MAJ, each classifier gives a vote for the predicted class, but in this case, the vote is weighted depending on the competence (accuracy) of the classifier in the training phase.
3. **Naive Bayes (NB)** [41]. This method assumes that the base classifiers are mutually independent. Hence, the predicted class is that one obtaining the highest posterior probability. In order to compute these probabilities, the confusion matrix of each classifier is considered.
4. **Behavior-Knowledge Space (BKS)** [18]. This is a multinomial method that indexes a cell in a look-up table for each possible combination of classifiers outputs. A cell is labeled with the class to which the majority of the instances in that cell belong to. A new instance is classified by the corresponding cell label; in case that for the cell is not labeled or there is a tie, the output is given by MAJ.

We always use the same training dataset to train all the base classifiers and to compute the parameters of the aggregation methods, as it is recommended in [23]. Using a separate set of examples to obtain such parameters can imply some important training data to be ignored and this fact is generally translated into a loss of accuracy of the final MCS built.

In MCSs built with heterogeneous classifiers, all of them may not return a confidence. Even though each classifier can be individually modified to return a confidence for its predictions, such confidences will come from different computations depending on the classifier adapted and their combination could become meaningless. Nevertheless, in MCSs built with the same type of classifier, this fact does not occur and it is possible to combine their confidences since these are homogeneous among all the base classifiers [23]. Therefore, in the case of Bagging, given that the same classifier is used to train all the base classifiers, the confidence of the prediction can be used to compute a weight and, in turn, these weights can be used in a weighted voting combination scheme.

3.4. Multiple Classifier Systems used in the experimentation

The choice of the learning algorithms used in this paper – SVM [5], C4.5 [36] and *k*-NN [29] – is based on their good behavior in a large number of real-world problems; moreover, they were selected because these methods have a highly differentiated and well known noise-robustness, which is important in order to properly evaluate the performance of MCSs in the presence of noise. This section briefly describes their operating procedures, with special reference to their noise handling mechanisms, allowing one to better understand the results of the impact of noise on them in the experimentation (Section 5).

- **C4.5 decision tree generator** [36]. C4.5 is considered a robust learner tolerant to noisy data. It iteratively builds a decision tree that correctly classifies the largest number of examples (defined by the *window size*). As indicated in [9], the main problem of *windowing* techniques is that the process may incorporate all noisy examples into the learning *window*, because they may be misclassified by an apparently good rule. In order to avoid this problem as well as over-fitting to noisy data, C4.5 uses pruning strategies to reduce the chances of classifiers being affected by noisy examples from the training data [34,35]. For example, in this paper, a post-pruning process is carried out, which discards unreliable parts from the fully grown decision tree. Nevertheless, this process is based on statistical assumptions that might be unreliable, since they depend on the training data. Therefore, when the noise level is relatively high, even a robust learner such as C4.5 may obtain a poor performance.
- **Support Vector Machine** [5]. SVM builds a hyperplane that separates examples of different classes maximizing the distance, that is, the margin, of the examples lying near the separating hyperplane, which are called support vectors. A better generalization is generally achieved when the distances from the examples of both classes to the hyperplane are larger, because the maximization of the margin of separation reduces the empirical risk instead of the expected risk. SVM relies on the support vectors, which are identified from the training instances, to derive the decision model. Thus, the hyperplanes found by SVM can be easily altered including or excluding a single noisy example [31]. Furthermore, the implicit interdependence among the input attributes – since they are fused into two factors in the learning phase – may also cause difficulties when noise is introduced into the training data, which disrupts the interrelations and correlations between the attributes. Thus, SVM should *a priori* be more noise-sensitive than C4.5. We have used the Puk kernel for SVM, whose expression is given in Eq. (1):

$$K(x_i, x_j) = \frac{1}{\left[1 + \left(\frac{2\sqrt{\|x_i - x_j\|^2}\sqrt{2^{(1/\omega)} - 1}}{\sigma}\right)^2\right]^{\omega}}, \tag{1}$$

where x_i and x_j are two vectors, and σ and ω control the half-width and the tailing factor of the peak of the function.

- ***k*-Nearest Neighbors** [29]. This method searches the k most similar examples to the given one using a distance function in order to determine its class. The function used in this paper is the Heterogeneous Value Distance Metric (HVDM), which computes the distance between the values of each attribute based on the training data distribution. In the experimentation, three different values of k are considered: 1, 3 and 5. The value of k determines a higher or lower sensitivity to noise [22]. Thus, if only one nearest neighbor is considered, k-NN relies on each single example to derive the decision model and the decision regions can be easily altered by individual examples. Otherwise, higher k values make the model more robust. In order to find the k nearest neighbors in the space of numerical and nominal attributes, the Heterogeneous Value Difference Metric (HVDM) metric is applied [44].

Considering the previous classifiers (SVM, C4.5 and k-NN), the following MCS are built:

1. **MCS3**-*k*. These MCSs are composed by 3 individual classifiers (SVM, C4.5 and k-NN). Three values of k are considered (1, 3 and 5), so we will create three different MCS3-*k* denoted as MCS3-1, MCS3-3 and MCS3-5.
2. **MCS5.** This is composed by 5 different classifiers: SVM, C4.5 and k-NN with the 3 different values (1, 3 and 5).
3. **BagC4.5.** This MCS considers C4.5 as baseline classifier. In this case, for the decisions combination method we use as confidence the accuracy of the leaf predicting the class, which is computed as the percentage of correctly classified train examples from the total number of covered ones.

Therefore, the MCSs built with heterogeneous classifiers (MCS3-*k* and MCS5) will contain a noise-robust algorithm (C4.5), a noise-sensitive method (SVM) and a method whose robustness varies depending on the value of k (k-NN). Hence, how the behavior of each MCS3-*k* and the MCS5 depends on that of each single method can be analyzed. In the case of BagC4.5, it is compared to its base classifier, that is, C4.5.

The configuration parameters used to build the classifiers considered are shown in Table 1. The same parameter setup for each classification algorithm, considered either as part of a MCS or individually, has been used for all the datasets. This will help us to know to what extent the same classifier trained with the same parameters over the same training data can imply an advantage incorporating it into a MCS with respect to its usage alone.

4. Experimental framework

First, this section describes the base datasets used in the experiments (Section 4.1). Second, the processes to induce noise into them are introduced (Section 4.2). Finally, the methodology for the analysis of the results is explained in Section 4.3.

4.1. Base datasets

The experimentation is based on forty real-world classification problems from the KEEL-dataset repository[2] [1]. Table 2 shows the datasets sorted by the number of classes (#CL). Moreover, for each dataset, the number of examples (#EX) and the number of attributes (#AT), along with the number of numeric and nominal attributes are presented. Some of the largest datasets (*nursery*, *page-blocks*, *penbased*, *satimage*, *splice*) were stratified at 10% in order to reduce the computational time required for training, given the large amount of executions carried out in this paper. For datasets containing missing values (such as *automobile* or *dermatology*), instances with missing values were removed from the datasets before the partitioning.

4.2. Introducing noise into datasets

In the previous datasets, as in most of the real-world datasets, the initial amount and type of noise present are unknown. Therefore, no assumptions about the base noise type and level can be made. For this reason, these datasets are considered to be noise free, in the sense that no recognizable noise has been induced into them. In order to control the amount of noise in each dataset and check how it affects the classifiers, noise is introduced into each dataset in a supervised manner. Four different noise schemes proposed in the literature, as explained in Section 2, are used in order to introduce a noise level $x\%$ into each dataset:

1. **Introduction of class noise.**
   - **Uniform class noise** [39]. $x\%$ of the examples are corrupted. The class labels of these examples are randomly replaced by another one from the M classes.

[2] http://www.keel.es/datasets.php.

Table 1
Parameter configuration for the classification algorithms.

SVM	C4.5	k-NN
• Cost: $C = 100$, Tolerance: $t = 0.001$ • Parameter for the round-off error: $\epsilon = 10^{-12}$ • Type of Kernel: *Puk* with $\sigma = 1$, $\omega = 1$ • Data preprocessing: *Normalization in* $[0,1]$ • Fit logictics models to the output	• Confidence level: $c = 0.25$ • Minimal instances per leaf: $i = 2$ • Prune after the tree building	• Number of neighbors: $k = 1,3$ and 5 • Distance function: *HVDM*

Table 2
Summary description for the classification datasets.

Dataset	#EX	#AT	#CL	Dataset	#EX	#AT	#CL	Dataset	#EX	#AT	#CL	Dataset	#EX	#AT	#CL
banana	5300	2(2/0)	2	spambase	4597	57(57/0)	2	hayes-roth	160	4(4/0)	3	glass	214	9(9/0)	7
german	1000	20(13/7)	2	twonorm	7400	20(20/0)	2	car	1728	6(0/6)	4	shuttle	2175	9(9/0)	7
heart	270	13(13/0)	2	wdbc	569	30(30/0)	2	lymphography	148	18(3/15)	4	zoo	101	16(0/16)	7
ionosphere	351	33(33/0)	2	balance	625	4(4/0)	3	vehicle	846	18(18/0)	4	satimage	643	36(36/0)	7
magic	19020	10(10/0)	2	splice	319	60(0/60)	3	nursery	1296	8(0/8)	5	segment	2310	19(19/0)	7
monk	432	6(6/0)	2	contraceptive	1473	9(9/0)	3	page-blocks	548	10(10/0)	5	ecoli	336	7(7/0)	8
phoneme	5404	5(5/0)	2	iris	150	4(4/0)	3	cleveland	297	13(13/0)	5	led7digit	500	7(0/7)	10
pima	768	8(8/0)	2	new-thyroid	215	5(5/0)	3	automobile	159	25(15/10)	6	penbased	1099	16(16/0)	10
ring	7400	20(20/0)	2	thyroid	720	21(6/15)	3	dermatology	358	33(1/32)	6	yeast	1484	8(8/0)	10
sonar	208	60(60/0)	2	wine	178	13(13/0)	3	flare	1066	11(0/11)	6	vowel	990	13(13/0)	11

- **Pairwise class noise** [53,52]. Let X be the majority class and Y the second majority class, an example with the label X has a probability of $x/100$ of being incorrectly labeled as Y.

2. **Introduction of attribute noise.**
   - **Uniform attribute noise** [54,52]. x% of the values of each attribute in the dataset are corrupted. To corrupt each attribute A_i, x% of the examples in the data set are chosen, and their A_i value is assigned a random value from the domain $\mathbb{D}_i$ of the attribute A_i. An uniform distribution is used either for numerical or nominal attributes.
   - **Gaussian attribute noise**. This scheme is similar to the uniform attribute noise, but in this case, the A_i values are corrupted, adding a random value to them following a Gaussian distribution of $mean = 0$ and $standard\ deviation = (max - min)/5$, being max and min the limits of the attribute domain $(\mathbb{D}_i)$. Nominal attributes are treated as in the case of the uniform attribute noise.

In order to create a noisy dataset from the original one, the noise is introduced into the training partitions as follows:

1. A level of noise x%, of either class noise (uniform or pairwise) or attribute noise (uniform or Gaussian), is introduced into a copy of the full original dataset.
2. Both datasets, the original one and the noisy copy, are partitioned into 5 equivalent folds, that is, with the same examples in each one.
3. The training partitions are built from the noisy copy, whereas the test partitions are formed from examples from the base dataset, that is, the noise free dataset.

We introduce noise, either class or attribute noise, only into the training sets since we want to focus on the effects of noise on the training process. This will be carried out observing how the classifiers built from different noisy training data for a particular dataset behave, considering the accuracy of those classifiers, with the same clean test data. Thus, the accuracy of the classifier built over the original training set without additional noise acts as a reference value that can be directly compared with the accuracy of each classifier obtained with the different noisy training data. Corrupting the test sets also affects the accuracy obtained by the classifiers and therefore, our conclusions will be not only limited to the effects of noise on the training process.

The accuracy estimation of the classifiers in a dataset is obtained by means of 5 runs of a stratified 5-fold cross-validation. Hence, a total of 25 runs per dataset, noise type and level are averaged. 5 partitions are used because, if each partition has a large number of examples, the noise effects will be more notable, facilitating their analysis.

As a consequence, a large collection of new noisy datasets are created from the aforementioned forty base datasets. Both types of noise are independently considered: class and attribute noise. For each type of noise, the noise levels ranging from $x = 0\%$ (base datasets) to $x = 50\%$, by increments of 5%, are studied. Therefore, 400 noisy datasets are created with each of the four aforementioned noise schemes. The total number of datasets in the experimentation is 1640. Hence, considering the 5x5fcv of the 1640 datasets, 41000 executions are carried out for each classification algorithm. All these datasets are available on the web-page associated with this paper.

4.3. Methodology of analysis

In order to check the behavior of the different methods when dealing with noisy data, the results of each MCS are compared with those of their individual components using two distinct properties:

1. The performance of the classification algorithms on the test sets for each level of induced noise, defined as its accuracy rate. For the sake of brevity, only averaged results are shown (the rest can be found on the web page associated with this paper), but it must be taken into account that the conclusions drawn in this paper are based on the proper statistical analysis, which considers all the results (not averaged).
2. The robustness of each method is estimated with the *relative loss of accuracy* (RLA) (Eq. (2)), which is used to measure the percentage of variation of the accuracy of the classifiers at a concrete noise level with respect to the original case with no additional noise:

$$RLA_{x\%} = \frac{Acc_{0\%} - Acc_{x\%}}{Acc_{0\%}}, \tag{2}$$

where $RLA_{x\%}$ is the relative loss of accuracy at a noise level x%, $Acc_{0\%}$ is the test accuracy in the original case, that is, with 0% of induced noise, and $Acc_{x\%}$ is the test accuracy with a noise level x%.

In order to properly analyze the performance and RLA results, Wilcoxon's signed rank statistical test is used, as suggested in the literature [6]. This is a non-parametric pairwise test that aims to detect significant differences between two sample means; that is, between the behavior of the two algorithms involved in each comparison. For each type and noise level, the MCS and each single classifier will be compared using Wilcoxon's test and the p-values associated with these comparisons will be obtained. The p-value represents the lowest level of significance of a hypothesis that results in a rejection and it allows one to know whether two algorithms are significantly different and the degree of this difference. Both performance and robustness are studied because the conclusions reached with one of these metrics need not imply the same conclusions with the other.

5. Analysis of the results: Noisy data in Multiple Classifier Systems

This section presents the results of performance and robustness of the MCSs trained with noisy data with respect to their individual classifier components. Section 5.1 analyzes the results obtained with class noise, whereas Section 5.2 is devoted to the attribute noise. Each one of these sections is divided into two main parts: the first part describes the results obtained comparing each MCS with each of its components, whereas the second one analyzes these results for each type of noise. Due to the large amount of results, they should be properly outlined. As a consequence, only the general behavior of the methods will be described in order to reach meaningful conclusions (without observing the characteristics of each single dataset). This section is devoted to the study of the MAJ scheme for the final class prediction in order to consider its results as a reference, since it is the simplest method; other decisions combination schemes are analyzed in Section 6.

5.1. First scenario: Datasets with class noise

Table 3 shows the performance (top part of the table) and robustness (bottom part of table) results of each classification algorithm at each noise level on datasets with class noise. Each one of these parts in the table (performance and robustness parts) is divided into other two parts: one with the results of the uniform class noise and another with the results of the pairwise class noise. A star '$*$' next to a p-value indicates that the corresponding single algorithm obtains more ranks than the MCS in Wilcoxon's test comparing the individual classifier and the MCS. Note that the robustness can only be measured if the noise level is higher than 0%, so the robustness results are presented from a noise level of 5% and higher.

The performance results in this table are summarized below:

- **Performance results with uniform class noise**.
  1. **Results of MCS3-*k***.
     - **MCS3-*k* vs. SVM**. MCS3-k is statistically better than SVM regardless of the value of k. Moreover, the higher the value of k, the lower the p-values are.
     - **MCS3-*k* vs. C4.5**. The performance of MCS3-k with respect to C4.5 heavily depends on the value of k. In MCS3-1 statistical differences are only found at the lowest noise levels (up to 5%), whereas MCS3-3 maintains these differences up to 30%. For the rest of the noise levels, both MCS3-1 and MCS3-3 are statistically equivalent to C4.5, although MCS3-3 generally obtains more ranks than C4.5. Finally, MCS3-5 is statistically better than all its individual components at all noise levels.
     - **MCS3-*k* vs. *k*-NN**. Statistical differences are found between MCS3-k and k-NN regardless of the value of k. Furthermore, as the value of k increases, so does the corresponding p-value at the different noise levels (note that very low p-values are obtained in the case of $k = 1$).

Table 3
Performance and robustness results on datasets with class noise.

x%	Results										*p*-values														
	Single methods					Multi-classifiers					MCS3-1 vs.			MCS3-3 vs.			MCS3-5 vs.			MCS5 vs.					BagC4.5 vs.
	SVM	C4.5	1-NN	3-NN	5-NN	MCS3-1	MCS3-3	MCS3-5	MCS5	BagC4.5	SVM	C4.5	1-NN	SVM	C4.5	3-NN	SVM	C4.5	5-NN	SVM	C4.5	1-NN	3-NN	5-NN	C4.5
Performance																									
Uniform class noise																									
0%	83.25	82.96	81.42	82.32	82.32	85.42	85.48	85.52	83.74	85.18	5.2E−03	1.8E−03	7.1E−04	1.3E−02	5.0E−04	3.1E−03	1.0E−02	2.8E−04	5.0E−03	4.4E−01	1.9E−02	1.4E−04	9.7E−06	2.2E−03	1.5E−07
5%	81.04	82.48	78.68	81.67	82.06	84.11	84.86	85.03	83.15	85.10	2.6E−04	3.0E−02	1.3E−06	2.4E−05	6.1E−04	7.8E−04	5.5E−06	2.2E−04	4.6E−03	6.6E−03	1.6E−02	4.2E−08	2.7E−06	3.5E−02	6.6E−08
10%	79.58	82.08	76.28	80.84	81.56	83.00	84.25	84.57	82.60	84.87	1.1E−04	3.9E−01	1.3E−07	1.5E−06	3.5E−03	7.8E−04	2.1E−07	4.5E−04	1.1E−02	1.7E−03	4.8E−02	3.9E−08	1.1E−06	8.1E−02	1.3E−07
15%	78.26	80.88	73.71	79.71	80.94	81.60	83.21	83.63	81.74	84.20	1.5E−04	7.1E−01	6.1E−08	1.3E−06	3.1E−03	4.7E−04	1.7E−07	6.1E−04	5.3E−02	2.1E−03	4.5E−02	3.6E−08	1.3E−06	2.9E−01	1.3E−07
20%	76.55	79.97	71.22	78.18	80.09	80.09	81.98	82.69	80.55	83.58	5.8E−05	9.5E−01*	1.0E−07	8.1E−07	2.0E−02	2.7E−04	1.7E−07	8.6E−04	6.2E−02	8.6E−04	7.4E−02	4.5E−08	1.1E−06	8.5E−01	6.3E−07
25%	75.05	79.02	68.78	76.68	79.27	78.53	80.87	81.78	79.50	82.80	3.7E−04	4.7E−01*	7.0E−08	2.1E−06	4.8E−02	2.2E−04	1.0E−07	1.2E−03	1.2E−01	2.4E−03	5.3E−02	3.6E−08	2.2E−06	9.3E−01*	8.1E−07
30%	73.82	77.90	65.88	75.09	78.18	77.10	79.69	80.75	78.18	81.79	2.8E−04	3.5E−01*	6.1E−08	1.7E−06	8.3E−02	1.9E−04	2.3E−07	2.0E−03	1.0E−01	1.9E−03	3.2E−01	3.6E−08	5.2E−06	5.2E−01*	4.0E−06
35%	72.31	76.28	63.51	72.95	76.67	75.11	77.87	79.16	76.44	80.39	1.7E−03	2.1E−01*	6.5E−08	2.9E−06	1.5E−01	9.2E−05	5.3E−07	5.4E−03	1.0E−01	3.1E−03	5.0E−01	3.6E−08	4.9E−06	3.3E−01*	4.6E−06
40%	70.69	74.51	61.00	70.71	75.09	73.20	76.04	77.56	74.72	78.85	7.2E−03	2.0E−01*	7.0E−08	1.8E−05	1.8E−01	7.8E−05	2.1E−06	5.6E−03	9.0E−02	5.2E−03	4.9E−01	3.6E−08	1.6E−06	3.8E−01*	3.9E−05
45%	69.10	72.07	58.68	68.00	72.57	70.68	73.61	75.27	72.16	77.06	4.3E−02	2.0E−01*	1.4E−07	1.2E−04	1.7E−01	7.3E−05	3.1E−06	6.6E−03	7.8E−02	2.4E−02	5.0E−01	3.6E−08	1.8E−06	5.5E−01*	9.2E−05
50%	67.07	69.22	55.55	65.39	70.40	67.64	70.82	72.70	69.43	74.31	5.4E−01	1.4E−01*	1.1E−07	6.7E−04	1.6E−01	2.3E−04	2.3E−05	4.8E−03	1.4E−01	8.1E−02	7.1E−01	3.6E−08	5.5E−06	2.4E−01*	1.3E−04
Pairwise class noise																									
0%	83.25	82.96	81.42	82.32	82.32	85.42	85.48	85.52	83.74	85.18	5.2E−03	1.8E−03	7.1E−04	1.3E−02	5.0E−04	3.1E−03	1.0E−02	2.8E−04	5.0E−03	4.4E−01	1.9E−02	1.4E−04	9.7E−06	2.2E−03	1.5E−07
5%	81.77	82.59	79.58	81.83	82.10	84.75	85.19	85.28	83.54	84.97	1.6E−04	1.0E−02	2.1E−06	1.2E−04	4.3E−04	7.4E−04	9.7E−05	1.7E−04	4.6E−03	2.7E−02	1.1E−02	1.6E−07	9.3E−07	1.8E−03	3.0E−07
10%	80.74	82.17	77.73	80.87	81.61	83.95	84.61	84.86	82.99	84.62	1.2E−04	5.8E−02	1.2E−07	6.2E−05	2.3E−03	4.3E−05	2.0E−05	3.9E−04	2.8E−03	1.7E−02	3.5E−02	6.1E−08	5.3E−07	6.1E−03	2.1E−07
15%	79.72	81.54	75.99	79.60	80.84	83.09	83.89	84.29	82.23	83.75	7.3E−05	1.6E−01	7.0E−08	2.0E−05	5.6E−03	4.9E−06	7.1E−06	5.2E−04	6.4E−04	1.2E−02	9.0E−02	4.5E−08	2.8E−07	5.2E−03	1.2E−06
20%	79.11	80.87	74.25	77.81	79.33	82.21	83.06	83.50	80.99	83.01	2.0E−04	3.5E−01	8.2E−08	3.3E−05	1.9E−02	3.8E−07	1.2E−05	1.8E−03	6.7E−06	4.4E−02	4.3E−01	4.5E−08	2.3E−07	2.9E−04	2.5E−06
25%	77.80	79.59	72.27	75.75	77.52	80.81	81.64	82.14	79.54	81.46	3.3E−04	4.5E−01	6.5E−08	9.7E−05	7.6E−02	3.3E−07	2.6E−05	2.8E−03	2.7E−06	8.5E−02	6.6E−01	4.5E−08	1.7E−07	1.1E−04	1.1E−04
30%	76.64	78.81	70.46	73.45	75.19	79.52	80.24	80.76	77.48	79.74	2.1E−03	8.2E−01	5.6E−08	2.7E−04	4.0E−01	2.0E−07	6.9E−05	8.8E−02	7.1E−07	5.5E−01	3.3E−01*	4.2E−08	1.4E−07	2.2E−05	5.0E−02
35%	75.17	76.90	68.88	71.05	72.64	77.65	78.13	78.66	75.20	77.26	2.2E−02	8.6E−01	1.0E−07	1.4E−02	6.2E−01	9.5E−08	3.4E−03	2.4E−01	4.6E−07	6.0E−01*	1.4E−01*	1.2E−07	4.5E−08	1.8E−05	4.0E−01
40%	73.13	74.83	66.58	68.17	69.25	75.25	75.61	75.93	72.21	74.15	3.8E−02	8.0E−01*	4.5E−08	1.9E−02	9.7E−01*	1.3E−07	1.4E−02	5.8E−01	3.0E−07	2.8E−01*	6.2E−02*	1.4E−07	1.5E−07	5.5E−06	4.3E−01*
45%	70.38	71.18	64.95	65.58	66.05	71.82	71.97	72.01	68.85	69.95	2.7E−01	7.5E−01	1.1E−07	2.2E−01	6.5E−01	2.0E−07	1.8E−01	5.1E−01	4.0E−07	1.5E−01*	2.3E−01*	8.7E−07	1.5E−07	1.4E−06	4.7E−01*
50%	65.92	60.29	63.06	62.71	62.42	64.46	64.23	64.00	63.67	62.40	2.6E−02*	2.6E−05	1.1E−01	1.8E−02*	3.3E−05	7.8E−02	1.1E−02*	6.6E−05	1.2E−01	7.6E−02*	4.1E−04	5.2E−01	2.5E−02	3.7E−02	2.4E−03
Robustness																									
Uniform class noise																									
5%	2.72	0.58	3.35	0.84	0.33	1.59	0.73	0.61	0.76	0.07	3.2E−03	2.3E−05*	3.3E−07	9.2E−06	2.3E−01*	3.4E−01	3.5E−06	9.5E−01	1.8E−01*	8.7E−05	6.1E−01*	3.6E−08	6.2E−03	4.5E−04*	1.2E−02
10%	4.44	1.10	6.16	1.85	0.95	2.91	1.47	1.12	1.43	0.36	7.2E−03	1.5E−06*	1.0E−07	4.6E−05	9.3E−02*	1.5E−02	3.1E−06	7.1E−01	2.7E−01*	1.2E−04	4.8E−01*	3.6E−08	1.3E−03	3.3E−04*	3.3E−04
15%	6.07	2.64	9.13	3.07	1.67	4.59	2.72	2.31	2.35	1.20	3.6E−02	2.8E−04*	9.3E−07	9.7E−05	6.9E−01*	3.8E−02	1.9E−05	3.4E−01	1.9E−01*	6.7E−04	4.7E−01	4.2E−08	1.0E−04	1.9E−03*	2.3E−04
20%	8.16	3.78	12.10	4.88	2.64	6.40	4.24	3.47	3.85	1.94	1.5E−02	3.1E−05*	1.2E−06	1.1E−04	2.6E−01*	4.4E−02	2.3E−05	3.8E−01	2.8E−01*	2.7E−04	7.4E−01	5.2E−08	5.5E−04	4.1E−04*	3.9E−04
25%	9.89	5.00	14.97	6.84	3.70	8.26	5.54	4.52	5.10	2.93	7.4E−02	1.1E−05*	1.2E−06	2.6E−04	2.5E−01*	3.5E−02	2.7E−05	3.7E−01	5.0E−01*	9.0E−04	9.4E−01*	3.9E−08	5.2E−05	1.4E−03*	2.4E−04
30%	11.38	6.36	18.71	8.60	4.97	9.95	6.92	5.78	6.72	4.08	8.8E−02	4.6E−05*	1.8E−07	1.5E−04	2.8E−01*	2.8E−02	8.6E−06	3.8E−01	5.9E−01*	1.2E−03	4.7E−01*	3.6E−08	1.8E−03	6.7E−04*	8.4E−03
35%	13.16	8.39	21.26	10.99	6.65	12.33	9.10	7.72	8.67	5.83	3.3E−01	3.3E−05*	6.6E−07	5.5E−04	2.6E−01*	3.3E−02	1.3E−04	4.3E−01	9.8E−01*	4.8E−03	5.5E−01*	3.9E−08	7.1E−04	5.8E−04*	4.6E−03
40%	15.08	10.54	24.15	13.59	8.31	14.54	11.20	9.55	10.56	7.60	6.6E−01	8.2E−05*	1.5E−06	2.8E−03	3.0E−01*	9.9E−03	1.7E−04	2.5E−01	9.7E−01*	1.1E−02	6.1E−01*	3.9E−08	1.1E−04	1.4E−03*	5.0E−03
45%	17.07	13.49	27.07	16.96	11.54	17.56	14.10	12.30	13.86	9.77	8.7E−01*	9.7E−05*	8.1E−06	1.2E−02	3.5E−01*	1.2E−02	3.5E−04	3.8E−01	8.6E−01	6.0E−02	6.2E−01*	5.2E−08	2.2E−04	1.4E−02*	7.8E−03
50%	19.47	17.00	30.58	19.56	13.49	21.08	17.41	15.35	16.81	12.99	1.9E−01*	1.3E−04*	1.3E−05	5.1E−02	4.9E−01*	2.1E−02	1.6E−03	1.5E−01	8.4E−01*	1.5E−01	7.1E−01*	4.2E−08	8.2E−04	5.9E−03*	8.8E−03
Pairwise class noise																									
5%	1.72	0.47	2.12	0.58	0.18	0.79	0.34	0.28	0.17	0.27	2.8E−02	1.3E−02*	8.7E−07	9.4E−04	8.0E−01	4.1E−02	7.8E−04	5.5E−01	6.7E−01*	9.4E−04	2.4E−01	1.2E−07	9.9E−05	7.6E−01*	5.4E−01
10%	2.97	1.00	4.20	1.59	0.78	1.73	1.05	0.81	0.76	0.69	6.6E−03	2.0E−04*	6.7E−06	3.9E−04	2.4E−01*	4.2E−03	6.6E−05	5.6E−01	4.6E−01	1.1E−03	8.6E−01	1.8E−07	4.6E−06	7.3E−01*	1.2E−01
15%	4.11	1.82	6.23	3.06	1.64	2.81	1.95	1.53	1.68	1.81	1.4E−02	1.7E−03*	2.1E−06	1.9E−04	1.3E−01*	1.6E−03	5.5E−05	5.1E−01	4.2E−02	2.7E−03	3.9E−01*	1.2E−07	4.0E−06	8.1E−01*	7.8E−01*
20%	4.86	2.66	8.21	5.10	3.33	3.86	2.97	2.51	3.08	2.67	7.2E−02	1.7E−03*	1.0E−05	3.7E−03	9.0E−02*	4.3E−04	4.1E−04	7.4E−01*	3.0E−03	6.8E−02	4.0E−02*	2.1E−07	3.3E−06	2.4E−01	7.2E−01*
25%	6.41	4.33	10.57	7.32	5.24	5.55	4.68	4.16	4.69	4.53	2.5E−01	2.1E−02*	6.7E−06	2.7E−02	4.2E−01*	4.1E−04	4.4E−03	7.5E−01	3.7E−03	3.5E−01	8.8E−02*	1.6E−07	9.9E−07	5.0E−02	3.1E−01*
30%	7.81	5.33	12.75	9.96	7.80	7.11	6.35	5.81	7.16	6.56	3.8E−01	2.7E−03*	6.3E−06	6.4E−02	7.4E−02*	7.3E−05	1.2E−02	3.6E−01*	8.6E−04	5.9E−01*	5.8E−04*	2.6E−07	4.6E−06	3.6E−02	5.0E−03*
35%	9.59	7.68	14.28	12.51	10.53	9.30	8.88	8.32	9.64	9.50	3.8E−01	6.9E−03*	6.6E−05	3.6E−01	1.5E−02*	1.1E−04	1.3E−01	1.1E−01*	3.9E−04	2.7E−01*	1.0E−03*	1.6E−05	8.1E−06	2.3E−02	3.2E−03*
40%	12.01	10.19	17.20	15.93	14.53	12.08	11.83	11.52	13.17	13.09	5.5E−01	7.8E−03*	4.4E−05	4.4E−01	5.6E−03*	5.5E−05	3.3E−01	1.2E−02*	6.9E−05	1.4E−01*	9.7E−05*	6.6E−05	8.6E−06	5.8E−04	1.2E−04*
45%	15.19	14.59	18.96	18.84	18.03	16.05	15.96	16.01	16.92	17.93	7.3E−01*	1.1E−01*	1.0E−03	8.1E−01*	1.2E−01*	3.3E−04	7.7E−01*	9.6E−02*	1.4E−04	8.8E−02*	1.7E−03*	1.9E−02	1.1E−04	2.7E−04	8.7E−05*
50%	20.30	26.70	21.18	21.94	22.11	24.13	24.54	24.90	22.70	26.41	1.4E−04*	2.6E−03	1.1E−01*	5.8E−05*	5.9E−03	7.8E−01*	3.1E−05*	2.4E−02	9.4E−01	4.1E−02*	1.6E−02	8.8E−03*	7.4E−01*	5.8E−01	8.4E−01

2. **Results of MCS5.** MCS5 statistically outperforms SVM (except without induced noise) and C4.5 (up to 25%; they are equivalent at the rest of the noise levels). Statistical differences are also found with 1-NN and 3-NN at all the noise levels and with 5-NN at the lowest noise levels (below of 15%; both are equivalent at the rest of the noise levels). The *p*-values increase with larger values of *k*.
3. **Results of BagC4.5.** BagC4.5 is statistically better than C4.5 at all the noise levels.

- **Performance results with pairwise class noise**.
  1. **Results of MCS3-***k*.
     - **MCS3-*k* vs. SVM**. MCS3-*k* statistically outperforms its individual components when the noise level is below 45%, whereas it only performs statistically worse than SVM when the noise level reaches 50% (regardless of the value of *k*). The *p*-values at the different noise levels decrease as *k* increases.
     - **MCS3-*k* vs. C4.5**. MCS3-*k* obtains more ranks than C4.5 in most of the cases; moreover, it is statistically better than C4.5 when the noise level is below 15% ($k = 1$), 30% ($k = 3$) and 35% ($k = 5$).
     - **MCS3-*k* vs. *k*-NN**. MCS3-*k* statistically outperforms *k*-NN regardless of the value of *k*. The *p*-values at the different noise levels increase with larger values of *k*.

  2. **Results of MCS5.** In general, MCS5 is statistically better than SVM up to 25% and C4.5 up to 15%; there are not differences at the rest of the noise levels. Statistical differences are also found with *k*-NN regardless the value of *k* (the *p*-values increase with larger values of *k*).
  3. **Results of BagC4.5.** BagC4.5 statistically outperform C4.5 up to a 30% of noise level.

Hereafter, the robustness results obtained are summarized:

- **Robustness results with uniform class noise**.
  1. **Results of MCS3-*k***.
     - **MCS3-*k* vs. SVM**. MCS3-1 is significantly more robust than SVM up to a noise level of 30%. Both are equivalent from 35% onwards-even though MCS3-1 obtains more ranks at 35–40% and SVM at 45–50%. In the cases of $k = 3$ and $k = 5$, MCS3-*k* statistically outperforms SVM.
     - **MCS3-*k* vs. C4.5**. The robustness of C4.5 excels with respect to MCS3-1, observing the differences found. Regarding MCS3-3 and MCS3-5, they are equivalent to C4.5-although C4.5 obtains more ranks with $k = 3$ and MCS3-*k* with $k = 5$.
     - **MCS3-*k* vs. *k*-NN**. MCS3-*k* is statistically better than *k*-NN (with $k = 1$ and $k = 3$, even though the *p*-values for $k = 3$ are higher than those for $k = 1$). Otherwise, MCS3-5 is statistically equivalent to 5-NN at all the noise levels.

  2. **Results of MCS5.** In general, MCS5 is statistically more robust than SVM, 1-NN and 3-NN, equivalent to C4.5 and less robust than 5-NN.
  3. **Results of BagC4.5.** BagC4.5 statistically outperforms C4.5.
- **Robustness results with pairwise class noise**.
  1. **Results of MCS3-*k***.
     - **MCS3-*k* vs. SVM**. MCS3-1 statistically overcomes SVM up to a 20% noise level, they are equivalent up to 45% and MCS3-1 is outperformed by SMV at 50%. Similar behavior is shown with MCS3-3 and MCS3-5, but the statistical differences in favor of the MCS3-*k* remain up to 30%.
     - **MCS3-*k* vs. C4.5**. C4.5 is statistically more robust than MCS3-1 (except in highly affected datasets, 45–50%). MCS3-3 and C4.5 are equally robust in most of the cases, even though C4.5 excels at some noise levels. No differences are found between MCS3-5 and C4.5, obtaining larger *p*-values than with the other values of *k*.
     - **MCS3-*k* vs. *k*-NN**. The superiority of MCS3-*k* against *k*-NN is notable, as it is statistically better at all noise levels; in addition, the values increase with the value of *k*.

  2. **Results of MCS5.** MCS5 is statistically more robust than SVM up to 20% of noise level, equivalent up to 40% and worse at 45–50%. It statistically outperforms C4.5 up to 15% and, in general, it is worse onwards. Statistical differences with 1-NN and 3-NN are found (except at the highest noise level). MCS5 is also statistically better than 5-NN between the noise levels 25–45% and it is equivalent at the rest of the noise levels.
  3. **Results of BagC4.5.** BagC4.5 is equivalent up to a noise level of 25% and worse at the rest of the noise levels.

From these performance and robustness results, the following points can be concluded:

1. The uniform scheme is the most disruptive class noise for the majority of the classifiers (single classifiers, MCS3-*k* and MCS5) but for BagC4.5, which is more affected by the pairwise noise. However, from a medium noise level (15–25%) onwards, the pairwise class noise becomes more disruptive for 3-NN and 5-NN (and consequently for MCS3-3, MCS3-5 and also for MCS5). This fact clearly indicates that the behavior of each single classification algorithm trained with noisy data influences that of the corresponding MCS into which it is incorporated (in the case of the MCSs built with

heterogeneous classifiers). In addition, the sampling mechanism used to select the training examples can also affect the results (in the case of BagC4.5). The reasons of the behavior of each type of MCS (built with heterogeneous classifiers and with bagging) with both kinds of class noise are given in the following points:

- The higher disruptiveness of the uniform class noise in MCSs built with heterogeneous classifiers can be attributed to two main reasons: (i) this type of noise affects all the output domain, that is, all the classes, to the same extent, whereas the pairwise scheme only affects the two majority classes; (ii) a noise level $x\%$ with the uniform scheme implies that exactly $x\%$ of the examples in the datasets contain noise, whereas with the pairwise scheme, the number of noisy examples for the same noise level $x\%$ depends on the number of examples of the majority class N_{maj}; as a consequence, the global noise level in the whole dataset is usually lower–more specifically, the number of noisy examples can be computed as $(x \cdot N_{maj})/100$.

  1-NN is very sensitive to the number of noisy examples and it is therefore highly affected by the uniform class noise. Nevertheless, the pairwise class noise affects 3-NN and 5-NN more than the uniform class noise when the noise level is high enough–despite the former generating a lower number of noisy examples. This fact can be attributed to the fact that the uniform class noise generates more examples, but their presence in the output domain is more dispersed (since it affects all the classes). 3-NN and 5-NN take into account more than one neighbor in their prediction, hence, they can correctly predict the class of the examples that are close to those noisy training examples, since the rest of the neighbors are likely to be correctly labeled. However, with the pairwise scheme, it is more likely for these noisy examples to be close to each other, since all of them belonged to the majority class. Therefore, most of the neighbors used in the predictions of 3-NN and 5-NN in these noisy areas are incorrectly labeled, since these methods are more sensitive to this type of noise.
- BagC4.5 is more affected by the pairwise noise than by the uniform noise, whereas C4.5 behaves oppositely. This can be attributed to the diversity that both noise schemes produce in BagC4.5. In this case, the diversity comes from the examples in each sample used to train a classifier, since the baseline classifier is always the same (C4.5). The uniform noise introduces more diversity than the pairwise noise (since the noise affects all the classes instead of only two classes) and BagC4.5 take advantage of this higher diversity. This good behavior of BagC4.5 with the uniform class noise has been also mentioned in previous works [7].

2. The performance results show that:
   - Uniform class noise. Each MCS3-*k* generally outperforms its single classifier components. MCS3-*k* is better than SVM and *k*-NN regardless of the value of *k*, whereas it only performs statistically better than C4.5 at the lowest noise levels – even though the noise level where MCS3-*k* is better increases together with the value of *k*, obtaining significant differences at all the noise levels with $k = 5$. MCS5 is better than SVM, 1-NN and 3-NN but only better than C4.5 and 5-NN at the lowest noise levels. BagC4.5 always outperforms C4.5.
   - Pairwise class noise. MCS3-*k* improves SVM up to a 40% noise level, it is better than C4.5 at the lowest noise levels – these noise levels are lower than those of the uniform class noise – and also outperforms *k*-NN. Therefore, the behavior of MCS3-*k* with respect to their individual components is better in the uniform scheme than in the pairwise one. MCS5 is better than SVM and C4.5 at the lowest noise levels and better than *k*-NN regardless the value of *k*. BagC4.5 outperforms C4.5 if the noise levels is not very high.
3. The robustness results show that the higher the value of *k*, the greater the robustness of the MCS3-*k*. This fact is particularly notable with the most disruptive noise scheme, i.e., the uniform class noise scheme. Moreover:
   - Uniform class noise. MCS3-*k* are generally more robust, even though they are never more robust than all their components. The same occurs with MCS5; it is even less robust compared to 5-NN. BagC4.5 is more robust than C4.5.
   - Pairwise class noise. MCS3-*k* are more robust against uniform class noise than against pairwise noise. They are more robust than their single classifiers at lower noise levels than in the case of the uniform noise (in the case of SVM) or are indeed less robust statistically or equivalent in all the noise levels (in the case of C4.5) – even though MCS3-*k* is generally more robust than *k*-NN. In this case, MCS5 is sometimes less robust than its single classifiers and BagC4.5 is equal (at lowest noise levels) or less robust than C4.5.

5.2. Second scenario: Datasets with attribute noise

Table 4 shows the performance and robustness results of each classification algorithm at each noise level on datasets with attribute noise.

The results in this table are summarized hereafter:

- **Performance results with uniform attribute noise**.
  1. **Results of MCS3-*k***.
     - **MCS3-*k* vs. SVM**. MCS3-*k* statistically outperforms SVM regardless of the value of *k*. The *p*-values at the different noise levels decrease when *k* increases.
     - **MCS3-*k* vs. C4.5**. MCS3-1 performs well at the lowest noise levels (up to 10%), and is equivalent to C4.5 at the rest of the noise levels. Statistical differences are found in favor of MCS3-3 and MCS3-5 up to 15%, and are equivalent to C4.5 at the rest of the noise levels (even though MCS3-5 obtains lower *p*-values than MCS3-3).

Table 4
Performance and robustness results on datasets with attribute noise.

x%	Results										p-values														
	Single methods					Multi-classifiers					MCS3-1 vs.			MCS3-3 vs.			MCS3-5 vs.			MCS5 vs.					BagC4.5 vs.
	SVM	C4.5	1-NN	3-NN	5-NN	MCS3-1	MCS3-3	MCS3-5	MCS5	BagC4.5	SVM	C4.5	1-NN	SVM	C4.5	3-NN	SVM	C4.5	5-NN	SVM	C4.5	1-NN	3-NN	5-NN	C4.5
Performance																									
Uniform attribute noise																									
0%	83.25	82.96	81.42	82.32	82.32	85.42	85.48	85.52	83.74	85.18	5.2E−03	1.8E−03	7.1E−04	1.3E−02	5.0E−04	3.1E−03	1.0E−02	2.8E−04	5.0E−03	4.4E−01	1.9E−02	1.4E−04	9.7E−06	2.2E−03	1.5E−07
5%	82.83	82.62	80.17	81.20	81.38	84.57	84.84	84.77	82.96	84.89	2.8E−03	1.2E−02	2.4E−05	6.7E−04	1.7E−03	8.7E−05	6.4E−04	1.7E−03	9.4E−04	9.8E−02	5.8E−02	5.2E−06	6.6E−07	1.4E−04	1.5E−07
10%	81.78	81.58	78.52	79.78	80.25	83.33	83.64	83.80	81.69	84.36	3.9E−03	8.3E−02	8.1E−06	1.3E−03	8.1E−03	1.4E−04	7.4E−04	1.8E−03	5.8E−04	2.1E−01	7.8E−02	2.0E−06	1.5E−06	1.0E−03	2.3E−07
15%	80.25	80.64	77.22	78.66	79.25	81.81	82.18	82.31	80.61	83.51	5.2E−03	2.8E−01	9.2E−06	3.9E−03	5.8E−02	2.4E−04	2.6E−03	3.8E−02	1.0E−03	1.4E−01	2.4E−01	2.2E−06	2.0E−06	7.4E−04	6.5E−08
20%	78.75	79.98	75.73	77.52	78.40	80.64	81.14	81.42	79.68	82.92	4.4E−03	6.2E−01	5.2E−06	2.1E−03	1.9E−01	7.8E−05	1.5E−03	1.3E−01	3.9E−04	1.3E−01	4.0E−01	4.0E−07	4.6E−07	2.5E−03	4.3E−07
25%	77.58	78.92	74.26	76.20	77.39	79.37	79.85	80.22	78.53	82.03	2.8E−03	9.7E−01	3.8E−06	2.1E−03	5.1E−01	1.3E−04	8.2E−04	1.6E−01	5.5E−04	1.2E−01	9.6E−01	3.0E−07	5.0E−07	5.9E−03	1.7E−07
30%	76.09	77.64	72.58	74.84	76.04	77.97	78.42	78.75	77.14	80.99	2.1E−03	9.4E−01*	1.0E−05	1.1E−03	6.9E−01	2.7E−04	3.7E−04	2.3E−01	4.8E−03	5.8E−02	9.1E−01*	4.6E−07	1.5E−06	6.6E−03	1.7E−07
35%	74.11	76.33	71.11	73.68	74.95	76.26	76.91	77.23	75.96	79.94	5.2E−04	7.8E−01*	1.2E−05	4.7E−04	7.8E−01	2.6E−04	1.2E−04	3.8E−01	7.5E−03	2.1E−02	9.4E−01*	3.5E−07	4.3E−07	5.3E−03	4.3E−07
40%	72.75	75.19	69.58	72.22	73.60	74.84	75.57	76.03	74.58	78.76	9.9E−03	7.6E−01*	1.3E−06	2.3E−03	7.7E−01	1.6E−04	1.0E−03	4.0E−01	2.8E−03	6.8E−02	7.4E−01*	1.5E−07	8.1E−07	4.6E−03	1.3E−06
45%	70.51	73.38	68.10	70.85	72.38	72.93	73.70	74.27	73.03	77.38	1.9E−03	6.3E−01*	3.7E−05	9.4E−04	9.7E−01	1.2E−03	2.4E−04	4.5E−01	1.4E−02	3.2E−02	7.0E−01*	2.0E−06	4.0E−06	4.2E−02	2.0E−06
50%	69.46	72.12	66.59	69.16	70.87	71.36	72.14	72.85	71.57	76.18	1.2E−02	3.1E−01*	2.7E−05	2.4E−03	6.8E−01*	5.2E−04	6.1E−04	7.9E−01	7.8E−03	5.6E−02	5.5E−01*	6.1E−07	3.3E−06	5.1E−02	1.8E−06
Gauss ian attribute noise																									
0%	83.25	82.96	81.42	82.32	82.32	85.42	85.48	85.52	83.74	85.18	5.2E−03	1.8E−03	7.1E−04	1.3E−02	5.0E−04	3.1E−03	1.0E−02	2.8E−04	5.0E−03	4.4E−01	1.9E−02	1.4E−04	9.7E−06	2.2E−03	1.5E−07
5%	83.40	82.70	80.79	81.87	81.89	85.12	85.27	85.29	83.45	85.27	2.4E−03	3.0E−03	4.0E−05	2.3E−03	4.5E−04	6.4E−04	3.5E−03	3.3E−04	3.0E−03	2.4E−01	1.9E−02	7.6E−06	3.5E−06	6.4E−04	1.2E−07
10%	82.83	82.15	80.08	81.25	81.32	84.49	84.62	84.55	82.89	84.71	3.4E−03	4.8E−03	4.0E−05	1.6E−03	1.7E−03	7.8E−04	2.5E−03	1.9E−03	2.5E−03	2.4E−01	2.1E−02	1.5E−05	1.7E−06	3.5E−04	2.4E−07
15%	82.35	81.74	79.30	80.55	80.93	84.03	84.24	84.35	82.43	84.51	1.2E−02	5.4E−03	1.0E−05	7.8E−03	7.7E−04	4.0E−05	1.9E−03	5.4E−04	1.1E−03	1.9E−01	2.2E−02	8.1E−06	6.6E−07	1.6E−03	6.1E−08
20%	81.62	81.16	78.52	79.95	80.46	83.33	83.52	83.65	81.83	84.00	4.4E−03	2.1E−02	1.0E−05	2.7E−03	5.9E−03	1.9E−04	5.2E−04	2.6E−03	1.2E−03	1.4E−01	7.6E−02	8.1E−06	8.7E−07	2.4E−03	5.4E−07
25%	81.14	80.60	77.70	79.42	80.03	82.62	82.92	83.16	81.41	83.44	3.4E−03	4.3E−02	4.0E−05	8.6E−04	4.8E−03	5.8E−04	3.5E−04	2.0E−03	3.7E−03	1.1E−01	6.0E−02	1.4E−06	9.3E−07	4.0E−03	2.0E−07
30%	80.48	80.25	76.74	78.42	79.20	81.85	82.15	82.37	80.56	83.08	1.1E−02	2.0E−01	1.0E−05	2.7E−03	6.4E−02	1.6E−04	9.9E−04	1.9E−02	1.4E−03	9.8E−02	1.9E−01	2.0E−06	2.8E−07	1.2E−03	3.5E−07
35%	79.76	79.25	75.85	77.68	78.46	81.03	81.32	81.54	79.75	82.27	2.6E−02	1.2E−01	2.0E−05	1.2E−02	3.0E−02	7.8E−04	7.8E−03	5.4E−03	2.5E−03	3.1E−01	1.5E−01	1.6E−06	1.9E−06	2.1E−03	1.1E−07
40%	78.88	78.84	74.82	77.11	78.03	80.34	80.87	81.04	79.32	81.87	1.6E−02	4.0E−01	1.0E−05	4.0E−03	6.8E−02	2.2E−04	3.2E−03	1.9E−02	2.4E−03	2.1E−01	3.6E−01	3.3E−07	8.1E−07	5.6E−03	7.3E−07
45%	78.27	77.80	74.01	76.02	76.99	79.45	79.71	79.98	78.25	81.04	3.5E−02	2.6E−01	0.0E+00	1.3E−02	1.3E−01	5.5E−04	6.6E−03	3.7E−02	1.9E−03	4.3E−01	3.1E−01	8.1E−07	3.1E−06	7.2E−03	2.4E−06
50%	77.26	77.01	73.31	75.59	76.50	78.49	79.08	79.21	77.86	80.39	2.2E−02	3.4E−01	3.0E−05	3.9E−03	7.6E−02	3.0E−04	2.9E−03	3.0E−02	5.4E−03	2.9E−01	2.7E−01	8.1E−07	1.8E−06	1.1E−03	2.4E−06
Robustness																									
Uniform attribute noise																									
5%	0.16	0.39	1.38	1.28	0.97	0.95	0.71	0.85	0.94	0.34	2.8E−01	1.3E−01*	2.6E−01	8.1E−02	3.7E−01*	4.9E−02	2.4E−01	1.7E−01*	2.7E−02	2.6E−01	6.0E−02*	1.2E−02	2.4E−03	4.7E−02	5.2E−01
10%	1.38	1.68	3.63	3.29	2.52	2.40	2.11	1.98	2.57	0.91	6.7E−01*	6.6E−02*	1.3E−02	5.5E−01	3.0E−01*	7.5E−03	4.1E−01	4.6E−01*	4.8E−02	8.4E−01*	1.5E−01*	7.8E−04	4.5E−04	5.6E−01	7.8E−03
15%	3.28	2.84	5.19	4.29	3.39	4.23	3.86	3.76	3.63	1.95	8.6E−01*	1.8E−02*	7.8E−02	5.9E−01	5.5E−02*	8.3E−02	5.3E−01	3.7E−02*	3.3E−01	6.5E−01	1.8E−01*	6.7E−04	7.2E−03	5.5E−01	7.8E−03
20%	5.06	3.60	6.78	5.53	4.35	5.54	5.02	4.80	4.68	2.63	6.1E−01	5.2E−03*	1.3E−02	4.3E−01	3.7E−02*	2.7E−02	3.2E−01	3.2E−02*	1.1E−01	3.5E−01	8.5E−02*	2.6E−05	3.7E−03	8.2E−01	1.2E−02
25%	6.45	4.98	8.47	7.04	5.41	7.09	6.61	6.23	6.10	3.73	9.8E−01	5.9E−03*	9.5E−03	7.6E−01	2.2E−02*	1.9E−02	2.9E−01	5.5E−02*	1.1E−01	5.5E−01	3.2E−02*	3.1E−05	6.9E−03	7.3E−01*	4.6E−03
30%	8.35	6.62	10.73	8.67	7.17	8.85	8.36	8.06	7.84	5.00	7.8E−01	2.6E−03*	1.2E−02	6.7E−01	6.9E−03*	6.0E−02	2.9E−01	2.2E−02*	2.6E−01	5.0E−01	4.0E−02*	4.6E−05	5.4E−03	1.0E+00	5.2E−04
35%	10.62	8.23	12.32	9.93	8.13	10.79	10.12	9.82	9.17	6.24	3.1E−01	4.4E−03*	2.0E−02	2.3E−01	2.8E−02*	4.1E−02	1.0E−01	4.1E−02*	4.4E−01	1.7E−01	8.5E−02*	3.5E−05	5.9E−03	8.6E−01*	1.4E−04
40%	12.13	9.60	14.29	11.64	9.71	12.44	11.67	11.23	10.74	7.57	7.8E−01*	1.1E−02*	2.0E−03	8.1E−01	4.8E−02*	2.0E−02	5.8E−01	9.3E−02*	1.3E−01	6.6E−01	7.4E−02*	2.4E−06	4.8E−03	9.0E−01	1.2E−03
45%	14.92	11.87	16.04	13.28	10.91	14.73	13.98	13.38	12.75	9.31	6.9E−01	1.1E−02*	2.5E−02	5.5E−01	6.8E−02*	9.3E−02	2.2E−01	1.3E−01*	3.5E−01	4.2E−01	1.6E−01*	2.4E−04	9.9E−03	6.6E−01*	3.2E−04
50%	16.28	13.46	17.70	15.17	12.48	16.68	15.85	15.09	14.40	10.75	8.5E−01*	3.4E−03*	2.0E−02	6.9E−01	4.0E−02*	3.0E−02	2.9E−01	1.3E−01*	2.9E−01	4.6E−01	1.0E−01*	1.6E−04	3.4E−03	7.5E−01*	3.3E−04
Gaussian attribute noise																									
5%	−0.46	0.33	0.80	0.50	0.58	0.31	0.19	0.25	0.32	−0.16	7.8E−02	1.4E−01*	3.2E−01	6.8E−02	9.6E−01	3.3E−01	1.2E−01	6.0E−01*	5.4E−01	1.6E−01	6.4E−01*	8.1E−02	1.2E−01	1.2E−01	8.9E−02
10%	0.25	0.94	1.60	1.28	1.20	1.03	0.98	1.16	0.92	0.53	1.7E−01	2.4E−01*	2.6E−01	2.5E−01	4.8E−01*	9.0E−02	7.7E−01	2.3E−01*	4.1E−01	2.2E−01	8.4E−01*	3.5E−02	2.2E−02	6.4E−02	1.7E−01
15%	0.75	1.48	2.51	2.09	1.84	1.54	1.39	1.32	1.48	0.77	9.9E−01*	1.3E−01*	5.3E−02	5.5E−01	6.0E−01*	1.6E−02	4.1E−01	7.5E−01*	2.2E−01	6.2E−01	6.5E−01*	8.2E−04	5.9E−03	5.3E−01	3.8E−02
20%	1.74	2.10	3.36	2.74	2.03	2.36	2.21	2.14	2.07	1.35	3.3E−01	1.2E−01*	1.0E−01	8.3E−02	1.4E−01*	2.6E−02	9.3E−02	3.3E−01*	3.1E−01	2.8E−01	2.0E−01*	8.8E−03	1.8E−03	8.3E−01	2.0E−01
25%	2.32	2.89	4.25	3.27	2.67	3.20	2.95	2.73	2.56	2.03	6.2E−01	1.7E−01*	5.8E−02	3.9E−01	3.1E−01*	8.5E−02	2.1E−01	6.2E−01*	5.0E−01	2.6E−01	7.0E−01*	3.9E−04	8.1E−03	6.6E−01	4.2E−02
30%	3.14	3.25	5.64	4.70	3.80	4.14	3.86	3.65	3.76	2.44	7.3E−01*	1.3E−02*	7.2E−02	5.4E−01	3.8E−02*	2.6E−02	2.6E−01	1.1E−01*	2.5E−01	4.0E−01	1.4E−01*	3.2E−03	1.3E−03	7.3E−01	5.0E−02
35%	4.11	4.52	6.67	5.60	4.48	5.09	4.82	4.63	4.62	3.38	7.1E−01*	1.4E−01*	4.5E−02	8.7E−01	3.8E−01*	5.6E−02	5.4E−01	5.8E−01*	4.6E−01	7.7E−01	4.9E−01*	2.1E−04	1.5E−03	9.8E−01*	6.0E−03
40%	5.23	5.02	7.91	6.17	4.96	5.93	5.38	5.24	5.11	3.89	9.4E−01	1.5E−02*	2.6E−03	4.0E−01	2.4E−01*	7.2E−03	3.8E−01	2.0E−01*	5.3E−01	4.0E−01	3.1E−01*	6.7E−06	1.0E−03	6.8E−01	4.2E−02
45%	6.02	6.30	8.96	7.54	6.39	7.03	6.80	6.53	6.38	4.88	8.3E−01*	1.8E−01*	1.4E−02	9.1E−01*	3.5E−01*	3.6E−02	9.4E−01*	5.5E−01*	7.2E−01	7.9E−01	4.5E−01*	4.1E−05	1.8E−03	6.3E−01	2.7E−02
50%	7.28	7.29	9.51	7.75	6.68	8.14	7.50	7.43	6.68	5.60	9.8E−01	6.0E−02*	5.1E−02	4.4E−01	3.9E−01*	3.0E−02	5.4E−01	2.6E−01*	5.9E−01	3.4E−01	4.8E−01*	9.2E−05	1.6E−03	5.8E−01	6.6E−03

   - **MCS3-*k* vs. *k*-NN**. MCS3-k is statistically better than k-NN regardless of the value of k. The p-values increase together with k.

  2. **Results of MCS5.** MCS5 outperforms k-NN for the different values of k (the p-values increase together with k). MCS5 is equivalent to SVM below of the noise level 30% and better from this level. MCS5 is better than C4.5 at the lowest noise levels (up to 10%), equivalent at intermediate noise levels (15–25%) and worse at the rest of the noise levels.
  3. **Results of BagC4.5.** BagC4.5 is better than C4.5 at all the noise levels.
- **Performance results with Gaussian attribute noise**.
  1. **Results of MCS3-*k*.**
     - **MCS3-*k* vs. SVM**. MCS3-k is significantly better than SVM regardless of the k value. The p-values at the different noise levels decrease as k increases.
     - **MCS3-*k* vs. C4.5**. MCS3-k performs well up to 25% ($k = 1$) and at all noise levels for $k = 3$ and $k = 5$. The p-values at the different noise levels decrease as k increases.
     - **MCS3-*k* vs. *k*-NN**. MCS3-k performs statistically better than k-NN regardless of the value of k. The p-values at the different noise levels increase with greater values of k.

  2. **Results of MCS5.** MCS5 outperforms k-NN for the different values of k (the p-values increase together with k). MCS5 works well below 30% compared with C4.5 and both are equivalent onwards. MCS5 is also equivalent to SVM at all the noise levels.
  3. **Results of BagC4.5.** BagC4.5 is better than C4.5 at all the noise levels.

The robustness results obtained are summarized below:

- **Robustness results with uniform attribute noise**.
  1. **Results of MCS3-*k*.**
     - **MCS3-*k* vs. SVM**. MCS3-k is equivalent to SVM regardless of the value of k. MCS3-k generally obtains more ranks, except for some cases with $k = 1$. The p-values at the different noise levels decrease as k increases.
     - **MCS3-*k* vs. C4.5**. C4.5 obtains more ranks, outperforming MCS3-k. This superiority is not significant at the lowest noise levels (up to 5–10%). The p-values at the different noise levels increase with k.
     - **MCS3-*k* vs. *k*-NN**. MCS3-k obtains more ranks than k-NN. Statistical differences are found with with $k = 1$ and $k = 3$. MCS3-5 outperforms 5-NN at the lowest noise levels (5% and 10%) and is equivalent at the rest. The p-values at the different noise levels increase with the value of k.

  2. **Results of MCS5.** MCS5 is equivalent to SVM, 5-NN and C4.5 (it is even worse at some isolated noise levels). However, it is better than 1-NN and 3-NN.
  3. **Results of BagC4.5.** Generally, BagC4.5 obtains more ranks than C4.5.
- **Robustness results with Gaussian attribute noise**.
  1. **Results of MCS3-*k*.**
     - **MCS3-*k* vs. SVM**. No remarkable differences are found between MCS3-k and SVM. For $k = 1$ and $k = 3$, MCS3-k is only better at 5%. In most of the cases MCS3-k obtains more ranks than SVM, although some exceptions are found with $k = 1$.
     - **MCS3-*k* vs. C4.5**. C4.5 obtains more ranks, although both methods are statistically equivalent. Moreover, C4.5 is better at some noise levels with $k = 1$.
     - **MCS3-*k* vs. *k*-NN**. MCS3-k obtains more ranks than k-NN, although statistical differences are only found with $k = 1$ and $k = 3$. The p-values with different noise levels increase together with k.

  2. **Results of MCS5.** MCS5 is equivalent to SVM, C4.5 and 5-NN and better than 1-NN and 3-NN at most of the noise levels.
  3. **Results of BagC4.5.** Generally, BagC4.5 is better than C4.5 except at some isolated noise levels.

From these performance and robustness results, the following remarks can be made:

1. The results on datasets with uniform attribute noise are much worse than those on datasets with Gaussian noise for all the classifiers, including MCSs. Hence, the most disruptive attribute noise is the uniform scheme.
2. The results of performance on datasets with attribute show:
   - Uniform attribute noise. This is the most disruptive noise scheme; MCS3-k outperforms SVM and k-NN. However, with respect to C4.5, MCS3-k is significantly better at the lowest noise levels (up to 10–15%), and is equivalent at the rest of the noise levels. MCS5 is better than k-NN regardless the value of k and it is better than SVM only from an intermediate noise level (30%). The results of MCS5 compared to C4.5 depends on the noise level: they are better if the noise level is lower. BagC4.5 outperforms C4.5.

- Gaussian attribute noise. This is the least disruptive noise scheme; MCS3-*k* outperforms its individual classifiers in most of the cases, particularly if the noise level is relatively low. Thus, MCS3-*k* is generally the best method with $k = 3$ and $k = 5$. In the case of $k = 1$, MCS3-1 is only better than 1-NN and SVM, and better than C4.5 at the lowest noise levels (up to 25%). MCS5 is better than *k*-NN regardless the value of *k*, equivalent to SVM and it is better or equivalent to C4.5 (depending if the noise level is low or high, respectively). BagC4.5 also outstands over C4.5 in this case.

3. The robustness results show, as in the case of the robustness results with class noise, that the higher the value of *k* is, the greater robustness the MCS3-*k* has. This fact is particularly notable with the most disruptive noise scheme (the uniform one) but also when the noise level becomes high in both schemes, uniform and Gaussian. Moreover, the following points must be stressed:
   - Uniform attribute noise. The robustness of the MCS3-*k* does not outperform that of its individual classifiers, as it is statistically equivalent to SVM and sometimes worse than C4.5. Regarding *k*-NN, MCS3-*k* performs better than 1-NN and 3-NN, but is equivalent to 5-NN. The same occurs with MCS5, since it is equivalent to SVM, 5-NN and C4.5 and, at some noise levels, even worse than C4.5. The exception is with BagC4.5, which is more robust than C4.5.
   - Gaussian attribute noise. The robustness results are better than those of the uniform noise. The main difference in this case is that MCS3-*k* and MCS5 are not statistically worse than C4.5. BagC4.5 is also more robust than C4.5, except at some isolated noise levels.

6. Comparing different decisions combination methods

In this section a series of decisions combination methods (W-MAJ, NB and BKS) are compared with respect to the MAJ scheme used in the previous section. The performance and robustness of the MCSs built with heterogeneous classifiers (MCS3-*k* and MCS5) using these combination methods are studied. The results of each one of these three techniques (W-MAJ, NB and BKS) are compared to those of the MAJ scheme using Wilcoxon test. Table 5 shows the performance and robustness results of the decisions combination methods on datasets with class noise.

From this table, it can be observed that:

1. **Performance results:**
   (a) **Uniform class noise.**
   - Attending to the average results, MAJ is generally the best method, followed by NB; W-MAJ and BKS are placed to a certain distance of the previous ones.
   - The *p*-values obtained show that MAJ is better than W-MAJ and NB (from a noise level of 5–10% approximately). At the lowest noise levels W-MAJ and NB are even better than MAJ. In addition, MAJ is better than BKS.

   (b) **Pairwise class noise.**
   - The average results show that, in MCS3-*k*, MAJ is the best method, followed by NB and W-MAJ (with equaled results at some noise levels). The worst method is BKS. In MCS5, the situation is different: the best methods are W-MAJ and NB, followed by BKS and MAJ to a certain distance.
   - Attending to the *p*-values, W-MAJ and NB are usually better than MAJ in MCS3-*k* at extreme noise levels (0–5% and 45–50%). In MCS5, W-MAJ and NB are clearly better than MAJ, but MAJ is better than BKS (except at the highest noise levels–from a noise level of 40% onwards).

2. **Robustness results:**
   (a) **Uniform class noise.**
   - W-MAJ and BKS share the best average results, followed by NB and MAJ.
   - However, the *p*-values show that MAJ is more robust than the rest of the methods.

   (b) **Pairwise class noise.**
   - The average results show that, in MCS3-*k*, W-MAJ and NB are the best methods, followed by BKS and MAJ in the last position. In MCS5, NB and BKS are the first methods in average results, followed by W-MAJ and MAJ (except at the highest noise levels where MAJ is better).
   - The *p*-values show that MAJ is more robust up to an intermediate noise level (25–30%). Then, the rest of the methods are equivalent or even more robust (45–50% of noise level approximately).

Table 6 shows the performance and robustness results of the decisions combination methods on datasets with attribute noise.

From this table, it can be observed that:

1. **Performance results.**
   (a) **Uniform attribute noise.**

Table 5
Performance and robustness results on datasets with class noise: comparison of decisions combination methods.

x%	Results																p-values											
	Majority				Weighted				Bayes				BKS				Weighted				Bayes				BKS			
	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5
Performance																												
Uniform class noise																												
0%	85.42	85.48	85.52	83.74	84.92	85.04	85.04	85.83	85.03	85.22	85.29	86.10	84.25	84.39	84.39	84.72	9.2E−01*	8.0E−01*	5.0E−01	3.0E−02*	9.3E−01*	9.2E−01*	9.5E−01	1.2E−02*	2.4E−02	3.4E−02	2.6E−02	7.8E−01
5%	84.11	84.86	85.03	83.15	82.27	83.05	83.17	84.61	82.36	82.61	82.73	84.57	81.54	81.48	81.51	82.26	1.5E−01	4.9E−02	2.1E−01	2.5E−01*	6.2E−02	4.9E−03	1.0E−02	2.9E−01*	5.4E−04	1.9E−05	5.9E−06	2.8E−02
10%	83.00	84.25	84.57	82.60	80.47	80.83	80.94	83.15	80.84	81.25	81.29	83.14	79.93	80.04	80.09	81.02	1.2E−02	4.2E−04	6.4E−05	6.0E−01	2.0E−02	9.0E−05	1.8E−04	9.1E−01	3.1E−05	5.5E−07	2.5E−07	5.6E−03
15%	81.60	83.21	83.63	81.74	79.02	79.43	79.44	80.98	79.23	79.51	79.67	81.28	78.37	78.37	78.31	79.60	5.7E−03	2.1E−04	6.6E−05	1.3E−01	3.5E−03	2.9E−05	2.2E−05	2.9E−01	2.3E−05	6.1E−07	1.3E−07	2.0E−03
20%	80.09	81.98	82.69	80.55	77.02	77.45	77.68	78.03	77.40	77.70	77.86	79.33	76.47	76.60	76.57	77.91	1.1E−03	4.2E−05	1.3E−05	6.4E−02	9.2E−04	5.5E−06	3.1E−06	9.6E−02	1.4E−06	9.8E−08	8.2E−08	1.2E−03
25%	78.53	80.87	81.78	79.50	74.86	75.73	76.12	75.42	75.54	75.93	76.18	77.57	74.71	74.94	74.97	76.53	3.4E−04	2.2E−05	1.3E−05	1.4E−02	7.1E−04	6.3E−06	9.2E−07	3.2E−02	5.2E−06	8.8E−08	4.8E−08	1.0E−03
30%	77.10	79.69	80.75	78.18	72.68	73.54	74.35	73.56	74.17	74.60	74.80	76.06	73.18	73.37	73.36	75.01	1.5E−04	2.0E−06	4.4E−06	5.1E−03	2.4E−04	1.3E−06	3.7E−07	2.3E−02	2.4E−06	8.2E−08	4.2E−08	9.4E−04
35%	75.11	77.87	79.16	76.44	70.91	71.37	71.92	71.39	72.37	72.80	73.15	74.29	71.46	71.66	71.62	73.15	9.6E−05	2.2E−06	1.9E−06	3.5E−03	3.5E−04	1.9E−06	9.0E−07	1.9E−02	3.8E−06	5.6E−08	3.6E−08	4.7E−04
40%	73.20	76.04	77.56	74.72	68.95	69.37	69.94	68.77	70.46	70.97	71.37	72.31	69.45	69.59	69.65	71.08	9.0E−05	5.4E−06	3.8E−06	3.4E−04	8.3E−04	7.5E−06	3.4E−06	1.8E−02	6.7E−06	4.3E−07	1.7E−07	3.2E−04
45%	70.68	73.61	75.27	72.16	66.57	67.30	67.75	66.19	68.45	68.91	69.30	70.08	67.35	67.68	67.63	69.01	1.1E−04	5.8E−06	2.1E−06	4.5E−04	2.2E−03	2.0E−05	2.6E−06	1.2E−02	1.2E−05	8.7E−07	1.7E−07	1.1E−03
50%	67.64	70.82	72.70	69.43	63.33	64.33	64.81	63.50	65.99	66.30	66.76	67.40	64.68	65.01	65.07	66.19	1.7E−03	2.6E−06	1.7E−06	2.2E−04	5.6E−02	2.5E−05	5.0E−06	3.3E−02	2.1E−04	8.1E−07	2.6E−07	1.7E−03
Pairwise class noise																												
0%	85.42	85.48	85.52	83.74	84.92	85.04	85.04	85.83	85.03	85.22	85.29	86.10	84.25	84.39	84.39	84.72	9.2E−01*	8.0E−01*	5.0E−01	3.0E−02*	9.3E−01*	9.2E−01*	9.5E−01	1.2E−02*	2.4E−02	3.4E−02	2.6E−02	7.8E−01
5%	84.75	85.19	85.28	83.54	83.50	84.12	84.18	85.19	83.43	83.63	83.66	85.01	82.74	82.76	82.78	83.27	1.2E−01	1.1E−01	1.3E−01	2.3E−01*	5.5E−03	7.1E−04	1.9E−03	6.4E−01*	8.7E−05	1.1E−05	2.3E−05	1.9E−02
10%	83.95	84.61	84.86	82.99	81.83	82.11	82.24	84.43	82.33	82.42	82.52	83.59	81.51	81.61	81.60	82.17	2.5E−02	3.1E−02	1.7E−02	2.6E−01*	1.9E−03	2.7E−04	1.8E−04	3.2E−01	3.3E−05	2.9E−06	1.1E−06	6.1E−03
15%	83.09	83.89	84.29	82.23	80.81	81.08	81.28	83.36	81.19	81.36	81.31	82.41	80.37	80.51	80.53	81.04	1.5E−02	3.1E−02	9.3E−03	2.9E−01*	1.4E−03	2.2E−04	6.9E−05	1.6E−01	1.3E−05	2.1E−06	4.6E−07	9.1E−03
20%	82.21	83.06	83.50	80.99	80.18	80.49	80.59	82.33	80.49	80.67	80.63	81.43	79.71	79.84	79.87	80.38	3.3E−02	2.8E−02	4.9E−03	2.1E−01*	2.1E−03	7.1E−04	1.2E−04	3.7E−01	6.9E−05	9.2E−06	2.5E−06	4.7E−02
25%	80.81	81.64	82.14	79.54	78.83	79.12	79.20	80.25	78.96	79.17	79.22	79.80	78.33	78.47	78.57	79.06	1.0E−02	9.8E−03	6.5E−03	2.8E−01*	5.0E−03	1.7E−03	1.1E−03	2.8E−01	2.1E−04	3.7E−05	1.4E−05	9.8E−02
30%	79.52	80.24	80.76	77.48	77.64	77.90	77.99	78.62	77.90	78.03	78.09	78.29	77.25	77.36	77.50	77.84	8.4E−02	5.6E−02	1.2E−02	7.4E−02*	4.0E−02	9.1E−03	3.0E−03	9.1E−01	2.2E−03	1.8E−04	6.2E−05	7.7E−01
35%	77.65	78.13	78.66	75.20	76.09	76.42	76.58	76.98	76.42	76.56	76.51	76.73	75.71	75.81	75.86	76.17	4.6E−01	2.8E−01	1.1E−01	6.5E−03*	2.2E−01	1.3E−01	4.1E−02	2.9E−01*	2.1E−02	1.0E−02	2.5E−03	5.6E−01*
40%	75.25	75.61	75.93	72.21	74.09	74.23	74.27	74.87	74.50	74.58	74.55	75.03	73.96	74.05	74.09	74.27	7.4E−01	3.2E−01	2.4E−01	1.0E−03*	4.9E−01	2.6E−01	1.4E−01	6.1E−03*	1.4E−01	5.1E−02	2.9E−02	8.5E−02*
45%	71.82	71.97	72.01	68.85	71.65	71.54	71.44	71.75	71.61	71.77	71.85	72.60	71.26	71.35	71.35	71.31	5.8E−01*	7.6E−01*	9.5E−01*	8.3E−05*	9.6E−01*	9.9E−01	9.1E−01	6.2E−05*	4.6E−01	4.5E−01	4.3E−01	4.7E−02*
50%	64.46	64.23	64.00	63.67	65.77	65.74	65.71	66.15	67.08	67.26	67.32	68.96	66.67	66.63	66.67	66.64	1.2E−02*	8.1E−03*	4.7E−03*	3.4E−03*	1.7E−04*	4.1E−05*	1.6E−05*	2.6E−07*	2.1E−03*	1.6E−03*	4.1E−04*	6.9E−03*
Robustness																												
Uniform class noise																												
5%	1.59	0.73	0.61	0.76	3.18	2.50	2.35	1.62	3.27	3.18	3.10	1.85	3.34	3.49	3.44	2.83	2.7E−02	9.8E−03	8.5E−02	5.8E−02	3.0E−04	1.5E−06	1.1E−05	1.5E−04	1.6E−04	4.3E−07	3.8E−07	7.3E−05
10%	2.91	1.47	1.12	1.43	5.22	4.99	4.88	3.42	5.04	4.77	4.78	3.46	5.23	5.18	5.08	4.26	4.4E−03	7.6E−04	1.6E−04	1.1E−03	8.8E−05	9.4E−06	1.7E−06	8.6E−04	4.9E−05	1.5E−06	4.0E−07	5.2E−05
15%	4.59	2.72	2.31	2.35	6.98	6.75	6.75	6.14	7.00	6.88	6.76	5.70	7.11	7.21	7.24	5.99	3.0E−02	2.1E−04	2.0E−04	1.2E−03	7.5E−04	1.9E−06	7.9E−07	2.3E−04	8.2E−04	7.6E−06	1.1E−06	1.4E−04
20%	6.40	4.24	3.47	3.85	9.37	9.10	8.87	9.67	9.15	9.01	8.91	7.99	9.42	9.35	9.37	8.06	3.0E−03	1.4E−04	1.9E−04	4.4E−04	1.7E−04	1.2E−06	2.4E−06	2.0E−04	4.4E−05	8.7E−07	1.4E−06	1.8E−05
25%	8.26	5.54	4.52	5.10	11.94	11.17	10.71	12.66	11.32	11.09	10.86	10.02	11.51	11.32	11.25	9.64	3.1E−03	8.0E−05	1.3E−04	3.7E−04	2.5E−04	3.1E−06	2.3E−07	1.5E−04	8.7E−05	9.9E−07	2.6E−07	3.7E−05
30%	9.95	6.92	5.78	6.72	14.63	13.85	12.87	14.93	12.93	12.64	12.48	11.79	13.29	13.15	13.13	11.37	1.7E−03	6.2E−06	6.1E−06	1.8E−04	4.4E−04	4.7E−07	3.4E−07	1.3E−04	4.6E−05	7.6E−07	2.6E−07	1.0E−04
35%	12.33	9.10	7.72	8.67	16.68	16.42	15.75	17.44	15.04	14.75	14.41	13.85	15.32	15.16	15.20	13.58	3.5E−03	1.2E−05	1.8E−05	8.2E−05	1.5E−03	2.6E−06	1.0E−06	2.6E−04	2.7E−04	2.4E−06	1.3E−06	3.5E−04
40%	14.54	11.20	9.55	10.56	18.99	18.77	18.07	20.46	17.25	16.86	16.49	16.12	17.70	17.58	17.51	15.95	2.6E−03	2.3E−05	1.1E−05	3.0E−05	3.5E−03	8.5E−06	2.9E−06	3.2E−04	2.2E−04	1.0E−05	4.3E−06	1.3E−04
45%	17.56	14.10	12.30	13.86	21.89	21.26	20.76	23.53	19.66	19.34	18.97	18.76	20.23	19.90	19.96	18.43	2.9E−03	1.7E−05	5.4E−06	7.3E−05	1.7E−02	4.1E−05	4.4E−06	8.6E−04	7.4E−04	2.3E−05	5.2E−06	9.4E−04
50%	21.08	17.41	15.35	16.81	25.63	24.67	24.07	26.38	22.47	22.33	21.87	21.80	23.29	23.00	22.91	21.68	1.4E−02	2.3E−05	4.0E−06	1.5E−05	1.7E−01	8.3E−05	1.2E−05	1.6E−03	9.1E−03	2.2E−05	6.7E−06	1.7E−03
Pairwise class noise																												
5%	0.79	0.34	0.28	0.17	1.65	1.17	1.09	0.83	1.83	1.83	1.87	1.27	1.74	1.88	1.84	1.67	1.2E−02	2.3E−02	6.4E−02	6.9E−03	2.2E−04	7.3E−05	6.5E−05	1.7E−05	6.7E−04	3.5E−05	2.1E−04	1.4E−04
10%	1.73	1.05	0.81	0.76	3.56	3.44	3.31	1.75	3.13	3.25	3.20	2.91	3.18	3.22	3.24	2.94	5.5E−03	5.2E−03	1.6E−02	1.5E−02	2.4E−04	1.3E−05	5.9E−06	2.4E−05	2.8E−04	9.2E−05	1.8E−05	2.1E−04
15%	2.81	1.95	1.53	1.68	4.72	4.61	4.40	3.00	4.39	4.45	4.58	4.22	4.51	4.50	4.46	4.27	2.7E−02	8.0E−03	5.8E−03	5.8E−02	7.1E−04	3.1E−05	2.5E−06	4.6E−05	4.3E−04	3.5E−05	1.1E−05	3.5E−04
20%	3.86	2.97	2.51	3.08	5.46	5.32	5.22	4.19	5.23	5.28	5.38	5.35	5.28	5.26	5.20	4.98	1.2E−01	4.5E−02	1.2E−02	5.8E−01	6.9E−03	6.7E−04	4.1E−05	9.0E−04	1.3E−02	1.3E−03	3.3E−04	2.5E−02
25%	5.55	4.68	4.16	4.69	7.06	6.95	6.87	6.58	7.01	7.01	7.02	7.23	6.87	6.86	6.73	6.52	2.3E−01	5.9E−02	3.3E−02	8.2E−01	1.9E−02	8.1E−03	1.1E−03	4.0E−03	5.3E−02	7.2E−03	3.9E−03	1.4E−01
30%	7.11	6.35	5.81	7.16	8.45	8.40	8.32	8.47	8.25	8.35	8.35	8.95	8.16	8.16	7.99	8.01	4.0E−01	4.1E−01	6.5E−02	5.1E−01*	8.8E−02	2.1E−02	6.6E−03	1.5E−01	2.2E−01	4.5E−02	2.1E−02	7.3E−01
35%	9.30	8.88	8.32	9.64	10.21	10.09	9.93	10.34	9.94	10.01	10.14	10.76	9.94	9.96	9.90	9.92	6.3E−01	8.4E−01	6.0E−01	1.0E−01*	2.0E−01	2.4E−01	1.0E−01	9.6E−01*	2.2E−01	2.1E−01	1.2E−01	5.9E−01*
40%	12.08	11.83	11.52	13.17	12.51	12.60	12.57	12.75	12.14	12.29	12.40	12.67	11.96	11.99	11.93	12.20	7.1E−01*	7.9E−01	7.7E−01	3.5E−02*	5.0E−01	4.1E−01	2.7E−01	1.2E−01*	7.7E−01	5.8E−01	4.8E−01	1.6E−01*
45%	16.05	15.96	16.01	16.92	15.33	15.64	15.78	16.20	15.45	15.46	15.42	15.37	15.10	15.12	15.11	15.59	3.6E−01*	5.4E−01*	6.5E−01*	2.5E−02*	7.2E−01*	8.1E−01*	7.4E−01*	1.0E−02*	6.6E−01*	8.0E−01*	7.2E−01*	1.3E−01*
50%	24.13	24.54	24.90	22.70	21.93	22.08	22.13	22.39	20.55	20.50	20.50	19.38	20.34	20.48	20.43	20.82	5.4E−03*	7.8E−03*	5.1E−03*	2.5E−01*	4.3E−05*	2.1E−05*	5.9E−06*	3.7E−04*	2.3E−05*	4.4E−05*	1.0E−05*	4.8E−02*

Table 6
Performance and robustness results on datasets with attribute noise: comparison of decisions combination methods.

x%	Results																p-values											
	Majority				Weighted				Bayes				BKS				Weighted				Bayes				BKS			
	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5	MCS3-1	MCS3-3	MCS3-5	MCS5
Performance																												
Uniform attribute noise																												
0%	85.42	85.48	85.52	83.74	84.92	85.04	85.04	85.83	85.03	85.22	85.29	86.10	84.25	84.39	84.39	84.72	9.2E−01*	8.0E−01*	5.0E−01	3.0E−02*	9.3E−01*	9.2E−01*	9.5E−01	1.2E−02*	2.4E−02	3.4E−02	2.6E−02	7.8E−01
5%	84.57	84.84	84.77	82.96	83.58	83.78	83.80	84.69	84.25	84.46	84.38	85.13	83.56	83.39	83.29	83.87	9.8E−02	6.8E−02	2.9E−02	1.1E−01*	7.4E−01	5.7E−01	7.7E−01	2.5E−02*	1.9E−02	6.4E−03	3.4E−03	2.9E−01
10%	83.33	83.64	83.80	81.69	82.51	82.63	82.59	82.86	83.10	83.25	83.27	84.09	82.28	82.17	82.17	82.65	1.1E−01	1.8E−01	5.5E−02	3.1E−01*	8.2E−01	8.1E−01	5.2E−01	1.3E−02*	1.2E−02	3.5E−03	5.9E−03	6.2E−01
15%	81.81	82.18	82.31	80.61	80.69	80.87	80.93	81.04	81.42	81.61	81.64	82.45	80.89	80.72	80.63	81.14	5.2E−02	1.0E−01	5.6E−02	8.0E−01	2.6E−01	1.7E−01	1.5E−01	9.0E−02*	3.2E−02	5.5E−03	7.7E−03	3.5E−01
20%	80.64	81.14	81.42	79.68	78.73	79.18	79.34	79.22	79.82	79.98	80.01	81.03	79.31	79.20	79.16	79.80	5.8E−03	6.4E−03	5.1E−03	6.6E−01	1.2E−01	5.3E−02	3.9E−02	2.9E−01*	2.2E−03	1.8E−03	3.0E−03	2.9E−01
25%	79.37	79.85	80.22	78.53	77.18	77.64	77.98	77.99	78.46	78.66	78.74	79.69	78.07	78.00	78.02	78.60	2.2E−03	3.2E−03	1.8E−03	3.6E−01	8.4E−02	4.6E−02	1.8E−02	3.1E−01*	2.3E−03	9.4E−04	5.0E−04	3.2E−01
30%	77.97	78.42	78.75	77.14	75.23	75.50	75.75	76.53	76.84	76.87	77.05	78.07	76.54	76.39	76.40	76.98	4.8E−04	2.9E−04	9.2E−05	4.0E−01	3.6E−02	1.5E−02	8.7E−03	6.5E−01*	1.6E−03	4.3E−04	1.1E−04	1.7E−01
35%	76.26	76.91	77.23	75.96	73.44	73.57	73.69	74.66	74.78	74.90	74.99	76.16	74.61	74.60	74.53	75.29	9.0E−05	8.5E−05	1.2E−05	1.6E−01	4.8E−03	4.4E−03	1.5E−03	5.4E−01	3.7E−04	5.8E−04	6.2E−05	5.1E−02
40%	74.84	75.57	76.03	74.58	72.34	72.60	72.76	72.92	73.24	73.37	73.44	74.43	73.02	72.99	73.18	73.73	5.5E−03	1.1E−03	4.5E−04	1.4E−01	3.8E−02	2.0E−02	6.5E−03	6.8E−01	5.6E−03	2.3E−03	7.4E−04	3.1E−01
45%	72.93	73.70	74.27	73.03	70.26	70.64	70.78	70.90	70.99	71.10	71.20	72.05	70.87	70.81	70.92	71.63	8.3E−04	5.4E−04	1.4E−04	5.5E−02	5.2E−03	5.2E−03	1.6E−03	4.4E−01	1.3E−03	4.4E−04	8.2E−05	1.6E−01
50%	71.36	72.14	72.85	71.57	69.22	69.32	69.45	69.61	69.65	69.82	69.83	70.54	69.47	69.51	69.69	70.37	5.2E−03	1.3E−03	2.1E−04	1.1E−01	1.2E−02	5.2E−03	8.6E−04	1.8E−01	3.0E−03	5.0E−04	6.2E−05	1.4E−01
Gaussian attribute noise																												
0%	85.42	85.48	85.52	83.74	84.92	85.04	85.04	85.83	85.03	85.22	85.29	86.10	84.25	84.39	84.39	84.72	9.2E−01*	8.0E−01*	5.0E−01	3.0E−02*	9.3E−01*	9.2E−01*	9.5E−01	1.2E−02*	2.4E−02	3.4E−02	2.6E−02	7.8E−01
5%	85.12	85.27	85.29	83.45	84.26	84.26	84.28	85.29	85.02	85.03	84.99	85.70	84.15	84.06	84.01	84.37	3.4E−01	1.3E−01	1.3E−01	5.1E−02*	8.0E−01*	7.8E−01*	8.8E−01*	2.2E−02*	2.3E−02	3.4E−02	2.3E−02	6.5E−01
10%	84.49	84.62	84.55	82.89	83.47	83.47	83.40	84.01	84.30	84.33	84.34	85.12	83.57	83.39	83.23	83.76	1.1E−01	4.9E−02	7.2E−02	3.5E−01*	9.9E−01*	9.1E−01	9.1E−01	1.4E−02*	1.2E−02	1.5E−02	1.0E−02	4.8E−01
15%	84.03	84.24	84.35	82.43	83.03	83.07	83.05	83.20	83.59	83.71	83.83	84.68	83.00	82.89	82.82	83.26	1.2E−01	1.3E−01	5.4E−02	4.6E−01*	7.4E−01	6.1E−01	8.4E−01*	1.9E−02*	2.1E−02	2.0E−02	5.6E−03	3.9E−01
20%	83.33	83.52	83.65	81.83	82.04	82.13	82.16	81.86	82.83	82.93	82.98	83.94	82.23	82.13	82.10	82.45	5.0E−02	4.8E−02	1.1E−02	8.7E−01	5.1E−01	4.5E−01	4.3E−01	2.6E−02*	1.0E−02	3.0E−03	2.4E−03	2.9E−01
25%	82.62	82.92	83.16	81.41	81.07	81.28	81.53	81.32	82.18	82.44	82.54	83.40	81.78	81.77	81.72	82.20	2.2E−02	5.1E−03	4.7E−03	4.6E−01	3.2E−01	2.3E−01	2.0E−01	8.9E−02*	1.5E−02	3.7E−03	2.1E−03	3.7E−01
30%	81.85	82.15	82.37	80.56	79.84	80.03	80.27	80.59	81.44	81.57	81.70	82.58	80.96	80.88	80.89	81.39	3.3E−02	8.6E−03	6.5E−03	4.7E−01	5.0E−01	2.7E−01	3.2E−01	1.1E−01*	2.6E−02	1.5E−02	1.6E−02	4.4E−01
35%	81.03	81.32	81.54	79.75	79.12	79.28	79.58	79.84	80.58	80.80	80.84	81.61	80.22	80.21	80.22	80.62	3.5E−02	1.9E−02	1.9E−02	5.8E−01	2.8E−01	6.3E−01	4.9E−01	1.7E−01*	3.4E−02	2.6E−02	3.5E−02	7.2E−01
40%	80.34	80.87	81.04	79.32	78.37	78.58	78.70	79.00	79.64	79.97	80.01	80.83	79.27	79.28	79.19	79.88	2.9E−02	2.1E−02	1.0E−02	7.5E−01	3.2E−01	1.8E−01	2.0E−01	4.8E−01*	3.7E−02	6.1E−03	6.6E−03	5.3E−01
45%	79.45	79.71	79.98	78.25	77.66	77.88	78.04	78.09	78.92	79.12	79.10	79.89	78.55	78.58	78.46	78.92	1.3E−01	7.0E−02	2.7E−02	8.9E−01	3.5E−01	3.9E−01	2.5E−01	1.9E−01*	4.0E−02	3.0E−02	6.9E−03	6.9E−01
50%	78.49	79.08	79.21	77.86	76.73	77.07	77.23	77.37	77.97	78.12	78.23	79.03	77.59	77.64	77.60	78.15	5.9E−02	1.9E−02	2.7E−02	9.1E−01	2.0E−01	6.1E−02	9.7E−02	6.6E−01*	2.2E−02	4.0E−03	3.9E−03	5.6E−01
Robustness																												
Uniform attribute noise																												
5%	0.95	0.71	0.85	0.94	1.44	1.47	1.47	1.39	0.84	0.85	1.07	1.12	0.69	1.10	1.21	0.89	1.1E−01	3.2E−02	2.4E−01	3.2E−01	7.6E−01	3.7E−01	9.5E−01	5.9E−01*	6.5E−01	1.7E−01	3.2E−01	4.9E−01
10%	2.40	2.11	1.98	2.57	2.64	2.78	2.86	3.69	2.16	2.28	2.36	2.31	2.19	2.52	2.48	2.35	4.1E−01	4.0E−01	2.6E−01	2.0E−01	9.5E−01*	7.6E−01	5.3E−01	1.6E−01*	9.6E−01	2.3E−01	3.1E−01	8.0E−01
15%	4.23	3.86	3.76	3.63	4.82	4.90	4.86	5.88	4.17	4.23	4.31	4.30	3.89	4.25	4.36	4.16	1.4E−01	2.1E−01	2.1E−01	4.2E−02	6.6E−01	4.4E−01	3.0E−01	5.0E−01	5.5E−01*	3.3E−01	3.1E−01	4.0E−01
20%	5.54	5.02	4.80	4.68	7.17	6.87	6.70	8.03	6.03	6.10	6.19	5.90	5.68	5.99	6.06	5.65	4.2E−02	2.3E−02	2.9E−02	1.2E−02	3.1E−01	1.3E−01	8.6E−02	4.6E−01	2.8E−01	8.5E−02	8.5E−02	3.1E−01
25%	7.09	6.61	6.23	6.10	8.99	8.69	8.28	9.43	7.58	7.64	7.63	7.44	7.16	7.42	7.37	7.08	9.3E−02	4.9E−02	2.8E−02	1.5E−02	2.6E−01	1.1E−01	3.8E−02	3.5E−01	7.6E−01	1.8E−01	1.1E−01	5.2E−01
30%	8.85	8.36	8.06	7.84	11.52	11.42	11.12	11.19	9.58	9.81	9.69	9.38	9.06	9.42	9.41	9.10	1.7E−02	8.0E−03	2.6E−03	2.0E−02	2.8E−01	7.4E−02	2.7E−02	3.7E−01	6.6E−01	1.7E−01	4.4E−02	4.6E−01
35%	10.79	10.12	9.82	9.17	13.50	13.59	13.45	13.24	11.89	11.98	11.98	11.50	11.26	11.41	11.50	11.00	6.8E−03	1.3E−03	5.4E−04	8.4E−03	1.3E−01	4.0E−02	8.7E−03	5.3E−02	3.8E−01	1.4E−01	4.2E−02	2.6E−01
40%	12.44	11.67	11.23	10.74	14.73	14.63	14.44	15.14	13.59	13.68	13.68	13.40	13.07	13.26	13.04	12.78	2.3E−01	6.0E−02	3.7E−02	2.0E−02	4.0E−01	2.0E−01	1.0E−01	1.7E−01	6.5E−01	4.7E−01	3.0E−01	8.8E−01
45%	14.73	13.98	13.38	12.75	17.30	17.05	16.88	17.53	16.31	16.42	16.37	16.23	15.71	15.91	15.79	15.31	4.3E−02	2.0E−02	1.0E−02	6.4E−03	1.7E−01	6.0E−02	3.4E−02	7.2E−02	4.0E−01	2.4E−01	9.8E−02	4.4E−01
50%	16.68	15.85	15.09	14.40	18.56	18.63	18.45	19.04	17.97	17.97	18.06	18.03	17.45	17.55	17.33	16.88	1.7E−01	3.2E−02	1.0E−02	1.2E−02	2.3E−01	8.1E−02	7.8E−03	3.0E−02	4.9E−01	1.8E−01	3.4E−02	3.9E−01
Gaussian attribute noise																												
5%	0.31	0.19	0.25	0.32	0.64	0.90	0.90	0.69	−0.04	0.22	0.38	0.47	0.02	0.33	0.38	0.31	1.9E−01	6.5E−02	3.0E−02	1.6E−01	5.8E−01	4.6E−01	7.0E−01	6.9E−01*	9.0E−01	2.1E−01	4.4E−01	7.7E−01
10%	1.03	0.98	1.16	0.92	1.63	1.86	1.97	2.34	0.81	1.09	1.17	1.16	0.69	1.12	1.32	1.05	8.6E−02	8.2E−02	3.5E−01	4.5E−02	5.8E−01	6.0E−01	8.9E−01*	6.0E−01*	9.8E−01	3.9E−01	4.6E−01	5.1E−01
15%	1.54	1.39	1.32	1.48	2.05	2.26	2.31	3.31	1.59	1.76	1.70	1.62	1.33	1.62	1.73	1.58	1.3E−01	7.5E−02	1.3E−01	3.2E−02	6.3E−01	3.4E−01	3.5E−01	8.2E−01*	7.7E−01*	5.6E−01	3.5E−01	6.2E−01
20%	2.36	2.21	2.14	2.07	3.29	3.41	3.41	4.93	2.54	2.69	2.75	2.48	2.33	2.58	2.65	2.58	1.8E−02	2.0E−02	2.9E−02	5.7E−03	3.0E−01	1.2E−01	1.8E−01	8.0E−01*	4.4E−01	2.2E−01	1.4E−01	4.3E−01
25%	3.20	2.95	2.73	2.56	4.53	4.51	4.20	5.58	3.31	3.27	3.24	3.14	2.84	3.00	3.09	2.83	4.4E−02	1.1E−02	1.5E−02	1.8E−03	5.6E−01	4.0E−01	2.3E−01	4.8E−01	6.2E−01*	6.0E−01	2.5E−01	4.4E−01
30%	4.14	3.86	3.65	3.76	6.07	6.05	5.76	6.38	4.17	4.28	4.23	4.04	3.82	4.07	4.06	3.81	1.7E−01	3.6E−02	2.7E−02	7.8E−03	7.2E−01	4.2E−01	3.3E−01	3.9E−01	5.0E−01*	6.8E−01	4.9E−01	7.6E−01
35%	5.09	4.82	4.63	4.62	7.00	7.02	6.61	7.25	5.25	5.23	5.28	5.20	4.77	4.94	4.92	4.76	7.4E−02	3.7E−02	6.7E−02	2.3E−02	8.0E−01	4.9E−01	4.1E−01	5.7E−01	4.6E−01*	8.3E−01*	8.9E−01	9.6E−01*
40%	5.93	5.38	5.24	5.11	7.95	7.90	7.74	8.27	6.44	6.28	6.32	6.18	5.97	6.13	6.22	5.69	7.4E−02	5.0E−02	2.1E−02	3.4E−02	5.0E−01	2.1E−01	1.8E−01	2.5E−01	7.1E−01*	6.2E−01	2.5E−01	7.5E−01
45%	7.03	6.80	6.53	6.38	8.82	8.73	8.51	9.32	7.31	7.29	7.41	7.28	6.90	6.98	7.15	6.91	2.1E−01	1.3E−01	1.8E−01	5.5E−02	8.9E−01	4.9E−01	3.2E−01	4.4E−01	8.3E−01*	9.3E−01	4.8E−01	6.0E−01
50%	8.14	7.50	7.43	6.68	9.94	9.68	9.47	10.10	8.43	8.47	8.42	8.23	8.04	8.09	8.17	7.78	9.2E−02	4.6E−02	5.2E−02	5.8E−02	6.1E−01	1.9E−01	2.2E−01	1.6E−01	7.7E−01*	5.8E−01	4.8E−01	2.9E−01

- The average results show that, in MCS3-*k*, MAJ and NB are the best methods, whereas BKS and W-MAJ are the worst ones. In MCS5, NB is the best method; BKS and MAJ are the following methods and W-MAJ is placed at the last positions at many noise levels.
- The *p*-values show that MAJ is better than W-MAJ from 20% onwards. It is better than NB for MCS3-3 and MCS3-5 from a noise level 20–25% and for MCS5 at the highest noise levels (45–50%), but this does not occur with MCS3-1. MAJ is neither better than BKS (only at some intermediate-high noise levels).

(b) **Gaussian attribute noise.**

- In MCS3-*k*, MAJ and NB are the best methods in average results, followed by BKS and W-MAJ. In MCS5, the ranking is composed by NB, and BKS, MAJ and W-MAJ (these last three obtain the lowest average results at most of the noise levels).
- The *p*-values show that MAJ is not better than NB nor BKS. It is generally better than W-MAJ, even though they are equivalent at some isolated noise levels.

2. **Robustness results.**

(a) **Uniform attribute noise.**

- The average results show that W-MAJ is the best method, followed by NB, BKS and MAJ.
- Attending to the *p*-values, we observe that, in general, MAJ is better than W-MAJ (from 20% onwards), even though they are equivalent at some high noise levels considering MCS3-1. MAJ is also better than NB for medium–high noise levels (except for MCS3-1, where both are equivalent). BKS is equivalent to MAJ, except for MCS3-5 and MCS3-5 where MAJ is better than BKS from 20% onwards.

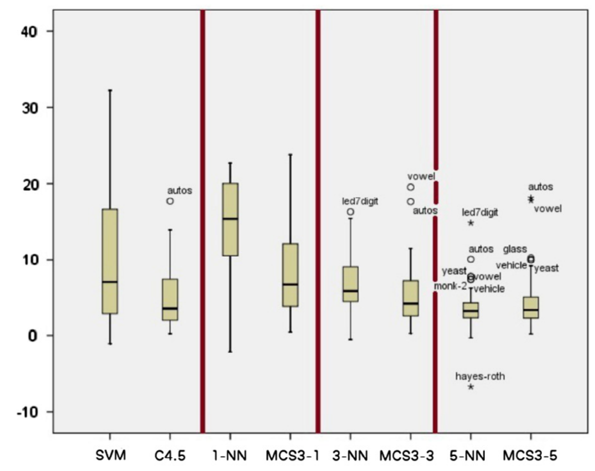
(b) **Gaussian attribute noise.**

- W-MAJ obtains the best average results. It is followed by NB, and the worst combination methods are BKS and MAJ.
- The *p*-values show that MAJ is usually better than W-MAJ (except at some isolated noise level where both are equivalent). In general, MAJ is equivalent to NB and BKS, even though NB and BKS are better than MAJ for some MCSs (NB is better for MCS5 and BKS is better for MCS3-1).

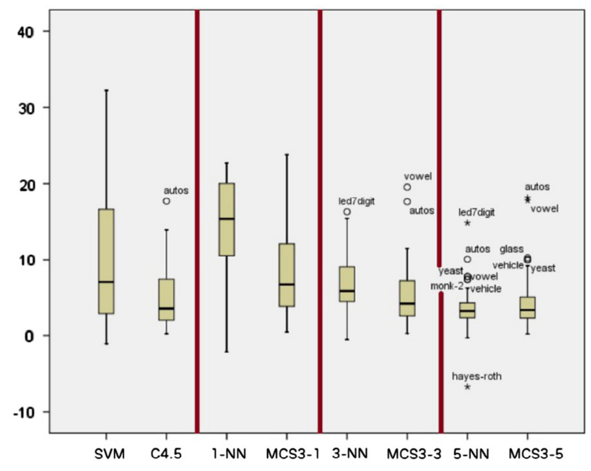
7. On the usage of Multiple Classifier Systems with noisy data: Lessons learned

In this paper a thorough empirical study has been performed in order to find the conditions under which the MCSs studied (the three MCS-*k*, the MCS5 and BagC4.5) behave well with noisy data, as well as the terms of this improvement. From the results shown in the previous section and their corresponding analysis, several lessons can be learned:

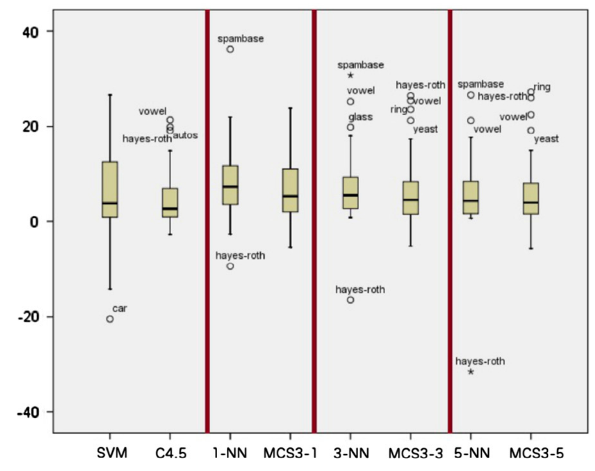
1. **The behavior of the studied MCSs built with heterogeneous classifiers depend on the behavior of their individual classifiers with noisy data.** This situation can be clearly observed in the obtained results where, for example, a higher value of *k*-and therefore a better performance and robustness of *k*-NN-implies that the corresponding MCS3-*k* attains a better performance and robustness. This higher robustness is particularly notable with the most disruptive noise schemes (uniform class and attribute noise) and the highest noise levels. However, BagC4.5 gain robustness against several noise types (such as the uniform class or attribute noises) which C4.5 is weaker with.
2. **The study of behavior of each single classifier considered to build an MCS in noisy environments is an important issue.** For example, SVM obtains the highest performance without noise but is the most affected when class noise is considered. Hence, it is recommended to execute all the available classifiers to check which are more suitable for the type and level of noise, finally choosing those performing better in order to build the MCS.
3. **Different types of noise have a very different disruptiveness and, therefore, they do not affect the performance in the same way.** For the classifiers studied, the uniform class noise is generally more disruptive than the pairwise class noise-except in the case of BagC4.5 and for the medium–highest noise levels with those classifiers involving *k*-NN with $k = 3,5$. We must not forget that uniform class noise can be more disruptive because it affects more examples than the pairwise class noise. The uniform attribute noise is more disruptive than the Gaussian attribute noise. Moreover, the uniform class noise is more disruptive than the uniform attribute noise.
4. **On the behavior of the studied MCSs with data with class noise.** The performance of the studied MCSs compared to that of their individual classifiers is better with the uniform class noise (the more disruptive scheme) than with the pairwise noise, since the differences between MCSs and their base classifiers are more accentuated with the uniform noise. The robustness results show a similar conclusion, with the MCSs generally more robust with the uniform class noise and obtaining in this case lower *p*-values in the comparisons with respect to their individual classifiers.
5. **On the behavior of the studied MCSs with data with attribute noise.** The studied MCSs perform better than their individual classifiers with the less disruptive attribute noise scheme (Gaussian) and slightly worse with the more disruptive one (uniform). Similar conclusions are drawn from the robustness results. The exception is BagC4.5, which perform clearly better than C4.5 with both schemes.



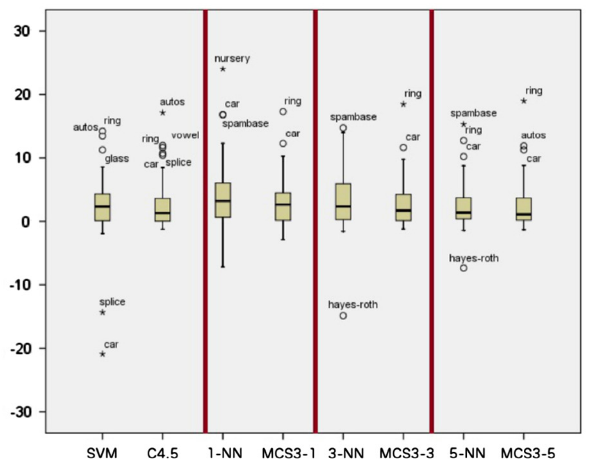
(a) Uniform class noise.



(b) Pairwise class noise.



(c) Uniform attribute noise.



(d) Gaussian attribute noise.

Fig. 1. Box-plots representing the distribution of the RLA results with a 25% noise level for MCS3-k and its individual classifiers. The RLA metric (Eq. (2)) is used to estimate the robustness of a classifier at a noise level comparing the accuracy of the classifier at that noise level with respect to that of the classifier trained without additional noise.

6. **Different types of noise affect the studied MCSs differently.** The studied MCSs built with heterogeneous classifiers work worse with attribute noise than with class noise. With attribute noise, MCSs significantly improve the results of the more robust single method, that is, C4.5, if the noise is not very disruptive (Gaussian) or if the noise level is low enough. However, it is important to note that BagC4.5 works well with both types of noise except with the pairwise class noise, in which it does not perform as accurate as with other types of noise.
7. **Even though the studied MCSs built with heterogeneous classifiers may lead to an improvement in the performance results working with noisy data, they will never be statistically more robust than the most robust of its individual classifiers.** The improvement of performance will depend on many factors, as mentioned above, but it is likely to occur if the proper base classifiers are selected. However, an improvement in the robustness of the methods seems to be difficult. In fact, the robustness of the MCS built can be thought of as the average of the robustness of each of its single classifiers. Fig. 1 shows the distribution of the robustness results of each single classification algorithm and each MCS3-*k* for each noise scheme at the medium noise level considered in this paper; that is, 25%. As it can be appreciated in this figure, the distribution of the robustness values of each MCS3-*k* is always between the distribution of the least and most robust single classifiers–even though this fact is more observable in the robustness results with class noise. However, from the results obtained we observe that using BagC4.5 the robustness of its only single classifier C4.5 can be improved.
8. **The majority vote scheme is a powerful alternative against other decision combination methods for the MCS studied when working with noisy data.** Even though MAJ is outperformed with some types of noise and at some noise levels, it performs relatively well for most of the noise types. Since the type and level of noise is usually unknown in real-world data, its usage can be recommended.

The main reason for the better performance of MCSs with noisy data can be attributed to the increment of the capability of generalization when an MCS is built. Each single classifier is known to make errors, but since they are different, that is, they have different behaviors over different parts of the domain, misclassified examples are not necessarily the same [21], and the same is true for noisy instances. This fact enables MCSs to achieve a better generalization from the examples of a problem and leads to better avoiding the overfitting of noisy data and, therefore, to obtain more accurate solutions. Some classifiers can tolerate the noise better or are more sensitive than others, depending on the different parts of the input/output space. Combining several classifiers, preferably noise-tolerant in most of the input/output space, may lead them to complement each other, obtaining a higher noise-robustness, or at least better generalization capabilities. Thus, if a set of noisy examples does not affect the learning of a concrete set of classifiers, their predictions will not be hindered.

8. Concluding remarks

This paper analyzes how well several MCSs behave with noisy data by means the realization of a huge experimental study. In order to do this, both the performance and the noise-robustness of different MCSs have been compared with respect to their single classification algorithms. A large number of noisy datasets have been created considering different types, schemes and levels of noise to perform these comparisons.

The results obtained have shown that the MCSs studied do not always significantly improve the performance of their single classification algorithms when dealing with noisy data, although they do in the majority of cases (if the individual components are not heavily affected by noise). The improvement depends on many factors, such as the type and level of noise. Moreover, the performance of the MCSs built with heterogeneous classifiers depends on the performance of their single classifiers, so it is recommended that one studies the behavior of each single classifier before building the MCS. Generally, the MCSs studied are more suitable for class noise than for attribute noise. Particularly, they perform better with the most disruptive class noise scheme (uniform one) and with the least disruptive attribute noise scheme (Gaussian one). However, BagC4.5 works well with both types of noise, with the exception of the pairwise class noise, where its results are not as accurate as with other types of noise.

The robustness results show that the studied MCS built with heterogeneous classifiers will not be more robust than the most robust among their single classification algorithms. In fact, the robustness can always be shown as an average of the robustnesses of the individual methods. The higher the robustnesses of the individual classifiers are, the higher the robustness of the MCS is. Nevertheless, BagC4.5 is again an exception: it becomes more robust than its single classifier C4.5.

The study of several decisions combination methods show that the majority vote scheme is a simple yet powerful alternative to other techniques when working with noisy data.

Acknowledgments

Supported by the Spanish Ministry of Science and Technology under Projects TIN2011-28488 and TIN2010-15055, and also by the Regional Project P10-TIC-6858. José A. Sáez holds an FPU scholarship from the Spanish Ministry of Education and Science.

References

[1] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, S. García, L. Sánchez, F. Herrera, KEEL data-mining software tool: data set repository, integration of algorithms and experimental analysis framework, Journal of Multiple-Valued Logic and Soft Computing 17 (2011) 255–287.
[2] J. Alcalá-Fdez, L. Sánchez, S. García, M. del Jesus, S. Ventura, J. Garrell, J. Otero, C. Romero, J. Bacardit, V. Rivas, J. Fernández, F. Herrera, KEEL: a software tool to assess evolutionary algorithms for data mining problems, Soft Computing-A Fusion of Foundations, Methodologies and Applications 13 (2009) 307–318.
[3] C. Catal, O. Alan, K. Balkan, Class noise detection based on software metrics and ROC curves, Information Sciences 181 (2011) 4867–4877.
[4] W.W. Cohen, Fast effective rule induction, in: Proceedings of the Twelfth International Conference on Machine Learning, Morgan Kaufmann Publishers, 1995, pp. 115–123.
[5] C. Cortes, V. Vapnik, Support vector networks, Machine Learning 20 (1995) 273–297.
[6] J. Demšar, Statistical comparisons of classifiers over multiple data sets, Journal of Machine Learning Research 7 (2006) 1–30.
[7] T. Dietterich, An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization, Machine Learning 40 (2000) 139–157.
[8] R.O. Duda, P.E. Hart, D.G. Stork, Pattern Classification, second ed., John Wiley, New York, 2001.
[9] J. Fűrnkranz, Noise-tolerant windowing, Journal of Artificial Intelligence Research 8 (1997) 129–164.
[10] S. García, A. Fernández, J. Luengo, F. Herrera, Advanced nonparametric tests for multiple comparisons in the design of experiments in computational intelligence and data mining: experimental analysis of power, Information Sciences 180 (2010) 2044–2064.
[11] S. García, F. Herrera, An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons, Journal of Machine Learning Research 9 (2008) 2677–2694.
[12] R.M. Haralick, The table look-up rule, Communications in Statistics-Theory and Methods A5 (1976) 1163–1191.
[13] M.A. Hernández, S.J. Stolfo, Real-world data is dirty: data cleansing and the merge/purge problem, Data Mining and Knowledge Discovery 2 (1998) 9–37.
[14] T.K. Ho, Multiple classifier combination: lessons and next steps, in: A. Kandel, E. Bunke (Eds.), Hybrid Methods in Pattern Recognition, World Scientific, 2002, pp. 171–198.
[15] T.K. Ho, J.J. Hull, S.N. Srihari, Decision combination in multiple classifier systems, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (1994) 66–75.
[16] J.L. Hodges, E.L. Lehmann, Ranks methods for combination of independent experiments in analysis of variance, Annals of Mathematical Statistics 33 (1962) 482–497.

[17] S. Holm, A simple sequentially rejective multiple test procedure, Scandinavian Journal of Statistics 6 (1979) 65–70.
[18] Y.S. Huang, C.Y. Suen, A method of combining multiple experts for the recognition of unconstrained handwritten numerals, IEEE Transactions on Pattern Analysis and Machine Intelligence 17 (1995) 90–93.
[19] P.J. Huber, Robust Statistics, John Wiley and Sons, New York, 1981.
[20] T. Khoshgoftaar, J. Van Hulse, A. Napolitano, Comparing boosting and bagging techniques with noisy and imbalanced data, IEEE Transactions on Systems, Man and Cybernetics, Part A: Systems and Humans 41 (2011) 552–568.
[21] J. Kittler, M. Hatef, R. Duin, J. Matas, On combining classifiers, IEEE Transactions on Pattern Analysis and Machine Intelligence 20 (1998) 226–239.
[22] I. Kononenko, M. Kukar, Machine Learning and Data Mining: Introduction to Principles and Algorithms, Horwood Publishing Limited, 2007.
[23] L. Kuncheva, Combining Pattern Classifiers: Methods and Algorithms, Wiley, 2004.
[24] L.I. Kuncheva, Diversity in multiple classifier systems, Information Fusion 6 (2005) 3–4.
[25] R. Maclin, D. Opitz, An empirical evaluation of bagging and boosting, in: Proceedings of the Fourteenth National Conference on Artificial Intelligence and Ninth Conference on Innovative Applications of Artificial Intelligence, pp. 546–551.
[26] E. Mandler, J. Schuermann, Combining the classification results of independent classifiers based on the Dempster/Shafer theory of evidence, in: E.S. Gelsema, L.N. Kanal (Eds.), Pattern Recognition and Artificial Intelligence, North-Holland, Amsterdam, 1988, pp. 381–393.
[27] U. Maulik, D. Chakraborty, A robust multiple classifier system for pixel classification of remote sensing images, Fundamenta Informaticae 101 (2010) 286–304.
[28] V.D. Mazurov, A.I. Krivonogov, V.S. Kazantsev, Solving of optimization and identification problems by the committee methods, Pattern Recognition 20 (1987) 371–378.
[29] G.J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, John Wiley and Sons, 2004.
[30] R.K. Nath, Fingerprint recognition using multiple classifier system, Fractals 15 (2007) 273–278.
[31] D. Nettleton, A. Orriols-Puig, A. Fornells, A study of the effect of different types of noise on the precision of supervised learning techniques, Artificial Intelligence Review 33 (2010) 275–306.
[32] R. Polikar, Ensemble based systems in decision making, IEEE Circuits and Systems Magazine 6 (2006) 21–45.
[33] B. Qian, K. Rasheed, Foreign exchange market prediction with multiple classifiers, Journal of Forecasting 29 (2010) 271–284.
[34] J.R. Quinlan, Induction of decision trees, in: Machine Learning, pp. 81–106.
[35] J.R. Quinlan, The effect of noise on concept learning, in: Machine Learning: An Artificial Intelligence Approach, Morgan Kaufmann Publishers, 1986, pp. 149–166.
[36] J.R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, San Francisco, CA, USA, 1993.
[37] C. Seiffert, T.M. Khoshgoftaar, J. Van Hulse, A. Folleco, An empirical study of the classification performance of learners on imbalanced and noisy software quality data, Information Sciences (2011), http://dx.doi.org/10.1016/j.ins.2010.12.016.
[38] L. Shapley, B. Grofman, Optimizing group judgmental accuracy in the presence of interdependencies, Public Choice 43 (1984) 329–343.
[39] C.M. Teng, Correcting noisy data, in: Proceedings of the Sixteenth International Conference on Machine Learning, Morgan Kaufmann Publishers, San Francisco, CA, USA, 1999, pp. 239–248.
[40] C.M. Teng, Polishing blemishes: issues in data correction, IEEE Intelligent Systems 19 (2004) 34–39.
[41] D.M. Titterington, G.D. Murray, L.S. Murray, D.J. Spiegelhalter, A.M. Skene, J.D.F. Habbema, G.J. Gelpke, Comparison of discriminant techniques applied to a complex data set of head injured patients, Journal of the Royal Statistical Society, Series A (General) 144 (1981) 145–175.
[42] R.Y. Wang, V.C. Storey, C.P. Firth, A framework for analysis of data quality research, IEEE Transactions on Knowledge and Data Engineering 7 (1995) 623–640.
[43] K.D. Wemecke, A coupling procedure for the discrimination of mixed data, Biometrics 48 (1992) 497–506.
[44] D.R. Wilson, T.R. Martinez, Improved heterogeneous distance functions, Journal of Artificial Intelligence Research 6 (1997) 1–34.
[45] M. Woźniak, M. Graña, E. Corchado, A survey of multiple classifier systems as hybrid systems, Information Fusion (2013). http://dx.doi.org/10.1016/j.inffus.2013.04.006.
[46] X. Wu, Knowledge Acquisition from Databases, Ablex Publishing Corp., Norwood, NJ, USA, 1996.
[47] X. Wu, V. Kumar, J. Ross Quinlan, J. Ghosh, Q. Yang, H. Motoda, G.J. McLachlan, A. Ng, B. Liu, P.S. Yu, Z.H. Zhou, M. Steinbach, D.J. Hand, D. Steinberg, Top 10 algorithms in data mining, Knowledge and Information Systems 14 (2007) 1–37.
[48] X. Wu, X. Zhu, Mining with noise knowledge: error-aware data mining, IEEE Transactions on Systems, Man, and Cybernetics 38 (2008) 917–932.
[49] L. Xu, A. Krzyzak, C.Y. Suen, Methods of combining multiple classifiers and their applications to handwriting recognition, IEEE Transactions on Systems, Man, and Cybernetics 22 (1992) 418–435.
[50] Y. Zhang, X. Wu, Integrating induction and deduction for noisy data mining, Information Sciences 180 (2010) 2663–2673.
[51] S. Zhong, T.M. Khoshgoftaar, N. Seliya, Analyzing software measurement data with clustering techniques, IEEE Intelligent Systems 19 (2004) 20–27.
[52] X. Zhu, X. Wu, Class noise vs. attribute noise: a quantitative study, Artificial Intelligence Review 22 (2004) 177–210.
[53] X. Zhu, X. Wu, Q. Chen, Eliminating class noise in large datasets, in: Proceeding of the Twentieth International Conference on Machine Learning, pp. 920–927.
[54] X. Zhu, X. Wu, Y. Yang, Error detection and impact-sensitive instance ranking in noisy datasets, in: Proceedings of the Nineteenth National Conference on Artificial Intelligence, AAAI Press, 2004, pp. 378–383.