A First Study on Decomposition Strategies with Data with Class Noise Using Decision Trees

José A. Sáez[1], Mikel Galar[2], Julián Luengo[3], and Francisco Herrera[1]

[1] Department of Computer Science and Artificial Intelligence of the University of Granada, CITIC-UGR, Granada, Spain, 18071
{smja,herrera}@decsai.ugr.es

[2] Department of Automática y Computación, Universidad Pública de Navarra, Pamplona, Spain, 31006
mikel.galar@unavarra.es

[3] Department of Civil Engineering, LSI, University of Burgos, Burgos, Spain, 09006
jluengo@ubu.es

Abstract. Noise is a common problem that produces negative consequences in classification problems. When a problem has more than two classes, that is, a multi-class problem, an interesting approach to deal with noise is to decompose the problem into several binary subproblems, reducing the complexity and consequently dividing the effects caused by noise into each of these subproblems. This contribution analyzes the use of decomposition strategies, and more specifically the One-vs-One scheme, to deal with multi-class datasets with class noise. In order to accomplish this, the performance of the decision trees built by C4.5, with and without decomposition, are studied. The results obtained show that the use of the One-vs-One strategy significantly improves the performance of C4.5 when dealing with noisy data.

Keywords: Noisy Data, Class Noise, One-vs-One, Decomposition Strategies, Ensembles, Classification.

1 Introduction

Any classification problem [1] consists of m training patterns, characterized by n attributes A_i, $i = 1, \ldots, n$, which are either numerical or nominal, being $\mathbb{D}_i$ their corresponding domains. Thus, an example $\mathbf{x}$ is represented as an n-dimensional attribute vector

$$\mathbf{x} = (x_1, \ldots, x_n) \in \mathbb{D} = \mathbb{D}_1 \times \cdots \times \mathbb{D}_n \ .$$

Each example is labeled with one of M possible classes $\mathbb{L} = \{\lambda_1, \ldots, \lambda_M\}$. Many current real-world classification problems, such as cancer classification [2] or the recognition of fingerprints [3], are characterized by having more than two classes, that is $M > 2$. These problems are formally known as multi-class classification problems.

E. Corchado et al. (Eds.): HAIS 2012, Part II, LNCS 7209, pp. 25–35, 2012.

Classification algorithms aim to extract the implicit knowledge from previously known instances of these problems by creating a model, called a classifier, that generalizes the peculiarities of the set of labeled examples and is capable of predicting the class for previously unobserved examples. Hence, the classification accuracy of a classifier is directly influenced by the quality of the training data used to build the model. Data quality depends on several components [4], for instance, the source of that data and the input of the data, inherently subject to errors, among others. Real-world datasets rarely avoid this type of errors and they usually contain corrupted data that may hinder the interpretations, decisions and therefore, the models created from the data.

Generally, the more classes in a problem, the more complex the decision boundaries are. Moreover, the presence of noise in such problems adds an extra complexity. Traditionally, decomposing a multi-class problem into several binary, easier to solve subproblems, has been related to obtaining a good performance when data are affected by noise –although this issue has not been explicitly addressed yet. In such a way, the complexity of the original problem is decreased, and as a consequence, noisy instances are divided into each subproblem, which also decreases the noise effect on the final performance of the classifier. These techniques are called binary decomposition strategies [8]. The *One-vs-One* (OVO) [9] and *One-vs-All* (OVA) [10] schemes are the most studied in the literature. OVO is based on dividing the problem into as many binary problems as possible combinations between pairs of classes, while OVA learns a classifier for each class. Generally, OVO outstands over OVA as reflected in the literature [11], [12], [13].

In this work our aim is to analyze the suitability of the OVO binary decomposition strategy with training data suffering from class noise. We will study the differences between OVO and non-OVO (baseline) classifiers built by C4.5 [6] through an analysis of their accuracy, which we will also contrast using the proper statistical tests as recommended in the specialized literature [14]. Notice that C4.5 is capable of handling multiple classes inherently; hence, we will be able to compare the OVO scheme with their baseline performances. In order to validate our hypothesis and to extract meaningful conclusions, we will prepare an experimental framework considering 21 real-world datasets. Four different levels of noise are introduced in the class labels in the training partitions: 5%, 10%, 15% and 20%. Thus, 84 new synthetic datasets with class noise will be created. The test sets will remain unchanged in order to check which strategy, OVO or baseline, performs better in the presence of noisy data.

The rest of this contribution is organized as follows. Section 2 presents an introduction to classification with noisy data. Section 3 is devoted to the motivations for the use of binary decomposition strategies in multi-class classification problems, recalling the OVO decomposition scheme. Next, Section 4 describes the experimental framework. Section 5 includes the analysis of the experimental results obtained by the classifiers with and without the use of the OVO decomposition scheme. Finally, in Section 6 we present our concluding remarks.

2 Classification with Noisy Data

Real-world data is never perfect and often suffers from corruptions that harm the interpretations of the data, the models created and the decisions made. In classification, noise can negatively affect the system performance in terms of classification accuracy, building time, size and interpretability of the classifier built [5].

The quality of any dataset is determined by a large number of components as described in [4]. Some of these are the source of the data and the input of the data, which are inherently subject to error.

Class labels and attributes are two information sources which can influence the quality of a classification dataset. The quality of the class labels represents whether the class of each instance is correctly assigned; and the quality of the attributes indicates how well the attributes characterize instances for classification purposes. Based on these two information sources, we can distinguish two types of noise in a given dataset [5]:

1. *Class noise.* These errors occur when an instance belongs to the incorrect class. Class noise can be attributed to several causes, including subjectivity during the labeling process, data entry errors, or inadequacy of the information used to label each object. There are two possible types of class noise:
   - Contradictory examples: the same examples appear more than once and are labeled with different classes.
   - Misclassifications: instances are labeled with the wrong classes [18].
2. *Attribute noise.* It is used to refer to corruptions in the values of one or more attribute of instances in the dataset. Examples of attribute noise include: erroneous attribute values, missing or unknown attribute values, and incomplete attributes or “do not care” values.

The two most common approaches to noisy data in the literature are:

- *Robust learners.* They are characterized by being less influenced by noisy data. An example of a robust learner is the C4.5 algorithm [6]. C4.5 uses pruning strategies to reduce the chances of trees being built with noise in the training data. However, when the noise level becomes relatively high, even a robust learner may obtain a poor performance.
- *Noise preprocessing techniques.* They are classifier-independent and try to remove the negative impact of noise in the datasets prior to creating a model over the original data. Among these techniques, the most well-known methods are noise filtering ones [7].

In this contribution, we focus on class noise because it is very common in real-world data [5], [18]. These errors can be produced in situations where different classes have similar symptoms, as generally happens on the class boundaries. We compare the performance of the C4.5 robust learner considering or not the use of decomposition. We want to verify that the effect of class noise on the accuracy of the decision trees created by C4.5 is lower considering the use of OVO, even if this classification algorithm is a robust learner.

3 Addressing Multi-class Classification by Decomposition

Classification tasks involving more than two categories or classes, commonly known as multi-class classification problems, are frequent in real-world problems. Multi-class problems are more general than the special case considering only two classes (binary classification problems).

A multi-class classification problem is intrinsically more complex than a binary one since the generated classifier must be able to separate the data into a higher number of categories, which increases the chances of incorrect classifications (in a two-class balanced problem, the probability of a correct random classification is $1/2$, whereas in a multi-class problem it is $1/M$).

In order to reduce the complexity of the original problems and/or to be able to use binary classification techniques to solve multi-class classification problems, in the literature two approaches have been adopted:

- Adaptation of the internal operations of the learning algorithm.
- Decomposition of the multi-class problem into a set of easier to solve two-class problems.

The extension of a binary learning algorithm to a multi-class version may be very difficult to perform in many cases [15]. Therefore, it is more common to use the alternative which decomposes the multi-class problem into binary subproblems, a strategy called decomposition.

3.1 Decomposition Strategies for Multi-class Problems

Several motivations for the use of binary decomposition strategies in multi-class classification problems can be found in the literature [11], [12]. For example, in [12], the reduction of the complexity involved in the classes' separation when using a decomposition approach was shown. Also in [16], the authors point out the advantages of the use of binary decompositions when the classification errors for different classes have distinct costs. This way, the binary predictors generated may impose preferences for some of the classes. Decomposition also opens up new possibilities for the use of parallel processing, since the binary subproblems are independent and can be solved with different processors.

Dividing a problem into several new problems which are then independently solved implies the need for a second phase where the outputs of each problem have to be aggregated. Therefore, decomposition includes two steps:

1. *Problem division.* In this phase, the problem is decomposed into several binary subproblems which are solved by independent binary classifiers, called base classifiers [12]. Different decomposition strategies can be found in the literature [8], the most common strategies are OVO [9] and OVA [10], as discussed above.
2. *Combination of the outputs* [11]. In this phase, the different outputs of the binary classifiers are aggregated in order to output the final class prediction. The simplest method is a voting strategy where each classifier gives a vote,

and the final prediction is given by the class achieving the largest amount of votes.

In this contribution, we consider the OVO decomposition strategy due to its several advantages shown in the literature [11], [12]:

- OVO creates simpler borders between classes than OVA. This is one of the main advantages of OVO that we want to exploit when training with noisy data. In such a way, the noise's corruptions in these regions will be less notable and the classifiers will be less influenced. Moreover, as OVO only distinguishes between two classes, if the noisy examples do not belong to one of the two classes that have been learned to be distinguished by a concrete classifier, this classifier will not be affected by noise and its predictions will not be altered.
- OVO generally obtains a higher classification accuracy and a shorter training time than OVA because it creates easier and smaller problems.
- OVO has less tendency to create imbalanced datasets which can be counterproductive [13].

In [11], an exhaustive study comparing different methods to combine the outputs of the base classifiers in the OVO and OVA strategies has been developed. However, the most used combination, also used in our experiments, is the voting strategy already mentioned.

3.2 One-vs-One Decomposition Scheme

The OVO decomposition strategy is based on dividing a classification problem with M classes, $\mathbb{L} = \{\lambda_1, \ldots, \lambda_M\}$, into $M(M-1)/2$ binary problems. Each new subproblem only considers the examples of the training data corresponding to a different pair of classes (λ_i, λ_j), with $i < j$.

In the learning phase, a binary classifier is created for each problem, which is capable of distinguishing between a different pair of classes. In the validation phase, an example is presented to each one of the binary classifiers. This way, each classifier discriminating between classes λ_i and λ_j provides a confidence degree $r_{ij} \in [0,1]$ in favor of the former class, and another confidence degree in favor of the latter $r_{ji} \in [0,1]$ (if the classifier does not provide the latter, it is computed by $r_{ji} = 1 - r_{ij}$). These outputs are represented by a score matrix R:

$$R = \begin{pmatrix} - & r_{12} & \cdots & r_{1m} \\ r_{21} & - & \cdots & r_{2m} \\ \vdots & & & \vdots \\ r_{m1} & r_{m2} & \cdots & - \end{pmatrix}. \tag{1}$$

The final output of the system is derived from the score matrix by different aggregation models. As we have previously mentioned, the voting strategy is the simplest method:

$$Class = \arg\max_{i=1,\ldots,m} \sum_{1 \le j \ne i \le m} s_{ij} \tag{2}$$

where s_{ij} is 1 if $r_{ij} > r_{ji}$ and 0 otherwise. This strategy has shown a competitive behavior with different classifiers [11] obtaining similar results in comparison with more complex strategies.

Although the number of classifiers is of M^2 order, as each classifier is only trained with examples from two classes, the required time is distributed, and hence is usually low. However, there is also a drawback: when a new example is submitted to all the classifiers, some of them may not have seen a similar instance before, so their output would not be significant; these cases are called non-competent examples [17]. In any case, OVO aggregations usually suppose that the base classifiers will be correctly predicted if the new pattern is one of the considered pairs of classes, and therefore, considering a voting strategy, the class with the largest number of votes will be the correct class.

4 Experimental Framework

In this section, we present the details of the experimentation developed in this contribution. First, in Subsection 4.1, we describe the base datasets of our experimentation. Then, we show how to build up the noisy datasets from the base ones in Subsection 4.2. Finally, the methodology for the analysis of the results is explained in Subsection 4.3.

4.1 Base Datasets

The experimentation is based on 21 real-world multi-class classification problems from the UCI repository[1]. Table 1 shows the datasets sorted by the number of classes (#Cla). Moreover, for each dataset, the number of instances (#Ins) and the number of attributes (#Att) along with the number of real, integer and nominal attributes (R/I/N) are presented. Some of the largest data-sets (nursery, page-blocks, penbased, satimage, shuttle and led7digit) were stratified at 10% in order to reduce the computational time required for training. For datasets containing missing values (automobile and dermatology), these instances with missing values were removed from the dataset before the partitioning.

4.2 Inducing Noise in Datasets

The initial amount of noise present in the previous datasets is unknown so we cannot make any assumption about it. We need to control in some way the amount of noise in each dataset. This will help us to check how a higher or a lower amount of noise affects the models obtained by the classification algorithms. For these reasons, we systematically and independently add noise to each dataset, as proposed in [5].

In order to introduce a level of class noise of $x\%$ in a dataset, we use a pairwise scheme as indicated in [18]: given a pair of classes (X, Y), with X the majority

[1] http://archive.ics.uci.edu/ml/datasets.html

Table 1. Summary description for classification datasets

Dataset	#Cla	#Ins	#Att (R/I/N)	Dataset	#Cla	#Ins	#Att (R/I/N)
balance	3	625	4 (4/0/0)	glass	7	214	9 (9/0/0)
contraceptive	3	1 473	9 (0/9/0)	satimage	7	643	36 (0/36/0)
iris	3	150	4 (4/0/0)	segment	7	2 310	19 (19/0/0)
splice	3	319	60 (0/0/60)	shuttle	7	2 175	9 (0/9/0)
thyroid	3	720	21 (6/15/0)	zoo	7	101	16 (0/0/16)
wine	3	178	13 (13/0/0)	ecoli	8	336	7 (7/0/0)
nursery	5	1 269	8 (0/0/8)	led7digit	10	500	7 (7/0/0)
page-blocks	5	547	10 (4/6/0)	penbased	10	1 099	16 (0/16/0)
automobile	6	150	25 (15/0/10)	yeast	10	1 484	8 (8/0/0)
dermatology	6	358	34 (0/34/0)	vowel	11	990	13 (10/3/0)
flare	6	1 066	11 (0/0/11)				

class and Y the second majority class, and a noise level $x\%$, an instance with the label X has a probability of $x\%$ to be incorrectly labeled as Y. As indicated in [5], this scheme is appropriate because it is more likely that only certain types of classes are mislabeled.

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

1. A level of noise $x\%$ 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 having the same examples per fold.
3. We use a cross-validation scheme for new noisy datasets, building the training partitions with the noisy copy, and the test partitions with the original dataset.

The accuracy estimation of each classifier in a dataset is obtained by means of 5 runs of a stratified 5-fold cross-validation. The dataset is divided into 5 partition sets with equal numbers of examples and maintaining the proportion between classes in each fold. Each partition set is used as a test set for the model learned from the four remaining partitions. This procedure is repeated 5 times. We use 5 partitions as if each partition has a large number of examples, the noise effects will be more notable, facilitating their analysis.

Introducing noise into training partitions makes it possible to observe how noise affects the test accuracy of the classifiers when training with noisy data. The accuracy of the model built over the original training set without additional noise can act as a reference value. In this way, we can observe how the accuracy of the models built with noisy training sets is more or less affected with respect to this value by the noise effect.

From the 21 base datasets from the UCI repository we have created a large collection of new noisy datasets. We have studied the levels of noise: $x = 5\%$, $x = 10\%$, $x = 15\%$ and $x = 20\%$. Therefore, 84 datasets with class noise were created.

4.3 Analysis Methodology

The main aim of this contribution is to check whether the use of the OVO binary decomposition strategy improves the classification performance in multi-class datasets affected by class noise. We will study the advantages provided by this strategy against not using it in this framework. For this reason, we consider the C4.5 algorithm which can deal with multi-class problems directly, and we use the OVO scheme in order to find whether there are improvements with respect to the original algorithm that does not use it.

In order to be able to make this comparison, we use the mean accuracy provided by C4.5 over the test sets for each level of induced noise, defined as its performance averaged across all classification problems. Along with the test accuracy, we use the Wilcoxon signed ranks statistical test [14]. This is a non-parametric pairwise test that aims to detect significant differences between two sample means; that is, the behavior of the two implicated algorithms in the comparison. Statistical analysis needs to be carried out in order to find significant differences among the results obtained by the studied methods. Accordingly, we do not only consider the mean accuracy, but we also take into account the statistical differences. Therefore, our conclusions are not only based on averaged mean results. For each level of noise, we compare C4.5 using OVO versus C4.5 trained with the complete dataset with the Wilcoxon test and we obtain the p-values associated with these comparisons.

5 Analysis of the One-vs-One Decomposition Strategy with Data with Class Noise

In this section we analyze the performance of C4.5 using the OVO decomposition with respect to its baseline results when dealing with data with class noise. Table 2 shows the test accuracy results of C4.5 in each single dataset (with and without OVO) and also the mean test accuracy as an indicator at each noise level. Table 3 shows the associated p-values of the Wilcoxon test between the OVO and non-OVO version at each noise level.

From these two tables of results we should stress several points:

- The good performance of C4.5 when using the OVO strategy must be highlighted, since the test accuracy increases with respect to the absence of decomposition.
- Although C4.5 is considered a robust learner tolerant to class noise, the binary decomposition into subproblems causes the algorithm to achieve better accuracy rates than baseline at all noise levels.
- Moreover, as shown in Table 3, we can observe from the associated p-values that there are significant differences in the results of C4.5 when using OVO with respect to the baseline results, in favor of OVO. This occurs for all noise levels.

Table 2. Test accuracy results on each single dataset with class noise. The best results between the OVO and the non-OVO version for each noise level have been stressed in bold.

	C4.5 without decomposition					C4.5 with OVO				
Noise %	**0%**	**5%**	**10%**	**15%**	**20%**	**0%**	**5%**	**10%**	**15%**	**20%**
autos	76.7339	78.0645	71.0484	72.9435	76.6935	**80.5242**	**79.8589**	**77.3387**	**77.3185**	**77.3790**
balance	77.2800	77.4400	77.6000	76.8000	76.9600	**80.1600**	**80.3200**	**81.4400**	**79.6800**	**78.2400**
contraceptive	**52.6798**	52.3406	50.6415	51.2561	48.8083	51.7297	**53.1588**	**51.7290**	**52.0689**	**50.0989**
dermatology	92.4648	92.4687	91.6236	93.0203	91.3498	**95.5321**	**94.1354**	**93.2981**	**94.9804**	**93.2942**
ecoli	78.2836	**79.4776**	77.6822	**77.0896**	77.6997	**78.5777**	77.1071	**80.0702**	76.8130	**78.5821**
flare	**74.4803**	**74.4803**	**74.4803**	**74.4803**	**74.4803**	74.2947	74.2947	74.2947	74.2947	74.2947
glass	68.7265	63.5991	65.4707	62.2148	**64.0421**	**71.9048**	**71.0188**	**67.2979**	**64.0310**	63.5327
iris	**93.3333**	**93.3333**	**93.3333**	**94.0000**	**92.6667**	**93.3333**	**93.3333**	**93.3333**	93.3333	92.0000
led7digit	70.6000	70.4000	70.0000	70.8000	70.8000	**71.8000**	**71.6000**	**72.0000**	**72.2000**	**72.0000**
newthyroid	91.1628	91.6279	89.7674	88.3721	**91.6279**	**94.4186**	**94.4186**	**92.5581**	**90.6977**	90.6977
nursery	**89.0446**	**89.0446**	**89.0446**	**89.0446**	**89.0446**	88.8907	88.9676	88.8138	88.7369	88.7369
page-blocks	97.0212	97.0030	96.8386	96.3269	96.4730	**97.1125**	**97.1126**	**97.1857**	**96.8384**	**97.0760**
penbased	96.1518	96.4701	96.2973	96.3609	96.2518	**97.0069**	**96.9341**	**96.6339**	**96.8158**	**96.8887**
satimage	85.5789	85.6410	84.2269	**83.6364**	82.9371	**87.0396**	**86.3869**	**84.6620**	**83.6364**	**85.0816**
segment	96.7100	96.5368	96.7100	96.1905	96.3636	**97.0130**	**96.7532**	**96.9697**	**96.7100**	**96.6234**
shuttle	99.5402	99.4943	99.5862	99.5402	98.9885	**99.7241**	**99.7241**	**99.6322**	**99.6782**	**99.1724**
splice	79.3105	78.9980	74.6230	74.2659	70.2282	**89.0179**	**85.2530**	**82.4454**	**82.7431**	**81.1756**
thyroid	**99.4861**	**99.4583**	**99.4444**	**99.3611**	**99.0556**	99.4722	99.4306	99.4167	99.2917	98.9167
vowel	79.4949	78.1818	77.0707	**78.6869**	76.8687	**79.7980**	**78.9899**	**78.7879**	78.0808	**77.5758**
wine	**94.9048**	90.9841	89.3016	92.6190	88.1746	92.1270	**92.6667**	**89.8413**	**93.7460**	**89.8889**
yeast	54.9181	55.7951	53.3697	54.0411	54.1792	**58.4239**	**58.2214**	**58.6257**	**58.2209**	**56.6043**
zoo	**94.0952**	**95.0476**	**95.0476**	**93.0952**	**93.1429**	93.0952	92.0952	92.0952	90.0952	90.1429
mean	83.7273	83.4494	82.4185	82.4612	82.1289	**85.0453**	**84.6264**	**84.0213**	**83.6369**	**83.0910**

Table 3. Test accuracy results and Wilcoxon's test p-values

Noise %	**0%**	**5%**	**10%**	**15%**	**20%**
p-value	**0.0129**	**0.0096**	**0.0017**	**0.0228**	**0.0262**

- There are several datasets with a noise level of 0%, that is, without additional noise introduced, such as *contraceptive* or *wine*, where the non-OVO version outperforms the OVO version. However, OVO perform better than non-OVO when we introduce noise into these datasets.

These results show the usefulness of decomposition strategies to deal with class noise. For this type of noise, the overall test accuracy of C4.5 using the OVO decomposition strategy is always better than that of its baseline classifier, at each level of induced noise. Also, as reflected by the Wilcoxon test p-values, a better and significant global behavior is shown when OVO is used. This clearly shows the better performance of C4.5 using this decomposition strategy in a noisy framework. This better behavior of the OVO scheme dealing with class noise can be attributed to two main causes:

- Decomposing the problem into several binary subproblems increases the separability of the classes, obtaining simpler and more regular borders between some pairs of classes, thereby facilitating the construction of the classifier. In such a way, more compact and general classifiers can be constructed.

- Collecting information from different models may provide a more robust method for classification in noisy environments than collecting information from a single model. Thus, if a noisy example does not belong to one of both classes involved in the training of a classifier, the classifier will not be affected by that noise, and its predictions will not be hindered.

6 Concluding Remarks

In this contribution we have analyzed the suitability of the OVO decomposition scheme when dealing with datasets with class noise in multi-class problems. We have created 84 datasets with class noise. We have tested the C4.5 algorithm over these datasets. This method has been compared in its original version, which directly address the multi-class problem, and considering the OVO decomposition strategy.

The test accuracy results have shown that C4.5 using OVO performs better when trained over noisy data than the baseline method. In addition, the statistical tests carried out have shown that these improvements using OVO are significant.

This better behavior of OVO with data with class noise can be attributed to two main causes: (1) decomposing the problem into several binary subproblems lead us to create simpler, easier to build, classifiers and (2) if a noisy example does not belong to one of both classes involved in the training of a classifier using OVO, the classifier will not be affected by that noise, and its predictions will not be hindered.

In future works, the consideration of other kinds and schemes of noise, e.g. the attribute noise; the incorporation of additional algorithms; or the comparison of the decomposition strategies with other preprocessing techniques to deal with noisy data can be interesting.

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

References

1. Duda, R.O., Hart, P.E., Stork, D.G.: Pattern Classification, 2nd edn. John Wiley, New York (2001)
2. Anand, A., Suganthan, P.N.: Multiclass cancer classification by support vector machines with class-wise optimized genes and probability estimates. Journal of Theoretical Biology 259(3), 533–540 (2009)
3. Hong, J.H., Min, J.K., Cho, U.K., Cho, S.B.: Fingerprint classification using one-vs-all support vector machines dynamically ordered with naïve bayes classifiers. Pattern Recognition 41(2), 662–671 (2008)

4. Wang, R.Y., Storey, V.C., Firth, C.P.: A Framework for Analysis of Data Quality Research. IEEE Transactions on Knowledge and Data Engineering 7(4), 623–640 (1995)
5. Zhu, X., Wu, X.: Class Noise vs. Attribute Noise: A Quantitative Study. Artificial Intelligence Review 22, 177–210 (2004)
6. Quinlan, J.R.: C4.5: programs for machine learning. Morgan Kaufmann Publishers, San Francisco (1993)
7. Brodley, C.E., Friedl, M.A.: Identifying Mislabeled Training Data. Journal of Artificial Intelligence Research 11, 131–167 (1999)
8. Lorena, A., de Carvalho, A., Gama, J.: A review on the combination of binary classifiers in multiclass problems. Artificial Intelligence Review 30, 19–37 (2008)
9. Knerr, S., Personnaz, L., Dreyfus, G.: Single-Layer Learning Revisited: A Stepwise Procedure for Building and Training a Neural Network. In: Fogelman Soulié, F., Hérault, J. (eds.) Neurocomputing: Algorithms, Architectures and Applications, pp. 41–50. Springer, Heidelberg (1990)
10. Anand, R., Mehrotra, K., Mohan, C.K., Ranka, S.: Efficient classification for multiclass problems using modular neural networks. IEEE Transactions on Neural Networks 6(1), 117–124 (1995)
11. Galar, M., Fernández, A., Barrenechea, E., Bustince, H., Herrera, F.: An overview of ensemble methods for binary classifiers in multi-class problems: Experimental study on one-vs-one and one-vs-all schemes. Pattern Recognition 44(8), 1761–1776 (2011)
12. Furnkranz, J.: Round Robin Classification (2002)
13. Sun, Y., Wong, A. K. C., Kamel, M. S.: Classification of Imbalanced Data: a Review. International Journal of Pattern Recognition and Artificial Intelligence, 687–719 (2009)
14. Demšar, J.: Statistical Comparisons of Classifiers over Multiple Data Sets. Journal of Machine Learning Research 7, 1–30 (2006)
15. Passerini, A., Pontil, M., Frasconi, P.: New results on error correcting output codes of kernel machines. IEEE Transactions on Neural Networks, 45–54 (2004)
16. Pimenta, E., Gama, J.: A study on error correcting output codes. In: Portuguese Conference on Artificial Intelligence EPIA, pp. 218–223 (2005)
17. Fürnkranz, J., Hüllermeier, E., Vanderlooy, S.: Binary Decomposition Methods for Multipartite Ranking. In: Buntine, W., Grobelnik, M., Mladenić, D., Shawe-Taylor, J. (eds.) ECML PKDD 2009. LNCS, vol. 5781, pp. 359–374. Springer, Heidelberg (2009)
18. Zhu, X., Wu, X., Chen, Q.: Eliminating class noise in large datasets. In: Proceeding of the Twentieth International Conference on Machine Learning, pp. 920–927 (2003)