Comparison of two topological approaches for dealing with noisy labeling

Fabien Rico, Fabrice Muhlenbach, Djamel Zighed, Stéphane Lallich

Comparison of two Topological Approaches for dealing with Noisy Labeling

Fabien Rico[a,d], Fabrice Muhlenbach[b,f], Djamel A. Zighed[a,c,e], Stéphane Lallich[a,e]

[a]*Laboratoire ERIC, EA 3083 – 5, avenue Pierre Mendès-France – 69676 Bron Cedex*
[b]*Lab. Hubert Curien, UMR CNRS 5516 – 18 rue du Pr. Benoît Lauras – 42000 St-Étienne*
[c]*Institut des Sciences de l'Homme – ISH – USR 3385 CNRS – 14 av. Berthelot, 69007 Lyon*
[d]*Université Claude Bernard, Lyon I, Université de Lyon, France*
[e]*Université Lumière, Lyon II, Université de Lyon, France*
[f]*Université Jean Monnet, Saint-Étienne, Université de Lyon, France*

Abstract

This paper focuses on the detection of likely mislabeled instances in a learning dataset. In order to detect potentially mislabeled samples, two solutions are considered which are both based on the same framework of topological graphs. The first is a statistical approach based on Cut Edges Weighted statistics (CEW) in the neighborhood graph. The second solution is a Relaxation Technique (RT) that optimizes a local criterion in the neighborhood graph. The evaluations by ROC curves show good results since almost 90% of the mislabeled instances are retrieved for a cost of less than 20% of false positive. The removal of samples detected as mislabeled by our approaches generally leads to an improvement of the performances of classical machine learning algorithms.

Keywords: Identification of mislabeled instance, relaxation, cut edges weighted, topological learning, geometrical graphs, separability index, machine learning.

1. Introduction

In machine learning, and more specifically in the framework of supervised learning, we more or less explicitly assume that the learning dataset might be noisy. Moreover, we suppose that this noise lies on the data related to the predictive variables. This noise may come from the lack of relevant predictors, from the small size of the learning sample, from the noise due to the observation/measurement tool of data acquisition, and so forth. On the other hand, we suppose that instances of the learning dataset are correctly labeled, therefore the

predicted attribute is not corrupted. This is a major assumption rarely discussed in the literature of machine learning, in comparison to the noise in predictive variables. In this paper, we will deal with the noise in the labeling.

Nowadays, specifically in the world of big data, storing, managing and retrieving information in data warehouses require real time annotation processes for indexing and labeling the continuous flow of data. These processes, which may be automatic or sometimes manual, are often imperfect and therefore generate wrong annotations and mislabeled observations. For example, manual annotation in data stream of images, video, medical curves and so on, can generate mislabeling because of human limitations, especially with a high-speed work flow. Automatic annotation processes can also produce errors in labeling objects or situations because of artifacts or because of intrinsic limitations in the automatic process/devices. For more details about situations that cause mislabeling, see [1]. Classical machine learning algorithms are not designed to deal with such noise, even though some algorithms are considered to be robust to the noise in labels. Therefore, specific pre-processing must be carried out before the learning itself. Handling mislabeled data involves at least two tasks. The first identifies samples that are likely mislabeled and the second decides what to do with them. For this latter task, two options are possible: (i) each supposed mislabeled sample is withdrawn from the learning dataset, or (ii) the true label is restored for each of them according to a specific rule. Whatever the set of tasks accomplished in order to fix the noise in the labels, at the end of the day, what we expect is an improvement, or at least no deterioration of the performances of any classifier, in which "performance" is taken to mean the accuracy of the prediction on the test sample that is not noisy. However, we can observe that the performances of a classifier might decrease after the treatment of the noise. We will propose some explanations for this later on. For now, let us focus on the process of handling the mislabeled samples in a learning dataset.

In this paper, we will focus on the detection of likely mislabeled instances in the dataset, which is the keystone of our work. What to do next? Removing, restoring or doing something else with the noisy samples that have been detected is another issue that we will discuss only briefly. We have designed two solutions for detecting potentially noisy labeling samples. Both are based on the same framework of topological graphs. The first is a statistical approach based on the Cut Edges Weighted statistics (CEW) in the neighborhood graph. The second is a relaxation technique (RT) that optimizes a local criterion in the neighborhood graph. Both solutions try to provide an estimate of the probability of the class Y for all points in the learning sample and, depending on this probability, the

samples that likely belong to a class other than to the one given are declared suspicious (likely corrupted). To compare these two methods (CEW vs. RT), we use ROC (Receiver Operating Characteristic) curves. We have also carried out some evaluations with four classifiers: KSVM, Random Forest, 1-NN and AdaBoost. The goal is to check to what extent each classifier is improved by removing the suspicious samples from the learning dataset. Various levels of noise are tried.

This article provides an update on approaches dealing with class noise that are based on topological graphs. It provides a synthesis of experiments conducted to evaluate and compare the two methods, both regarding the quality of the filtered learning set and the performance of various machine learning methods on this filtered learning set. This paper is organized in three main parts. The first introduces the concepts of topological learning using proximity graphs. It leads to a valuable tool that is a statistical test for assessing the separability of classes into a multidimensional representation space. We show that this statistic can be used to estimate the accuracy of any machine-learning algorithm. An evaluation of the relevance of this approach is carried out and discussed. In this first part, we suppose that there is no noise in the labeling. The second part deals with the noise in the labeling. It presents and compares two methods, CEW and RT using ROC curves. The third part assesses to what extent, removing likely mislabeled (suspicious) samples may improve the performance of well-known classifiers. The conclusion includes some future paths of research.

2. Definitions and Notation

Throughout this article we use the following notation and conventions.

Considering a global population Ω, the supervised learning methods aim to produce a model that predicts the unknown belonging class label $Y(i)$ of an instance i extracted from the global population Ω using its vector representation $X(i)$ associated with various real predictive attributes. The construction of the model requires a set of labeled data, called the learning set, denoted by Ω_L. We denote the size of the learning set by n, p the number of descriptive attributes, and k the number of categories of the class variable Y. The learning dataset Ω_L is a set of pairs $(X(i), Y(i)), i = 1, 2, \ldots, n$, where $Y(i)$ is the class label of i and $X(i) = (X_1(i), X_2(i), \ldots, X_p(i))$ is the p-dimensional vector corresponding to the representation of the instance i in the p-dimensional space according to the different predictive attributes. The quality of the model obtained is assessed on a test set, denoted by Ω_T, another dataset of labeled data which was not used during the learning step.

The learning ability of a given method is strongly associated with its class separability degree in $X(\Omega)$. We consider that the classes will be easier to separate, therefore to learn, if they fulfill the following conditions:

- the instances of the same class appear mostly gathered in the same subgroup in the representation space;
- the number of groups is nearly equal to the number of the classes;
- the borders between groups are "simple", with few boundary points.

2.1. Proximity Graphs and Regions of Influence

To express the proximity between examples in the representation space, we introduce the notions of *regions of influence* and *proximity graph* used by many researchers [2, 3], especially for the research of clusters in the representation space [4, 5, 6]. Proximity graphs are graphs in which points close to each other by some definition of closeness are connected.

Let Ξ be a set of points in a real space $\mathbb{R}^p$. In our case, Ξ is $X(\Omega_l)$ and p is the number of attribute. The most fundamental and natural *proximity graph* defined on a set of points Ξ is the *nearest neighbor graph* (NNG). Here, each point in Ξ is joined by an edge to its nearest neighbor [7, 8].

Another ubiquitous proximity graph, which contains the nearest neighbor graph as a subgraph, is the *minimum spanning tree* (MST) [4] (see the Figure 1(1)). Unlike the NNG, the MST and the following graphs are connected graphs. Nevertheless for machine learning problems such as instance-based learning, the most useful proximity graphs are adaptive in the sense that the number of edges they contain is a function of how the data are distributed, which is not the case of the MST. Actually the MST is a non adaptive graph, because for n points it always contains $n - 1$ edges.

Let us consider two points points α and $\beta \in \Xi$. According to Theodoridis and Koutroumbas [9], the edge between α and β is added to the graph if there is no other points $\gamma \in \Xi$ lying in the *region of influence* $R(\alpha, \beta)$. The *region of influence* [10] of two distinct points $\alpha, \beta \in \Xi$ is defined as:

$$R(\alpha, \beta) = \{\gamma : cond[d(\gamma, \alpha), d(\gamma, \beta), d(\alpha, \beta)], \alpha \neq \beta\} \quad (1)$$

where $cond[d(\gamma, \alpha), d(\gamma, \beta), d(\alpha, \beta)]$ may be defined for example as:

1. $\max[d(\gamma, \alpha), d(\gamma, \beta)] < d(\alpha, \beta)$ (***i.e.***, the relative neighborhood graph)

2. $d^2(\gamma, \alpha) + d^2(\gamma, \beta) < d^2(\alpha, \beta)$ (*i.e.*, the Gabriel graph)

The first condition defines the relative neighborhood graph (RNG) proposed by Toussaint in 1980 as a tool for extracting the shape of a planar pattern [3]. In this case, the region of influence is the lune between the vertices α and β (*e.g.*, the Figure 1(2)). The second condition defines the Gabriel graph introduced in 1969 by Gabriel and Sokal [2]. The region of influence is the sphere with the diameter defined by the points α and β (*e.g.*, the Figure 1(3)).

Proximity graphs have many applications in pattern recognition, and the most well known proximity graphs, in addition to those graphs already mentioned, are the Delaunay triangulation (DT, *e.g.*, the Figure 1(4)) and the Urquhart graph (UG) [5]. All these graphs are nested together in the following relationship:

$$NNG \subseteq MST \subseteq RNG \subseteq UG \subseteq GG \subseteq DT \tag{2}$$

In the following, we will use the Relative neighborhood Graph (RNG) of Toussaint [3, 11] defined in Equation 1, particularly because of its good properties for operation in the framework of data mining [12, 13, 6].

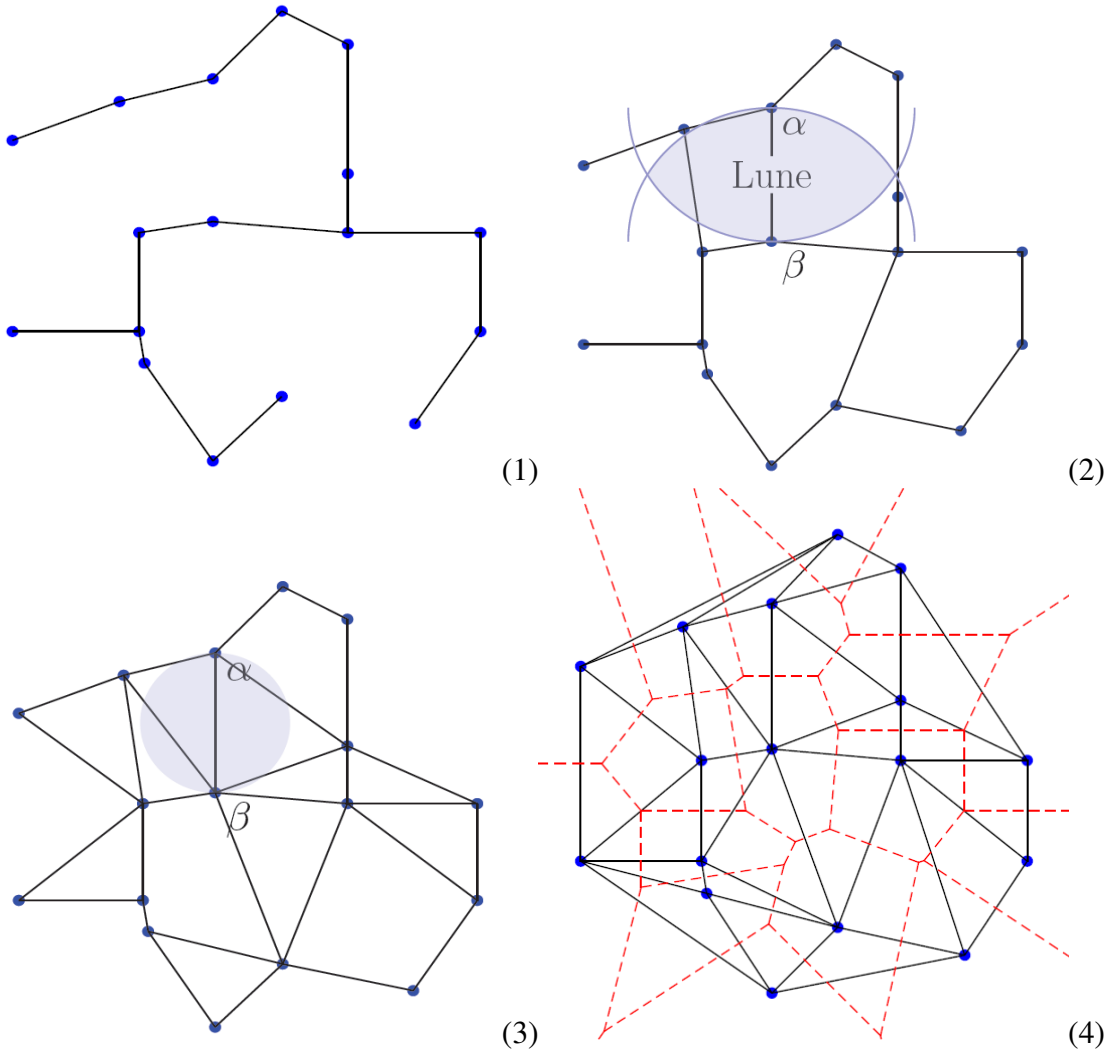
2.2. Relative neighborhood Graph and Clusters

The *relative neighborhood graph* (RNG) of Ξ is a proximity graph with vertex set Ξ, and the set of edges of the RNG of Ξ are exactly those pairs (α, β) of points for which $\max[d(\gamma, \alpha), d(\gamma, \beta)] \geq d(\alpha, \beta), \forall \gamma, \gamma \neq \alpha, \beta$, where $d(\alpha, \beta)$ denotes the distance between two points α and β in $\mathbb{R}^p$.

This definition means that the *lune* $L_{(\alpha,\beta)}$ – constituted by the intersections of hyperspheres centered on α and β with range being the edge (α, β) – is empty. For example, on the Figure 2 (a), vertices 13 and 15 are connected because there is no vertex on the *lune* $L_{(13,15)}$.

Unlike previous work done in the context of unsupervised learning [4, 5], we are interested here in the use of proximity graphs in supervised learning. Consequently, in the following, we define a *cluster* as a set of close points having the same class. The cluster is a connected sub-graph of the neighborhood graph where all vertices belong to the same class. There may be more clusters than the number of classes. To build all clusters required for characterizing the structures of the scattered data points, we proceed in two steps:

1. We generate the geometrical neighborhood graph on the learning set.
2. We remove the edges connecting two vertices belonging to different classes, so we obtain connected sub-graphs where all vertices belong to the same class.



(1) (2) (3) (4)

Figure 1: Graphs and Regions of Influence: MST (1), RNG (2), GG (3), and DT (4)

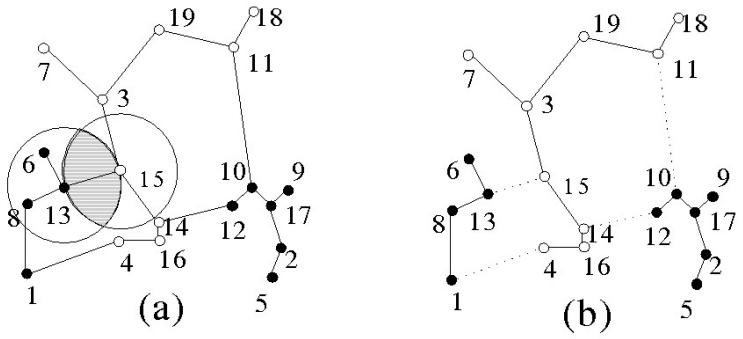


Figure 2: RNG and clusters with two classes: the black and the white points

The number of clusters generated gives partial information on the class separability. If the number of clusters is low – at least equal to the number of classes –, the classes are quite separable and we can find a learning method capable of exhibiting the model that underlies the particular group structure. For example, in Figure 2 (b), after cutting the four edges connecting vertices of different colors (in dotted line), we obtain three clusters for the two classes. But if this number tends to increase, close to the number of clusters that we might have in a random situation, the classes can no longer be learned due to the lack of a non-random geometrical structure.

Actually, this number of clusters cannot characterize a certain limited number of situations that seem intuitively different. For the same number of clusters, the situation may be very different depending on whether the clusters are easily isolated in the neighborhood graph or not. As soon as $p > 1$, rather than studying the number of clusters, we prefer to take an interest in the edges cut for building the clusters and we will calculate the relative weight (based on the distance or rank of the neighborhood between two vertices) of these edges in the edge set. In our example in Figure 2 (b), we have cut four edges for isolating three clusters.

2.3. Graph-Based Indices

We will consider the following notation:

- Number of nodes in the graph: n
- Connection matrix: $V = (v_{ij}), i = 1, 2, ..., n; j = 1, 2, ..., n$; where $v_{ij} = 1$ if i and j are linked by an edge

Table 1: Simplified Notations used for the Calculations

Notations	Definition	Case : $W = V$
S_0	$\sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} w_{ij}$	$2a$
S_1	$\frac{1}{2} \sum_{i=1}^{n} \sum_{j=1, i \neq j}^{n} (w_{ij} + w_{ji})^2$	$4a$
S_2	$\sum_{i=1}^{n} (w_{i+} + w_{+i})^2$	$4 \sum_{i=1}^{n} v_{i+}^2$

- Weight matrix: $W = (w_{ij}), i = 1, 2, ..., n; j = 1, 2, ..., n$; where w_{ij} is the weight of edge (i, j). The weight w_{ij} equals:

  1. 1 if a weight equals a simple connection ($W = V$)
  2. $(1 + d_{ij})^{-1}$ when the weight is based on the distance between the vertices i and j
  3. ${r_i}^{-1}$ for a weight based on the rank with r_j the rank of the vertex j among the neighbors of the vertex i

  Let w_{i+} and w_{+j} be the sums of row i and column j. We consider that W matrix is symmetrical (for the rank, the weights are not symmetrical, then we will use $w_{ij} = \frac{1}{2}(w_{ij} + w_{ji})$)

- Number of edges: a
- Number of classes: k
- Proportion of vertices corresponding to the class y_r: π_r, $r = 1, 2, ..., k$

According to Cliff and Ord [14], we adopt the simplified notation presented in Table 1, defining certain quantities used in the calculations. This simplification is convenient especially when the weight W corresponds to a simple connection, which means that this weight is equal to 1 if there is an edge, and 0 otherwise $(W = V)$.

3. Quality Measures in Supervised Learning

3.1. Classical Separability Indices

There are different ways for measuring the separability of the classes in a dataset. One is to use the Wilks' λ [15], another is to measure the quality of the kernel generated from this set. We can use the KTA index [16] and FSM index [17].

3.1.1. Wilks' Lambda

The usual separability indicator is Wilks' lambda, very often associated with discriminant analysis [15]. The reference method is linear discriminant analysis, which is based on the hypothesis of a Gaussian model with the same diagonal covariance matrix for all classes. To find out if classes are separable in the representation space, we test H_0, the hypothesis of equality of conditional means, using Wilks' lambda.

By denoting WGC the covariance matrix within a group and BGC the covariance matrix between groups, the definition of Wilks' lambda is :

$$\lambda = \frac{|WGC|}{|BGC|} \tag{3}$$

Lambda is between 0 and 1. It approaches 0 as the discrimination quality improves. The null hypothesis H_0 is rejected if lambda is exceptionally small considering the distribution of lambda under H_0. To calculate the p-value of lambda, we can use the Bartlett transformation, which refers back to the Chi-square distribution or the Rao transformation, which refers back to the Fisher distribution.

3.1.2. Kernel Target Alignment (KTA)

The Kernel Target Alignment (KTA) [16] tests the alignment between this kernel and a supposed ideal kernel matrix. To do that, let $C = (c_{ij})_{1 \leq i,j \leq n}$ be the matrix such that:

$$\begin{cases} c_{ij} & = & 1 \text{ if } i \text{ and } j \text{ have the same class} \\ c_{ij} & = & 0 \text{ otherwise} \end{cases}$$

The KTA measures the alignment between K, the Gaussian radial basis function, and C using the Frobenius product:

$$KTA(K) = \frac{\langle K, C \rangle_F}{\sqrt{\langle K, K \rangle_F \langle C, C \rangle_F}} \tag{4}$$

3.1.3. Feature Space-based Kernel Matrix Evaluation Measure (FSM)

The *Feature Space-based Kernel Matrix Evaluation Measure* (FSM) [17] is an index of the quality measure of a kernel matrix in a 2-class problem. It takes into account two factors: the distance between the class centers and the intra-class variance in the direction between those centers.

Let η_+ and η_- the two centers, and $e = \frac{\eta_+ - \eta_-}{|\eta_+ - \eta_-|}$ the unit vector in direction between the class centers:

$$FSM(K) = \frac{\sqrt{\frac{1}{n_+}\sum_{\eta \in C_+} \langle \eta - \eta_+, e \rangle^2} + \sqrt{\frac{1}{n_-}\sum_{\eta \in C_-} \langle \eta - \eta_-, e \rangle^2}}{|\eta_+ - \eta_-|} \quad (5)$$

3.2. Cut Edge Weight Statistic: Statistical Framework

Following our previous work [18], in a common point between supervised classification and spatial analysis, we consider a spatial contiguity graph that plays the role of the neighborhood graph [14]. The vertices of this graph are colored with k distinct colors, using the color corresponding to its modality for each vertex. The problem is (1) to describe the link between the adjacency of two vertices and the fact they have the same color, and (2) to test the hypothesis of non significance. This would require us to test the hypothesis of no spatial autocorrelation between the values taken by a categorical variable over spatial units. In the case of a neighborhood graph, this would be the results for testing the hypothesis that the class Y cannot be learned from neighborhood-based methods.

3.2.1. Definition of the Cut Edge Weight Statistic

In order to take into account a possible weighting of the edges, we deal with the matrix of symmetrized weights W, which is reduced to the connection matrix V if all the weights are equal to 1.

Edges linking two vertices of the same class (non cut edges) have to be distinguished from those linking two vertices of different classes (cut edges in order to obtain clusters).

Let us denote by I_r the sum of weights relative to edges linking two vertices of class r, and by $J_{r,s}$ the sum of weights relative to edges linking a vertex of class r and a vertex of class s.

The statistic I, corresponding to the non cut edges, is defined as:

$$I = \sum_{r=1}^{k} I_r \quad (6)$$

The statistic J corresponding to the cut edges is defined as:

$$J = \sum_{r=1}^{k-1} \sum_{s=r+1}^{k} J_{r,s} \quad (7)$$

Insofar as I and J are connected by the relation $I + J = \frac{1}{2}S_0$ (Table 1), we have only to study the J statistic or its normalization $\frac{J}{I+J} = \frac{2J}{S_0}$. Both give the

same result after standardization. We can see that I generalizes Mood's test of runs in $\mathbb{R}^p$ and k groups [19, 20].

3.2.2. Random Framework

Like Jain and Dubes [21], we consider binomial sampling in which the null hypothesis is defined by:

H_0 : the vertices of the graph are labeled independently of each other, according to the same probability distribution (π_r) where π_r denotes the probability of the class $r, r = 1, 2, ..., k$.

We could consider hypergeometric sampling by adding into the null hypothesis the constraint of having to have n_r vertices of the class $r, r = 1, 2, ...k$.

Rejecting the null hypothesis means either the classes are not independently distributed or the probability distribution of the classes is not the same for the various vertices. In order to test the null hypothesis H_0 using statistic J (or I), we first had to study the distribution of these statistics under H_0.

3.2.3. I and J Distribution under the Null Hypothesis

To test H_0 with the statistic J, we will use two-sided tests if we are surprised by abnormally small values of J (good separability of the classes) and by abnormally large values (deterministic structure or pattern presence). Hypothesis H_0 is rejected when J produces an extraordinary value taking into account its distribution under H_0. So, we have to establish the distribution of J under H_0 in order to calculate the p-value associated with the observed value of J as well as to calculate the critical value of J at the significance level α_0. This can be calculated either by simulation, by permutation or by normal approximation. In the last case, we have to calculate the mean and the variance of J under H_0. According to Cliff and Ord [14], the proof of asymptotic normality for statistic J under binomial sampling follows from a theorem of Noether [22]: J will be asymptotically normally distributed if $S_0 - 2 \times Var(J)$ is exactly of order n^{-1}.

3.2.4. Boolean Case

The two classes defined by Y are denoted 1 and 2. According to Moran [23], $U_i = 1$, if the class of the i^{th} vertex is 1 and $U_i = 0$ if the class is 2, $i = 1, 2, ..., n$. We denote π_1 the vertex proportion of class 1 and π_2 the vertex proportion of class 2. Thus:

$$J_{1,2} = \frac{1}{2} \sum_{2} w_{ij} \left(U_i - U_j\right)^2 = \frac{1}{2} \sum_{2} w_{ij} Z_{ij} \tag{8}$$

Table 2: Results related to the Statistic $J_{1,2}$

Variable	Mean	Variance
U_i	π_1	$\pi_1\pi_2$
$Z_{ij} = (U_i - U_j)^2$	$2\pi_1\pi_2$	$2\pi_1\pi_2\left(1 - 2\pi_1\pi_2\right)$
$J_{1,2}$	$S_0\pi_1\pi_2$	$S_1\pi_1^2\pi_2^2 + S_2\pi_1\pi_2\left(\frac{1}{4} - \pi_1\pi_2\right)$
$J_{1,2}$ if $w_{ij} = v_{ij}$	$2a\pi_1\pi_2$	$4a\pi_1^2\pi_2^2 + \pi_1\pi_2\left(1 - 4\pi_1\pi_2\right)\sum_{i=1}^{n} v_{i+}^2$

where U_i are independently distributed according to Bernoulli's distribution of parameter π_1, denoted $B(1, \pi_1)$. It should be noted that the variables $Z_{ij} = (U_i - U_j)^2$ are distributed according to the distribution $B(1,\, 2\pi_1\pi_2)$, but are not independent. Actually, the covariances $Cov(Z_{ij}, Z_{kl})$ are null only if the four indices are different. Otherwise, when there is a common index, one can obtain:

$$Cov(Z_{ij}, Z_{il}) = \pi_1\pi_2(1 - 4\pi_1\pi_2) \tag{9}$$

Table 2 summarizes the various results related to the statistic $J_{1,2}$.

The p-value of $J_{1,2}$ is calculated from standard normal distribution after centering and reducing its observed value. The critical values for $J_{1,2}$ at the significance level α_0 are:

$$J_{1,2;\alpha_{0/2}} = S_0\pi_1\pi_2 - u_{1-\alpha_{0/2}}\sqrt{S_1\pi_1^2\pi_2^2 + S_2\pi_1\pi_2\left(\frac{1}{4} - \pi_1\pi_2\right)} \tag{10}$$

$$J_{1,2;1-\alpha_{0/2}} = S_0\pi_1\pi_2 + u_{1-\alpha_{0/2}}\sqrt{S_1\pi_1^2\pi_2^2 + S_2\pi_1\pi_2\left(\frac{1}{4} - \pi_1\pi_2\right)} \tag{11}$$

3.2.5. Multiclass Case

To extend these results to the multi-class case, according to Cliff and Ord [14], we reason with the I and J statistics already defined. These statistics are I and J defined on formula 12 and formula 13:

$$I = \sum_{r=1}^{k} I_r = \frac{1}{2}\sum_{2} w_{ij}T_{ij} \tag{12}$$

$$J = \sum_{r=1}^{k-1}\sum_{s=r+1}^{k} J_{r,s} = \frac{1}{2}\sum_{2} w_{ij}Z_{ij} \tag{13}$$

Table 3: Test Statistics and their Means

Test statistic	Mean
$I = \sum_{r=1}^{k} I_r$	$\frac{1}{2} S_0 \sum_{r=1}^{k} \pi_r^2$
$J = \sum_{r=1}^{k-1} \sum_{s=r+1}^{k} J_{r,s}$	$S_0 \sum_{r=1}^{k-1} \sum_{s=r+1}^{k} \pi_r \pi_s$

where T_{ij} and Z_{ij} are random boolean variables that indicate if the vertices i and j are of the same class (T_{ij}) or not (Z_{ij}). From previous results, we easily obtain the mean of I and J, as presented in Table 3.

Because I and J are connected by the relation $I+J = \frac{1}{2} S_0$, these two variables have the same variance, denoted $\sigma^2 = Var(I) = Var(J)$. The calculation of σ^2 is complicated due to the need to consider the covariances. In accordance with Cliff and Ord [14], we obtain the following results for binomial sampling:

$$4\sigma^2 = S_2 \sum_{r=1}^{k-1} \sum_{s=r+1}^{k} \pi_r \pi_s \tag{14}$$

$$+ \quad (2S_1 - 5S_2) \sum_{r=1}^{k-2} \sum_{s=r+1}^{k-1} \sum_{t=s+1}^{k} \pi_r \pi_s \pi_t \tag{15}$$

$$+ \quad 4\,(S_1 - S_2) \left[\sum_{r=1}^{k-1} \sum_{s=r+1}^{k} \pi_r^2 \pi_s^2 - 2 \sum_{r=1}^{k-3} \sum_{s=r+1}^{k-2} \sum_{t=s+1}^{k-1} \sum_{u=t+1}^{k} \pi_r \pi_s \pi_t \pi_u \right] \tag{16}$$

3.2.6. Complexity of the Test

Different steps are taken into consideration: computing the matrix distance is in $O(p \times n^2)$, with n the number of examples and p the attributes, and building the neighborhood graph in $\mathbb{R}^p$ is in $O(n^3)$. Because the number of attributes p is very small compared to the number of instances n, the test is in $O(n^3)$.

Several solutions are possible to apply CEW and RT methods to big data. The first one, especially useful for online learning of data flows is to use incremental construction procedures of topological graphs [24]. The second solution is to adapt sampling graph methods [25] to our problem and to work with only a sample of the training data. A sample, particularly a stratified sample, is enough to provide a good idea of the class separability in the database. We intend to develop these approaches in our future work.

4. Tests on Benchmark Datasets

4.1. Benchmark Description

The Cut Edge Weight (CEW) statistic and other classical separability indices were studied experimentally on 22 benchmarks (see Table 4) from the UCI Machine Learning Repository [26]: *Arcene*, *Bupa* (Liver Disorders, BUPA), *Ecoli*, *Glass Identification*, *Image Segmentation*, *Ionosphere*, *Iris*, *Letter Recognition* (subset R vs. B), *Musk* (Version 1), *Parkinsons*, *Pima Indians Diabetes*, *Planning Relax* (EEG records during two mental states: planing/relax), *Ringnorm* Benchmark Problem, Connectionist Bench (*Sonar*, Mines vs. Rocks), *Spam* base, *Threenorm* Benchmark Problem, Blood *Transfusion* Service Center, *Twonorm* Benchmark Problem, *Waveform* Database Generator (Version 1), *Wine*, *Breast Cancer* Wisconsin (Original), and *Yeast*. Note that the datasets "Ringnorm Benchmark Problem", "Three norm Benchmark Problem", and "Two norm Benchmark Problem" are generated by the R functions mlbench.ringnorm, mlbench.threenorm, and mlbench.twonorm respectively from the *mlbench* R package [27, 28, 26]. These databases were chosen because they have only digital attributes and a symbolic class. Because the geometrical neighborhood is sensitive to the scale measure, we standardized all the attributes of the domains.

4.2. Test Values on a Benchmark Set

The normalized values obtained on the Cut Edge Weight (CEW) statistic were compared to error rates obtained by different supervised machine learning algorithms on all these benchmarks:

1. AdaBoost [29]
2. C4.5 [30, 31]
3. 1-NN (KNN with K=1) [32, 33]
4. SVM [34, 35]
5. Random Forest [36, 37]
6. Logistic Model Trees [38, 31]
7. LogitBoost [39, 31]
8. Mean error of the classifiers [40]
9. Majority Committee of the classifiers [40]

The results are presented in Figures 3 to 5. Each point in these figures represents a benchmark using the value of a classification index (as Y-axis) and machine learning algorithm (X-axis). First, Figure 3 and Tables 5 and 6 present the error

Table 4: Dataset Description: numbers of instances, of attributes and classes, mean of the neighbours obtained for the Relative neighborhood Graph (RNG) and for the Gabriel Graph (GG)

Name (UCI)	# inst.	# attr.	# classes	RNG neighb.	GG neighb.
Arcene	200	1017	2	3	174
Bupa	345	5	2	3	12
Ecoli	336	7	8	3	10
Glass	214	9	6	3	10
Image Segm.	2309	18	7	3	17
Ionosphere	351	33	2	2	23
Iris	150	4	3	2	6
Letter R vs. B	1524	16	2	4	49
Musk 1	476	166	2	3	121
Parkinsons	195	22	2	3	18
Pima	768	8	2	4	30
Planning Relax	182	12	2	3	10
Ringnorm	2128	20	2	4	359
Sonar	208	60	2	3	18
Spam base	4601	57	2	5	105
Threenorm	2128	20	2	6	561
Transfusion	748	4	2	13	17
Twonorm	2128	20	2	6	530
Waveform	5000	21	3	6	581
Wine	178	13	3	3	27
Wisc. Breast Cancer	681	9	2	20	56
Yeast	1484	8	10	4	23

rates obtained by these methods on a 10-cross validation with the benchmarks and the CEW statistical values previously calculated without weighting, which means by using only the simple connection between two vertices of the computed graph: 1 if there is an edge between these vertices, otherwise 0. Second, the Figure 4 presents the results obtained by the supervised learning methods while changing the method of computing the weights of the CEW statistic (without weighting = simple connection, by using the distance or by using the rank). Third, Figure 5 presents the results obtained with other indices known for giving an idea of the separability of the classes on a dataset presented in the previous section.

In Figure 3, the error rates for the different learning methods, and particulary the mean of these methods, are well correlated with the relative cut edge weight $(J/(I+J))$. Except for AdaBoost, there is a linear relation between the classification methods and the values obtained for the CEW statistic. In the Table 5 and Table 6, the values of the coefficient of determination (the adjusted R^2) are very close to 1, which confirms the linear relationship between the error rate obtained by standard supervised machine learning methods and the CEW statistic. This linear relationship between another standard class separability index (FSM, KTA or Wilks' Lambda) and the error rate obtained by the classification methods cannot be found as shown in the Figure 5. Moreover, as shown in Figure 4, there is no significant difference between the different ways of computing the weights of the CEW statistic. In the following, we will then use the simplest way for computing the weight of the CEW statistic: the simple connection.

We can conclude from all these results that the CEW statistic is a good indicator of the separability of classes. Moreover, It provides us with a way to estimate the accuracy rate of any classifier. Hence, knowing the linear relation between CEW and any classifier among those we have tested, we may identify which one will be the best for a given dataset.

5. Noisy Data – Experimental Validation

Keeping the framework of neighborhood graphs and CEW indicator in mind, we will now introduce two methods for identifying suspicious samples, *i.e.* samples that may have been mislabeled. Contrary to the previous section, where the noise was assumed to lie in the predictive attributes, and not on the predicted one, here we assume the inverse. We will present two methods. One is based on the CEW criterion and the other on a relaxation technique. Both will be evaluated according to the same protocol and the same dataset. First, we describe the evaluation process.

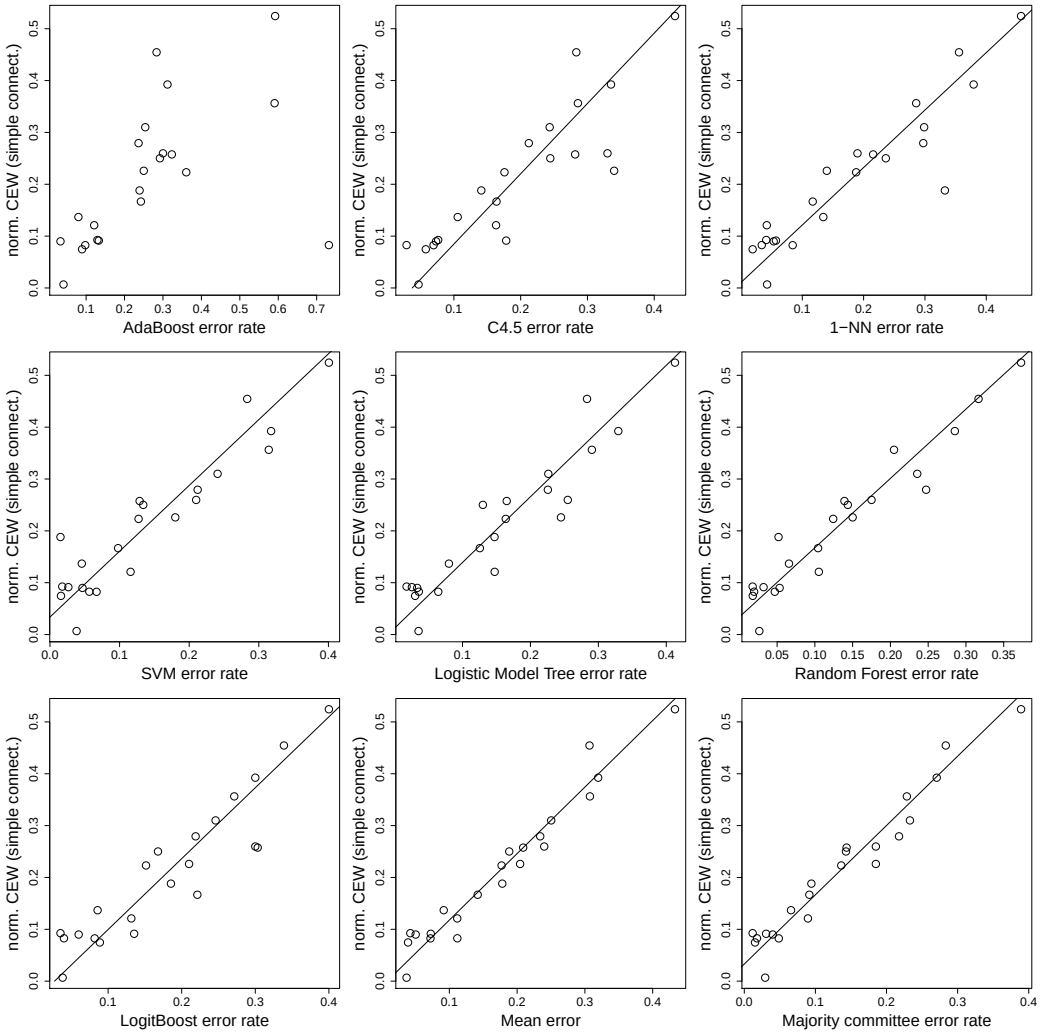


Figure 3: CEW Values and Classification Method Error rate on a Benchmark Set

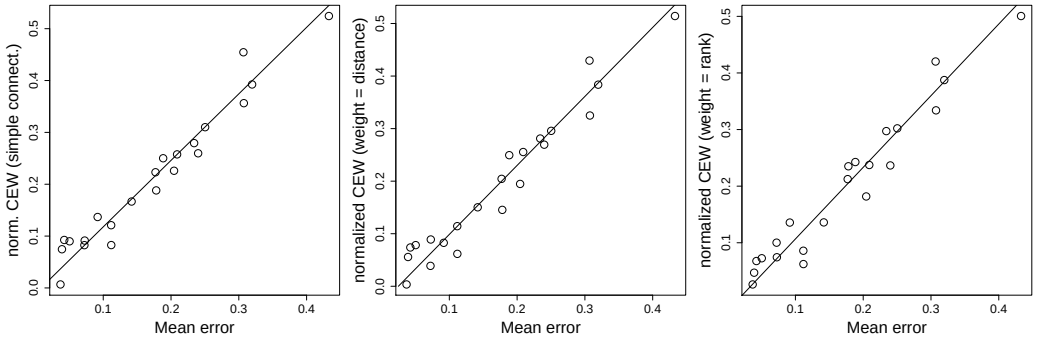


Figure 4: CEW Values and Weights: Simple Connection, Distance or Rank

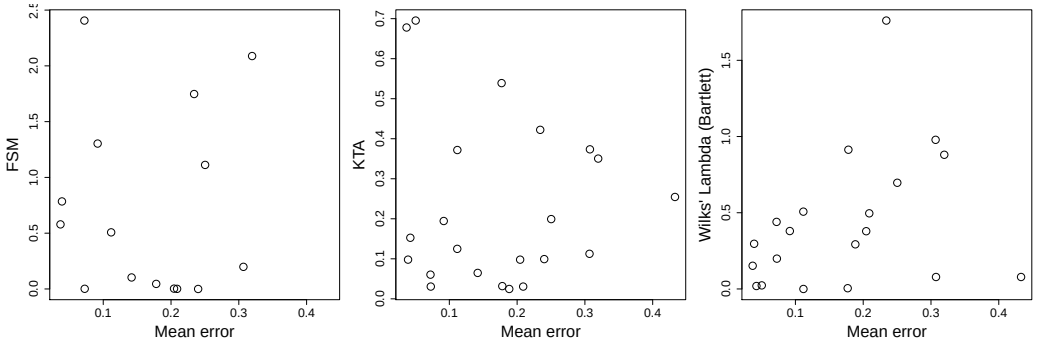


Figure 5: Other Classification Indices and Mean of the Error Rate obtained by the Classification Methods: Feature Space-based Kernel Matrix Evaluation Measure (FSM), Kernel Target Alignment (KTA) and Wilks' Lambda

Table 5: Mean Error Rates obtained by Cross-Validation on the Datasets (1/2)

Dataset	RF	KSVM	1-NN	3-NN	C4.5	LMT	adaBoost	LogitBoost	Majority
Arcene	0.175	0.210	0.190	0.215	0.330	0.255	0.300	0.300	0.185
Bupa	0.285	0.318	0.379	0.347	0.335	0.329	0.312	0.300	0.271
Ecoli	0.124	0.127	0.188	0.167	0.176	0.164	0.361	0.152	0.136
Glass	0.205	0.314	0.286	0.295	0.286	0.290	0.590	0.271	0.229
Image segm.	0.020	0.057	0.034	0.042	0.029	0.036	0.732	0.040	0.018
Ionosphere	0.066	0.046	0.134	0.163	0.106	0.080	0.080	0.086	0.066
Iris	0.053	0.047	0.053	0.060	0.073	0.033	0.033	0.060	0.040
Letter RvsB	0.018	0.016	0.019	0.018	0.058	0.030	0.089	0.089	0.015
Musk1	0.104	0.098	0.117	0.113	0.164	0.126	0.243	0.221	0.091
Parkinsons	0.105	0.116	0.042	0.089	0.163	0.147	0.121	0.132	0.089
Pima	0.236	0.241	0.299	0.275	0.243	0.226	0.254	0.246	0.233
Planning relax	0.317	0.283	0.356	0.333	0.283	0.283	0.283	0.339	0.283
Ringnorm	0.052	0.015	0.333	0.397	0.141	0.147	0.239	0.185	0.094
Sonar	0.150	0.180	0.140	0.140	0.340	0.245	0.250	0.210	0.185
Spam base	0.047	0.067	0.084	0.090	0.070	0.064	0.098	0.082	0.048
Threenorm	0.139	0.129	0.216	0.180	0.282	0.165	0.323	0.303	0.144
Transfusion	0.247	0.212	0.297	0.239	0.212	0.226	0.236	0.219	0.218
Twonorms	0.032	0.026	0.057	0.035	0.178	0.025	0.133	0.135	0.031
Waveform	0.144	0.134	0.236	0.205	0.244	0.130	0.292	0.168	0.143
Wine	0.018	0.018	0.041	0.035	0.076	0.018	0.129	0.035	0.012
Wisc. Breast Cancer	0.03	0.04	0.04	0.03	0.05	0.04	0.04	0.04	0.03
Yeast	0.37	0.4	0.46	0.44	0.43	0.41	0.59	0.4	0.39

Table 6: Mean Error Rates obtained by Cross-Validation on the Datasets (2/2)

Dataset	CEW s. connex.	CEW dist.	CEW rank	Wilks' λ	KTA	FSM
Arcene	0.260	0.269	0.237	–	0.099	0.000
Bupa	0.392	0.384	0.387	0.880	0.350	2.088
Ecoli	0.223	0.204	0.212	0.005	0.539	–
Glass	0.356	0.325	0.334	0.079	0.374	–
Image segm.	0.083	0.062	0.062	0.000	0.372	–
Ionosphere	0.137	0.083	0.136	0.380	0.194	1.304
Iris	0.090	0.078	0.073	0.023	0.695	–
Letter RvsB	0.075	0.056	0.047	0.296	0.098	0.785
Musk1	0.167	0.150	0.136	–	0.065	0.103
Parkinsons	0.121	0.114	0.086	0.507	0.125	0.508
Pima	0.310	0.296	0.302	0.697	0.199	1.112
Planning relax	0.455	0.429	0.420	0.979	0.113	0.198
Ringnorm	0.188	0.145	0.235	0.914	0.032	0.046
Sonar	0.226	0.195	0.182	0.379	0.098	0.003
Spam base	0.082	0.039	0.100	0.440	0.060	2.406
Threenorm	0.258	0.255	0.237	0.496	0.031	0.001
Transfusion	0.279	0.281	0.297	1.761	0.422	1.748
Twonorms	0.091	0.089	0.074	0.199	0.031	0.002
Waveform	0.250	0.249	0.243	0.293	0.025	–
Wine	0.093	0.074	0.068	0.019	0.153	–
Wisc. Breast Cancer	0.0067	0.0034	0.026	0.15	0.68	0.58
Yeast	0.52	0.51	0.5	0.079	0.25	–

Table 7: Characteristics of the Linear Relationship between the CEW Statistic and the Error Rate obtained by Classification Methods: Coefficient, Constant and Adjusted R^2 (Measure of Goodness-of-Fit of Linear Regression)

	coefficient	constant	adjusted R^2
Random Forest	0.748	-0.0253	0.921
SVM	0.788	-0.0266	0.865
1-NN	0.902	-0.00953	0.848
C4.5	0.737	0.0376	0.765
Logist. Mod. Tree	0.789	-0.00971	0.882
AdaBoost	0.742	0.103	0.267
LogitBoost	0.731	0.0272	0.845
Majority committee	0.748	-0.0247	0.945
Mean	0.780	0.00823	0.950

5.1. CEW Test: Identification of Suspect Examples

As shown in [41], an example label is subject to caution when its class is different from that of the examples belonging to its geometrical neighborhood, even more so as the classes are easily discernible in the representation space. In order to identify such an example, we propose calculating the sum of the cut edges weights that run from this example (***i.e.***, edges which link this example to other examples in its geometrical neighborhood that do not have the same label).

Let i be an example whose class is $y_{r(i)}$. We denote by $\pi_{r(i)}$ the global proportion of the class $y_{r(i)}$ in the learning set. The labeling of an instance i is considered as a *non suspicious* if, in its geometrical neighborhood, the proportion of examples that do not have the same label $y_{r(i)}$ is significantly smaller than $1 - \pi_{r(i)}$.

We denote the matching null hypothesis as H_0. Thus, for a *non suspicious* example the weight of cut edges is significantly smaller than its expected value under H_0.

Under H_0, in the neighborhood of i, the probability of an instance being different from the class of i is $1 - \pi_{r(i)}$. We note as n_i the number of examples belonging to the neighborhood of i, w_{ij} is the weight of the edge connecting the vertices i and j, and J_i is the absolute weight of cut edges running from i.

$$J_i = \sum_{j=1}^{n_i} w_{ij} I_i(j) \tag{17}$$

where $I_i(j)$ are independent and identically distributed random variables, according to Bernouilli's law of parameters $1 - \pi_{r(i)}$, under H_0. The mean E and the variance Var of J_i under H_0 are given by:

$$E(J_i/H_0) = \left(1 - \pi_{r(i)}\right) \sum_{j=1}^{n_i} w_{ij} \tag{18}$$

$$Var(J_i/H_0) = \pi_{r(i)} \left(1 - \pi_{r(i)}\right) \sum_{j=1}^{n_i} w_{ij}^2 \tag{19}$$

We propose ranking all examples i, $i = 1, 2, ..., n$, according to the level of the normal distribution function to which the realization of J_i corresponds under H_0 (left unilateral p-value). In so doing, we define ideally three categories of examples: *likely uncorrupted* examples located in the left rejection region (significantly less cut edges than expected under H_0, and consequently more neighbors

of the same label), *doubtful* examples that do not contradict H_0 and *likely corrupted* examples located in the right rejection region (significantly more cut edges than expected under H_0, and consequently fewer neighbors with the same label).

If n_i is large enough and the weights are not too unbalanced, we can use the normal approximation of J_i^s (J_i standardized). Otherwise, we can proceed by simulation in order to calculate the associated p-values to each J_i. In order to characterize those three types of examples efficiently, we chose to control two parameters that are fixed by the user:

- θ_1, the left risk, poses the boundary between *likely uncorrupted* and *doubtful* examples;
- θ_2, the complement of the right risk, poses the boundary between *doubtful* and *likely corrupted* examples.

An example will be considered as *likely uncorrupted*, *doubtful* or *likely corrupted*, depending on J_i^s left unilateral p-value being smaller than θ_1, between θ_1 and θ_2 or greater than θ_2. By varying θ_1 and θ_2, we can modulate the definition of each type of example. The closer θ_1 is to 0, the more severe the definition of *likely uncorrupted* examples. The closer θ_2 is to 1, the more severe for the *likely corrupted* examples. The closer θ_2 is to θ_1, the less room there is for *doubtful* examples.

5.2. Relaxation Principle

The second method we propose for detecting samples that belong to a suspicious class considers a criterion termed label consistency [42]. The underlying assumption of classification is that similar individuals in the representation space should have have the same labels.

Suppose the predicted attribute Y has k labels. First we can define a profile for each instance i by $l_i = l_{i1}, \ldots, l_{ik}$ where l_{ij} is the proportion of examples labeled y_j in the neighborhood of i. Then

$$L = \begin{pmatrix} l_1 \\ l_2 \\ \vdots \\ l_n \end{pmatrix}$$

be the $n \times k$ matrix containing all the profile for each instance of the learning dataset. We need to change the profile to a new matrix L' in order to improve the

local consistency. The consistency is a combination of two criteria, termed local similitude and global similitude defined as follows:

Let $\mathcal{N}_i$ be the neighborhood of each individual i, in [42], Zighed et al. try to minimize a combination of 2 functions:

- local similitude:

$$F_{loc} = \sum_{i \in \Omega} \frac{1}{\#\left(\mathcal{N}_i\right)} \sum_{j \in \mathcal{N}_i} \left| l'_i - l'_j \right|^2 \tag{20}$$

- global similitude:

$$F_{glob} = \sum_{i \in \Omega} \left| l'_i - l_i \right|^2 \tag{21}$$

Let $p_i = \frac{1}{\#(\mathcal{N}_i)}$ then we need to minimize

$$\begin{aligned}
F(L) &= \alpha F_{loc} + (1-\alpha) F_{glob} && (22)\\
&= \alpha \sum_{i \in \Omega} p_i \sum_{j \in \mathcal{N}_i} \left| l'_i - l'_j \right|^2 + (1-\alpha) \sum_{i \in \Omega} \left| l'_i - l_i \right|^2 && (23)\\
&= \alpha \sum_{i \in \Omega} p_i \sum_{j \in \mathcal{N}_i} \sum_{l=1}^{k} (l'_{il} - l'_{jl})^2 + (1-\alpha) \sum_{i \in \Omega} \sum_{l=1}^{k} (l'_{il} - l_{il})^2 && (24)\\
&= \sum_{l=1}^{k} \underbrace{\left[\alpha \sum_{i \in \Omega} p_i \sum_{j \in \mathcal{N}_i} (l'_{il} - l'_{jl})^2 + (1-\alpha) \sum_{i \in \Omega} (l'_{il} - l_{il})^2 \right]}_{F_l} && (25)
\end{aligned}$$

Each term F_l may be minimized separately, which means that $\forall l \in \{1, \ldots, k\}$ we have to find the vector $l'^l = {}^t(l'_{1l}, \ldots, l'_{nl})$ minimizing F_l. Since F_l is a sum of convex functions, it is convex and admits a general minima for which the gradient is null. This implies that $\forall i \in \Omega$, $\frac{\partial F_l}{\partial l'_{ik}} = 0$ and, consequently, l'^l is the solution of the linear equation:

$\forall i \in \{1, \ldots, n\}$,

$$\alpha l'_{il} \sum_{j \in \mathcal{N}_i} (p_i + p_j) - \alpha \sum_{j \in \mathcal{N}_i} l'_{jl} (p_i + p_j) + (1-\alpha)(l'_{il} - l_{il}) = 0 \tag{26}$$

Note that this equation is possible thanks to the symmetry of the neighborhood graphs as $i \in \mathcal{N}_j \Leftrightarrow j \in \mathcal{N}_i$.

The optimum profile of each individual is to be compared to the original profile, a vector of classes where only one component is 1, which corresponds to the initial belonging class, while the other components are null since at the beginning each sample belongs to only one class.

5.3. Evaluation

Here we wish to assess the ability of both methods to find instances with a noisy label (instances with corrupted label). For each instance, our methods provide a way to compute a suspicion index, but the chance to detect the corrupted instances depends on the chosen decision threshold. We carried out several comparisons on different databases:

- First we carried out a comparison based on the ROC curve. We chose the ROC curve because it provides an overall assessment of performance and integrates the various trade-offs between benefits (true positives) and costs (false positives). In addition, the area under the ROC curve is insensitive to class imbalance.

- For information purposes, we compute recall, precision and F-measure for some remarkable decision thresholds (0.25, 0.50, 0.75 for both CEW and RT). The corresponding values are reported in Table 8 for the Cut Edge Weight (CEW) and Table 9 for the Relaxation Technique (RT).

Table 8: Precision and Recall for different Values of the Threshold using CEW

Database	Noise	Threshold=0.25			Threshold=0.5			Threshold=0.75		
	Rate	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Bupa	0.1	0.54	0.14	0.22	0.77	0.12	0.21	0.88	0.10	0.19
Bupa	0.25	0.55	0.31	0.40	0.76	0.27	0.40	0.92	0.26	0.41
Bupa	0.40	0.48	0.43	0.46	0.71	0.41	0.52	0.89	0.41	0.56
Iris	0.10	0.85	0.67	0.75	0.93	0.37	0.53	0.99	0.23	0.37
Iris	0.25	0.77	0.69	0.73	0.94	0.51	0.66	0.98	0.38	0.55
Iris	0.40	0.65	0.68	0.66	0.86	0.56	0.68	0.96	0.48	0.64
Spam base	0.1	0.85	0.39	0.54	0.94	0.24	0.38	0.98	0.16	0.27
Spam base	0.25	0.72	0.49	0.58	0.87	0.38	0.53	0.96	0.31	0.47
Spam base	0.4	0.55	0.51	0.53	0.78	0.45	0.57	0.94	0.43	0.59
Twonorms	0.1	0.91	0.64	0.75	0.96	0.41	0.58	0.98	0.25	0.40
Twonorms	0.25	0.78	0.66	0.72	0.92	0.50	0.65	0.97	0.37	0.54
Twonorms	0.4	0.56	0.59	0.58	0.79	0.51	0.62	0.93	0.45	0.61
Waveform	0.1	0.77	0.52	0.62	0.88	0.35	0.50	0.94	0.24	0.38
Waveform	0.25	0.70	0.67	0.68	0.86	0.52	0.65	0.94	0.41	0.57
Waveform	0.4	0.60	0.69	0.64	0.82	0.58	0.68	0.93	0.51	0.66
Wisc. B. C.	0.1	0.94	0.70	0.80	0.98	0.47	0.64	0.99	0.32	0.49
Wisc. B. C.	0.25	0.88	0.74	0.80	0.93	0.58	0.72	0.97	0.47	0.64
Wisc. B. C.	0.4	0.63	0.67	0.65	0.83	0.57	0.68	0.94	0.50	0.65

Six classical datasets were used for this. The labels were corrupted at the rate of 10% with respect to the initial distribution of the classes. The Figure 6 presents the ROC (Receiver Operating Characteristics) curves of both methods

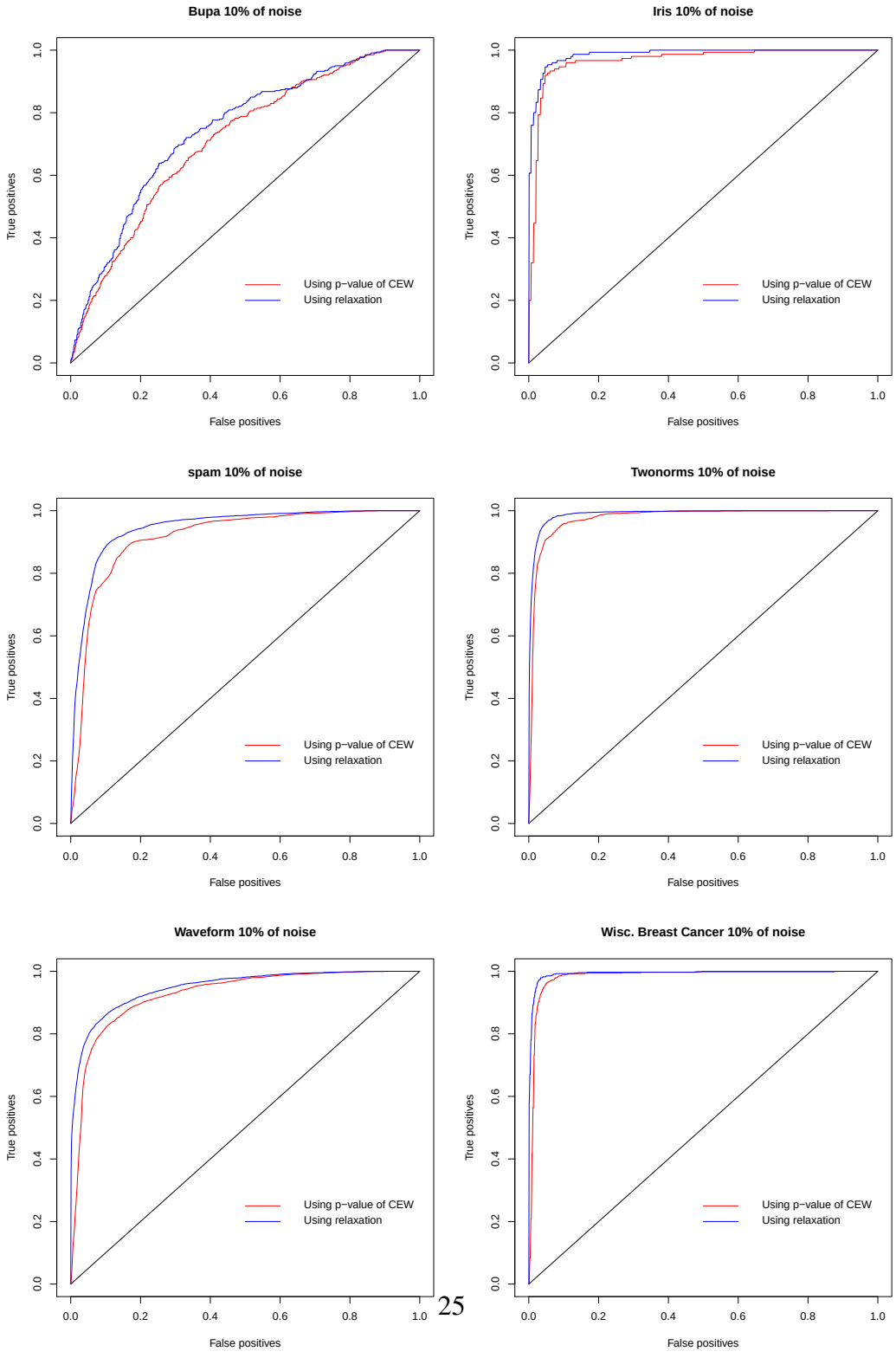


Figure 6: Receiver Operating Characteristic (ROC) for 6 Datasets with 10% of Noise

Table 9: Precision and Recall for different Values of the Threshold using Relaxation

Database	Noise	Threshold=0.25			Threshold=0.5			Threshold=0.75		
	Rate	Recall	Precision	F-measure	Recall	Precision	F-measure	Recall	Precision	F-measure
Bupa	0.1	0.11	0.22	0.14	0.50	0.18	0.27	0.85	0.11	0.20
Bupa	0.25	0.08	0.38	0.14	0.45	0.36	0.40	0.87	0.27	0.41
Bupa	0.4	0.08	0.52	0.14	0.38	0.45	0.41	0.85	0.41	0.55
Iris	0.1	0.68	0.86	0.76	0.91	0.64	0.75	0.99	0.30	0.46
Iris	0.25	0.58	0.88	0.70	0.92	0.70	0.80	1.00	0.41	0.58
Iris	0.4	0.41	0.80	0.54	0.84	0.69	0.76	0.98	0.49	0.65
Spam base	0.1	0.51	0.71	0.59	0.85	0.53	0.65	0.96	0.29	0.44
Spam base	0.25	0.28	0.78	0.42	0.71	0.65	0.68	0.93	0.39	0.55
Spam base	0.4	0.14	0.65	0.23	0.51	0.59	0.55	0.88	0.45	0.60
Twonorms	0.1	0.60	0.96	0.74	0.94	0.72	0.82	0.99	0.30	0.46
Twonorms	0.25	0.19	0.96	0.32	0.88	0.81	0.84	0.99	0.34	0.51
Twonorms	0.4	0.08	0.81	0.14	0.65	0.72	0.68	0.97	0.42	0.59
Waveform	0.1	0.67	0.77	0.72	0.91	0.36	0.51	0.99	0.15	0.26
Waveform	0.25	0.56	0.87	0.68	0.92	0.53	0.67	0.99	0.28	0.44
Waveform	0.4	0.38	0.84	0.52	0.93	0.57	0.71	1.00	0.41	0.58
Wisc. B. C.	0.1	0.79	0.92	0.85	0.95	0.80	0.87	0.99	0.50	0.66
Wisc. B. C.	0.25	0.36	0.95	0.52	0.90	0.87	0.88	0.98	0.42	0.59
Wisc. B. C.	0.4	0.06	0.80	0.12	0.74	0.81	0.77	0.96	0.43	0.59

applied, respectively, with the six datasets. As we can see, results are very good since we retrieve around 90% of the corrupted labels for only around 20% of false detection. Both methods behave similarly and their results are very close even though the method based on relaxation slightly outperforms the other one. The results are stable even though we vary the percentage of corrupted labels up to 30%.

6. Prediction from Filtered Training Set

In the literature, many authors have already shown that some machine learning algorithms are resistant to the label noise in the training set. We have also observed these results on many datasets.

6.1. Removing Suspicious Instances

We have aimed to assess to what extend the removal of suspicious instances might improve the accuracy of a given machine learning method. We selected four machine learning algorithms (MLA): SVM, Random Forest, 1-NN and AdaBoost. Four datasets (D) from machine learning repository were used. For each (MLA,D) pair, we carried out tree experiments with corrupted labels of the learning dataset. The noise was not processed in the first experiment, the second experiment removed those instances detected as mislabeled by CEW and the third removed the instances detected as mislabeled by the relaxation method. The labels of the learning datasets were corrupted at several levels up to 40% in respect to the original distribution of the classes. At each noise level up to 40%, a 10-cross validation was performed. The labels of the 10 test samples of the cross validation remained at the original values without alteration, contrary to the 10 learning

datasets generated. On the "Spam" Dataset, the Figure 7 shows the results of the four classifiers: AdaBoost (top-left), 1-NN (top-right), SVM (bottom, left) and Random Forest (bottom, right) are shown. On this dataset, removing instances deemed corrupted by our methods does not significantly improve either AdaBoost nor SVM as the tree curves are almost superposed. These two MLAs seem not to be sensitive to the noise of the labeling on this case. Nevertheless, removing instances deemed mislabeled improves the accuracy of Random-Forest and 1-NN in cross validation. We observe a slight advantage when the detection is done by the CEW method. For the "two norms" dataset, the Figure 8, there are slight improvements for all MLAs specially for 1-NN. For the "waveform" dataset, a significant improvement for AdaBoost and 1-NN whereas KSM and Random Forest behave similarly and remain almost insensitive to the noise.

6.2. Filter or Relabeling

Since we detected suspicious points, it is possible to relabel them instead of removing them. For the relaxation algorithm, this is quite natural as the obtained profile gives a label which maximizes the similitude. But for the CEW filtering, this leads to several issues:

- Which relabeling algorithm should be used?
- To relabel a given point, should we consider the other suspicious point or not?

In preliminary test, we tried several solutions to these problems. During these tests, the relabeling technique did not give any better results than the filtering technique. For example, for "Waveform" dataset and Random Forest, relabeling using CEW statistic gives results similar to filtering: slightly worse for small perturbation (15.5% err. rate vs 14.9% error rate) and slightly better for bigger ones (20% vs 22%). In other examples, the relabeling technique is always the worst! Using waveform, with 1-NN algorithm, the error rate of filtering using CEW is always smaller (by 10%) than the one using relabeling. To our mind, these poor results are due to the fact that relabeling may add errors. As the filtering technique is not perfect, the relabeled points may be good ones. In that case, filtering may be considered as a resumption of the learning set (it suppresses the border points) while relabeling may just add errors. This is especially the case when the wrong points are quite rare and this explains that relabeling is generally worse for small perturbations.

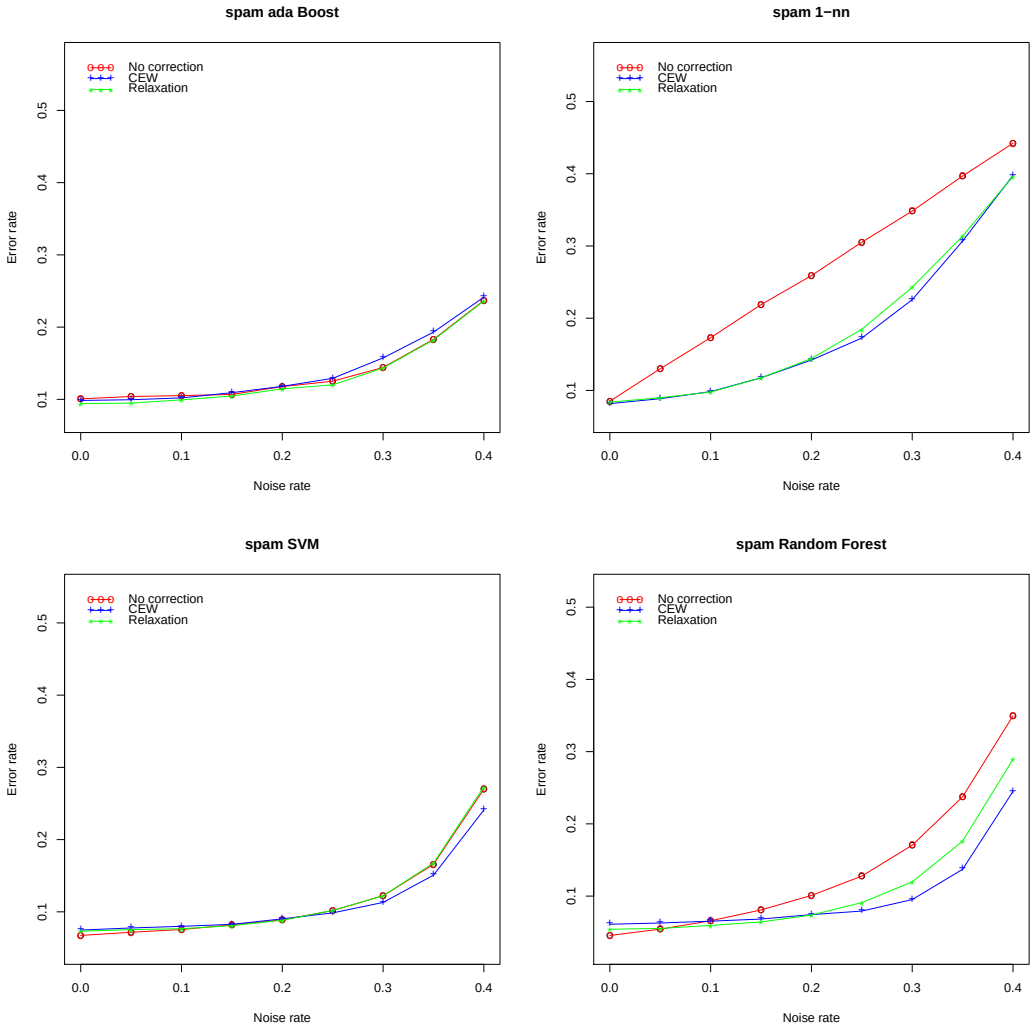


Figure 7: Filtering Improvement on *Spam* Dataset

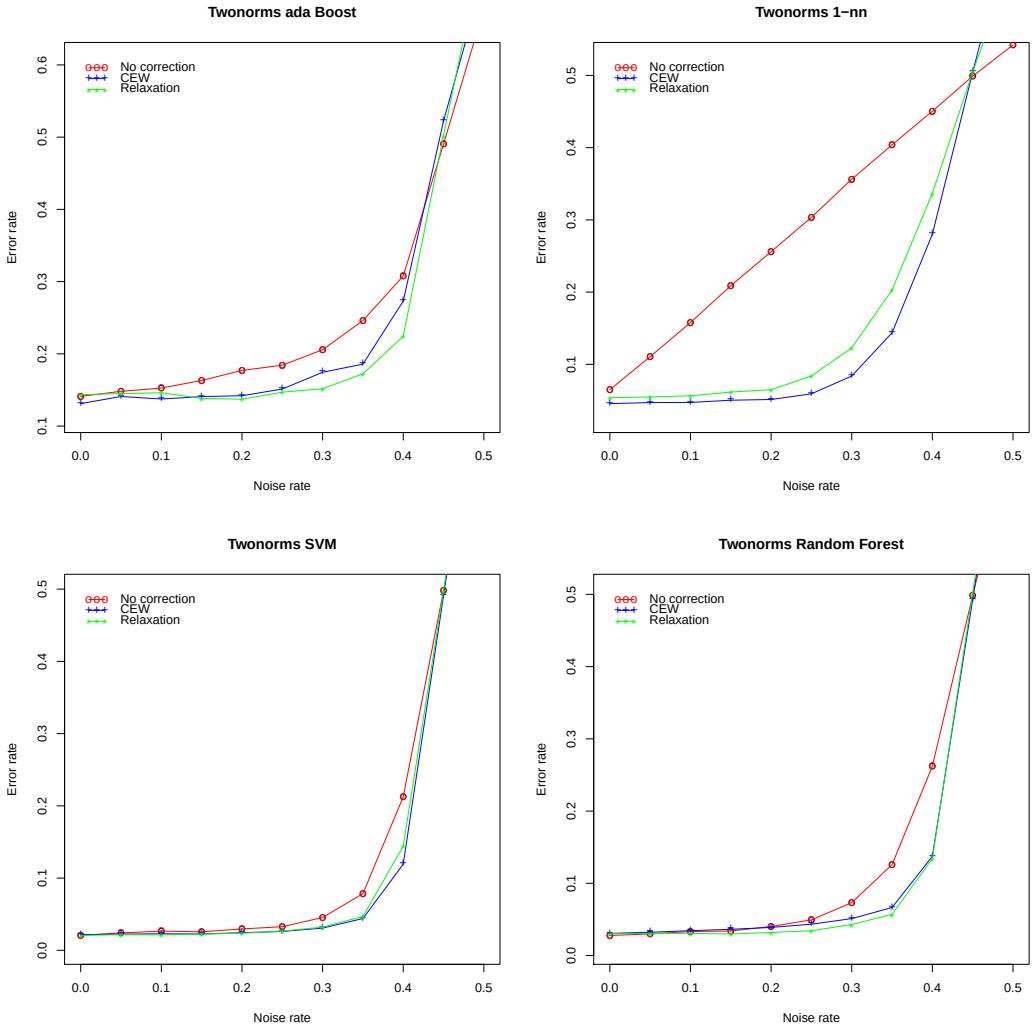


Figure 8: Filtering Improvement on *Twonorm* Dataset

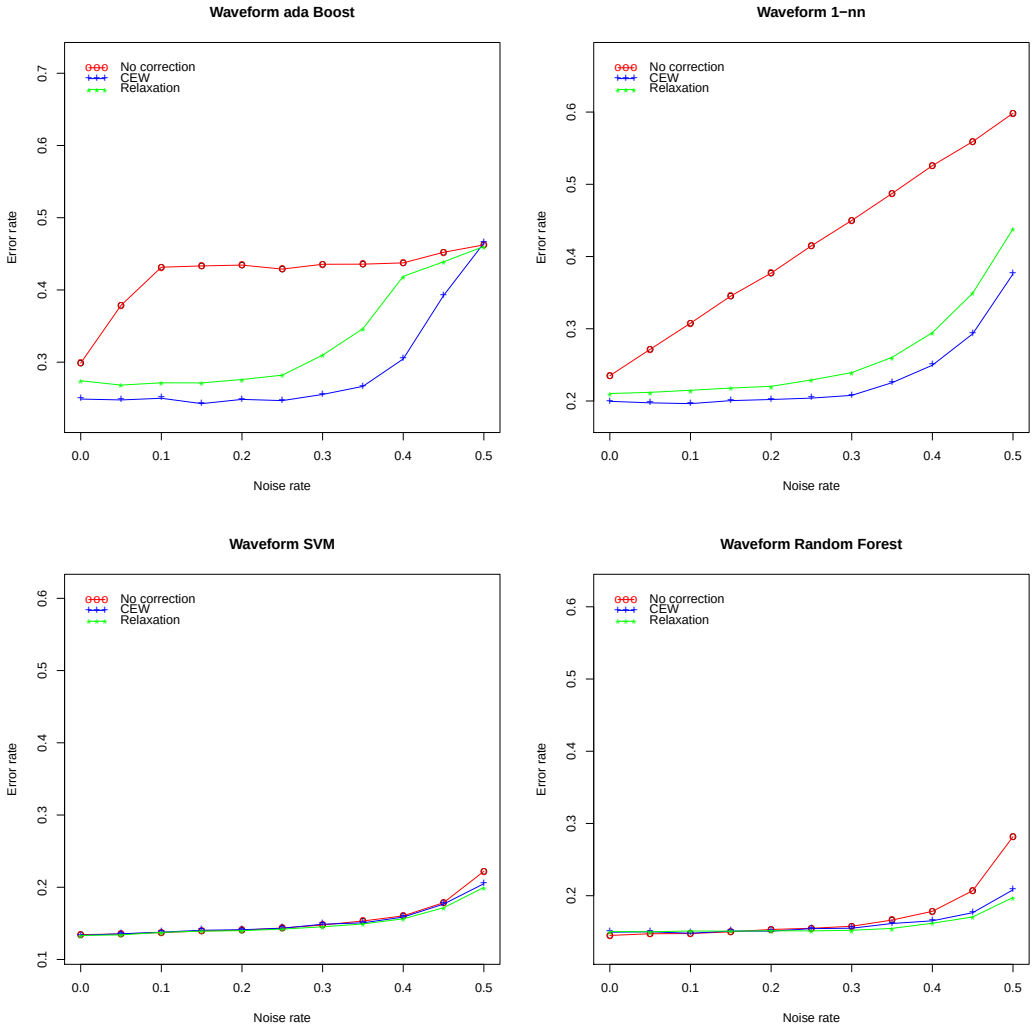


Figure 9: Filtering Improvement on *Waveform* Dataset

7. Conclusion and Future Work

In this paper we have proposed two solutions for detecting noisy labels in learning datasets. An extensive evaluation has shown that both methods proposed here are very efficient since they are able to detect around 90% of the noise for about 20% false alerts.

Several possibilities may be considered to improve our methods:

- SVM shows insensitivity to the correction of the label corruption performed by our methods. We think that some improvements might be achieved in the way that detected mislabeled instances are removed.
- The class imbalance is a major problem in machine learning [43]. Depending on the learning method used, different strategies have been proposed to deal with class imbalance (e.g., for decision trees [44] or for self-organizing learning [45]). Can the class imbalance affect CEW and RT methods? CEW is only partially affected by the class imbalance: it compares an observed proportion and a global fixed proportion. The class imbalance can only have an impact through the size of the geometrical neighborhood of the considered instance. This size is probably smaller if the considered instance belongs to the minority class, which affects the statistical significance of the test statistic. The same applies to RT which is based on two criteria, local similarity and global similarity. Only the overall similarity may be affected by the class imbalance. An interesting idea would be to distinguish the outliers on the majority class and the minority class outliers.
- The adaptation of CEW and RT methods to big data is an important issue, which should be sorted out through the use of graph sampling methods.

References

[1] B. Frenay, M. Verleysen, Classification in the presence of label noise: a survey, IEEE Transactions on Neural Networks and Learning Systems PP (99). doi:10.1109/TNNLS.2013.2292894.

[2] K. R. Gabriel, R. R. Sokal, A new statistical approach to geographic variation analysis, Systematic Zoology 18 (1969) 259–278.

[3] G. Toussaint, The relative neighborhood graph of a finite planar set, Pattern recognition 12 (1980) 261–268.

[4] C. T. Zahn, Graph-theoretical methods for detecting and describing Gestalt clusters, IEEE Transactions on Computers C (20) (1971) 68–86.

[5] R. Urquhart, Graph theoretical clustering based on limited neighbourhood sets, Pattern Recognition 15 (3) (1982) 173–187.

[6] F. Muhlenbach, S. Lallich, A new clustering algorithm based on regions of influence with self-detection of the best number of clusters, in: W. Wang, H. Kargupta, S. Ranka, P. S. Yu, X. Wu (Eds.), ICDM 2009, The Ninth IEEE International Conference on Data Mining, Miami, Florida, USA, 6-9 December 2009, IEEE Computer Society, 2009, pp. 884–889.

[7] M. Paterson, F. F. Yao, On nearest-neighbor graphs, in: W. Kuich (Ed.), Automata, Languages and Programming, 19th International Colloquium, ICALP92, Vienna, Austria, July 13-17, 1992, Proceedings, Vol. 623 of Lecture Notes in Computer Science, Springer, 1992, pp. 416–426.

[8] D. Eppstein, M. Paterson, F. F. Yao, On nearest-neighbor graphs, Discrete & Computational Geometry 17 (3) (1997) 263–282.

[9] S. Theodoridis, K. Koutroumbas, Pattern Recognition, 4th Edition, Academic Press, 2008.

[10] G. C. Osbourn, R. F. Martinez, Empirically defined regions of influence for cluster analysis, Pattern Recognition 28 (11) (1995) 1793–1806.

[11] J. W. Jaromczyk, G. T. Toussaint, Relative neighborhood graphs and their relatives, Proceedings of the IEEE 80 (9) (1992) 1502–1517.

[12] F. Muhlenbach, R. Rakotomalala, Multivariate supervised discretization, a neighborhood graph approach, in: Proceedings of the 2002 IEEE International Conference on Data Mining (ICDM 2002), 9-12 December 2002, Maebashi City, Japan, IEEE Computer Society, 2002, pp. 314–321.

[13] G. T. Toussaint, Geometric proximity graphs for improving nearest neighbor methods in instance-based learning and data mining, International Journal of Computational Geometry and Applications 15 (2) (2005) 101–150.

[14] A. D. Cliff, J. K. Ord, Spatial processes, models & applications, Pion Limited, London, 1986.

[15] G. J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, Wiley Interscience, 2004.

[16] N. Cristianini, J. Shawe-Taylor, A. Elisseeff, J. S. Kandola, On kernel-target alignment, in: T. G. Dietterich, S. Becker, Z. Ghahramani (Eds.), Advances in Neural Information Processing Systems 14 [Neural Information Processing Systems: Natural and Synthetic, NIPS 2001, December 3-8, 2001, Vancouver, British Columbia, Canada], MIT Press, 2001, pp. 367–373.

[17] C. H. Nguyen, T. B. Ho, Kernel matrix evaluation, in: M. M. Veloso (Ed.), IJCAI 2007, Proceedings of the 20th International Joint Conference on Artificial Intelligence, Hyderabad, India, January 6-12, 2007, pp. 987–992.

[18] D. A. Zighed, S. Lallich, F. Muhlenbach, A statistical approach to class separability, Applied Stochastic Models in Business and Industry (ASMBI) 21 (2) (2005) 187–197.

[19] A. M. Mood, The distribution theory of runs, The Annals of Mathematical Statistics 11 (4) (1940) 367–392.

[20] A. Wald, J. Wolfowitz, On a test wether two samples are from the same population, The Annals of Mathematical Statistics 11 (2) (1940) 147–162.

[21] A. K. Jain, R. C. Dubes, Algorithms for clustering data, Prentice Hall, 1988.

[22] G. E. Noether, A central limit theorem with nonparametric applications, The Annals of Mathematical Statistics 41 (1970) 1753–1755.

[23] P. A. P. Moran, The interpretation of statistical maps, Journal of the Royal Statistical Society, Series B (Methodological) 10 (2) (1948) 246–251.

[24] H. Hacid, D. A. Zighed, An effective method for locally neighborhood graphs updating, in: K. V. Andersen, J. K. Debenham, R. Wagner (Eds.), Database and Expert Systems Applications, 16th International Conference, DEXA 2005, Copenhagen, Denmark, August 22-26, 2005, Proceedings, Vol. 3588 of Lecture Notes in Computer Science, Springer, 2005, pp. 930–939.

[25] J. Leskovec, C. Faloutsos, Sampling from large graphs, in: T. Eliassi-Rad, L. H. Ungar, M. Craven, D. Gunopulos (Eds.), Proceedings of the Twelfth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, Philadelphia, PA, USA, August 20-23, 2006, ACM, 2006, pp. 631–636.

[26] K. Bache, M. Lichman, UCI machine learning repository (2013).
URL http://archive.ics.uci.edu/ml

[27] F. Leisch, E. Dimitriadou, mlbench: Machine Learning Benchmark Problems, r package version 2.1-1 (2010).

[28] L. Breiman, Bias, variance, and arcing classifiers, Technical Report 460, Statistics Department, University of California, Berkeley, CA, USA (1996).

[29] Y. Freund, R. Schapire, A decision-theoretic generalization of on-line learning and an application to boosting, Journal of Computer and System Sciences 55 (1) (1997) 119–139.

[30] J. R. Quinlan, C4.5: Programs for Machine Learning, Morgan Kaufmann Publishers, 1993.

[31] K. Hornik, C. Buchta, A. Zeileis, Open-source machine learning: R meets Weka, Computational Statistics 24 (2) (2009) 225–232.

[32] T. M. Cover, P. E. Hart, Nearest neighbor pattern classification, IEEE Transactions on Information Theory 13 (1) (1967) 21–27.

[33] W. N. Venables, B. D. Ripley, Modern Applied Statistics with S, 4th Edition, Springer, New York, 2002.

[34] C. Cortes, V. Vapnik, Support-vector networks, Machine Learning 20 (3) (1995) 273–297.

[35] A. Karatzoglou, A. Smola, K. Hornik, A. Zeileis, kernlab – an S4 package for kernel methods in R, Journal of Statistical Software 11 (9) (2004) 1–20.

[36] L. Breiman, Random forests, Machine Learning 45 (1) (2001) 5–32.

[37] A. Liaw, M. Wiener, Classification and Regression by randomForest, R News 2 (3) (2002) 18–22.

[38] N. Landwehr, M. Hall, E. Frank, Logistic model trees, Machine Learning 59 (1-2) (2005) 161–205.

[39] J. Friedman, T. Hastie, R. Tibshirani, Additive logistic regression: a statistical view of boosting, Annals of Statistics 28 (2) (2000) 337–407.

[40] C. A. Shipp, L. I. Kuncheva, Relationships between combination methods and measures of diversity in combining classifiers, Information Fusion 3 (2) (2002) 135–148.

[41] F. Muhlenbach, S. Lallich, D. A. Zighed, Identifying and handling mislabelled instances, Journal of Intelligent Information Systems (JIIS) 22 (1) (2004) 89–109.

[42] D. A. Zighed, D. Tounissoux, J. P. Auray, C. Largeron, Discrimination by optimizing a local consistency criterion, in: B. Bouchon-Meunier, R. R. Yager, L. A. Zadeh (Eds.), Uncertainty in Knowledge Bases, 3rd International Conference on Information Processing and Management of Uncertainty in Knowledge-Based Systems, IPMU '90, Paris, France, July 2-6, 1990, Proceedings, Vol. 521 of Lecture Notes in Computer Science, Springer, 1991, pp. 337–348.

[43] G. M. Weiss, Mining with rarity: a unifying framework, SIGKDD Explorations 6 (1) (2004) 7–19.

[44] D. A. Cieslak, N. V. Chawla, Learning decision trees for unbalanced data, in: W. Daelemans, B. Goethals, K. Morik (Eds.), Machine Learning and Knowledge Discovery in Databases, European Conference, ECML/PKDD 2008, Antwerp, Belgium, September 15-19, 2008, Proceedings, Part I, Vol. 5211 of Lecture Notes in Computer Science, Springer, 2008, pp. 241–256.

[45] Q. Cai, H. He, H. Man, Imbalanced evolving self-organizing learning, Neurocomputing 133 (2014) 258–270.