Applied Intelligence
https://doi.org/10.1007/s10489-021-02447-7

A semi-hard voting combiner scheme to ensemble multi-class probabilistic classifiers

Rosario Delgado[1]

Accepted: 20 April 2021

Abstract
Ensembling of probabilistic classifiers is a technique that has been widely applied in classification, allowing to build a new classifier combining a set of base classifiers. Of the different schemes that can be used to construct the ensemble, we focus on the simple *majority vote* (MV), which is one of the most popular combiner schemes, being the foundation of the meta-algorithm *bagging*. We propose a non-trainable weighted version of the simple *majority vote* rule that, instead of assign weights to each base classifier based on their respective estimated accuracies, uses the confidence level CL, which is the standard measure of the *degree of support* that each one of the base classifiers gives to its prediction. In the binary case, we prove that if the number of base classifiers is odd, the accuracy of this scheme is greater than that of the *majority vote*. Moreover, through a sensitivity analysis, we show in the multi-class setting that its resilience to the estimation error of the probabilities assigned by the classifiers to each class is greater than that of the *average* scheme. We also consider another simple measure of the *degree of support* that incorporates additional knowledge of the probability distribution over the classes, namely the *modified confidence level* MCL. The usefulness for *bagging* of the proposed weighted *majority vote* based on CL or MCL is checked through a series of experiments with different databases of public access, resulting that it outperforms the simple *majority vote* in the sense of a statistically significant improvement regarding two performance measures: Accuracy and Matthews Correlation Coefficient (MCC), while holding up against the *average* combiner, which *majority vote* does not, being less computationally demanding.

Keywords Ensemble classifier · Confidence Level · Majority vote · Average rule · Error sensitivity · Bagging

1 Introduction

1.1 Ensemble of probabilistic classifiers

Classification is one of the main tasks of Supervised Learning. Given a dataset consisting of information about objects relating to some attributes that describe them, and to a categorical output variable (the class to which each object belongs), with $r \geq 2$ different possible classes $y_1, \ldots, y_r$, a classifier is an algorithm that allows to infer the class of a new object or instance from its known attributes. Different methodologies are used in Machine Learning to learn classifiers from a dataset of solved cases in which both, the attributes and the class, are consigned for each of the instances (except missing data). Among them, we are interested in *probabilistic classifiers*, which not only predict the class, but estimate the probability distribution over the set of classes, being the predicted class that one with the highest associated probability, that is, the most likely class that the instance should belong to, following the *maximum a posteriori* (MAP) criterium.

Probabilistic classifiers provide a prediction that can be useful in its own right, and particularly when classifiers are combined to create ensembles. Ensemble of classifiers (also known as "combining classifiers", see [18]) is a technique that has been widely applied in classification learning, the idea being to build a new classifier combining a set of base classifiers, in the hope of improving their behaviour, as it effectively emerges from different works (see [5, 11, 16, 18, 27]). Obviously, an important research topic in this field is that of combination schemes: their comparison and how to choose between them, as well as the type of base classifier

✉ Rosario Delgado
delgado@mat.uab.cat

[1] Department of Mathematics, Universitat Autònoma de Barcelona, Edifici C- Campus de la UAB. Av. de l'Eix Central s/n., 08193 Bellaterra (Cerdanyola del Vallès), Barcelona, Spain

Published online: 09 July 2021

used to build the ensemble. For example, in [20] the authors experimentally prove that the ensembles of Naive Bayes classifiers following different schemes are significantly better than standard Naive Bayes, and also slightly better than an ensemble of Naive Bayes and decision tree.

1.2 Bagging

As single decision classifiers can suffer from high variance, a simple way to address this flaw is to use them in the context of randomization-based ensemble methods, by introducing random perturbations into the learning procedure in order to produce several different classifiers from a single training set. It is the case of *bagging*, an acronym for *Bootstrap AGGregatING*, as it was introduced by Breiman in [6]. Indeed, *bagging* is a meta-classifier used to reduce the variance of a base classifier by introducing randomization into its construction through the use of learning datasets obtained as *bootstrap samples* of the original learning set, that is, fits base classifiers, each on random subsets of the original dataset drawn with replacement, and then aggregate their individual predictions making an ensemble out of them by means of the *majority vote* combiner scheme. With this procedure we lose the interpretability of a single simple classifier, but potentially gain in predictive power.

In many cases, bagging is a simple way to improve the predictive power of a single model without needing to change the underlying classifier. The improvement on it, using the *bagging* procedure, is obtained for unstable underlying classifiers, as is the case of a *decision tree* (see [6]), which will be the one we will use in our experimental phase. Paraphrasing Breiman ([6]), we can say that *"The evidence[...] is that bagging can push a good but unstable procedure a significant step towards optimality. On the other hand, it can slightly degrade the performance of stable procedures."*.

Random forest, which are outstanding examples of probabilistic classifiers, were also introduced by Breiman in [7], as a variant of *bagging* in which a randomization of the input attributes is used when considering candidates to split internal nodes (see also [26]). In addition, ensemble of classifiers is also the main idea behind *boosting* (see for instance [14] for general description of boosting and [10] for an application in the field of medicine). A comparison of the effectiveness of *randomization*, *bagging* and *boosting* to improve the performance of the *decision tree* algorithm C4.5 can be found in [12].

1.3 Combiner schemes

In general, we build $M \geq 2$ different base probabilistic classifiers, $\mathcal{C}_1, \ldots, \mathcal{C}_M$, which may or may not correspond to the *bagging* meta-algorithm, and then combine their outputs to construct an ensemble. The final decision of the ensemble is derived using a combination rule, that can fall into one of following two groups (see for instance [30]):

i) **Hard voting:** combination rules that apply to class labels, as the simple *majority vote*, which is just by going with the prediction that appears the most times in the base classifiers, and is the one used by *bagging*. The criterion of the *majority vote* scheme coincides, in the binary setting, with the classical *Condorcet* criterion, according to which to be the winner a class must win one-on-one matches with all other classes, that is, must be preferred to each other class when compared to them one at a time.

ii) **Soft voting:** combination rules obtained by polling the continuous outputs of each base classifier using a function (average, maximum, minimum, product,... see [18, 19]) that returns the class label that maximizes the value of the function applied to the predicted probabilities, the *average* being the strongest from a viewpoint of predictive power. The *average* combiner is a natural competitor of the *majority vote* for *bagging*, showing similar results in the experimental phase of [6] (Section 6.1). Our experimental evidence is different, however, as we will see in Section 6.3, finding confirmation that the *average* scheme outperforms the *majority vote*, although at the price of a greater computational requirement.

The combiner scheme that we will introduce in this work is halfway between the *majority vote* and the *average*, we will see latter in what sense, so it can be considered a **semi-hard voting** scheme.

Other possible grouping of the combination rules is trainable vs. non-trainable. The simple *majority vote* and the *average* are non-trainable, but their usual weighted counterpart are trainable, since the weights are determined through a separate training algorithm, usually as a function of the estimated accuracies of the base classifiers. Using non-trainable rules provides simplicity and is less memory and computationally demanding.

The combiner scheme that we propose is **non-trainable** although it is a weighted version of the simple *majority vote*, because its weights are obtained just as a *measure of the degree of support* that any of the base classifiers assigns to the class it predicts. As measure of the degree of support we propose the *confidence level* CL, which for each base classifier is the *degree of support* to its own choice. But it could also be natural to think about assigning another alternative measure, namely the *modified confidence level* (MCL), which takes into account additional knowledge of the probability distribution between classes. The corresponding non-trainable weighted versions of

the simple *majority vote* scheme are named in what follows: **CL-MV** and **MCL-MV**, for Confidence Level (respectively, Modified Confidence Level) based Majority Vote.

From a heuristic perspective, our contribution consists of supporting the following hypotheses through experimentation with different datasets:

- Main hypothesis: using CL-MV (or MCL-MV) instead of simple *majority vote*, the obtained scheme significantly improves the classifying power of *bagging*.
- Secondary hypotheses:
  - Although there is no clear significant difference between them, in some cases the MCL-MV scheme gives better results than CL-MV (and in some others, the opposite).
  - Although *bagging* with the *average* rule outperforms that with the simple *majority vote*, the same does not happen with CL-MV nor MCL-MV, which hold up and do not show statistically significant differences with the *average*, while they are computationally less demanding.

And from a theoretical perspective, our main contribution is to obtain two important results on the combiner schemes comparison:

- In the binary case, we prove that if the number of base classifiers is odd, the accuracy of CL-MV is greater than that of the *majority vote*, if the probability for each base classifier to give correct label is big enough (Section 3).
- In the general multi-class setting, we perform a sensitivity analysis (Section 4) showing that under some reasonable hypothesis, CL-MV is more resilient to probability estimation errors than both the *average* and the *product* rules.

1.4 Measures of performance comparison

To compare the predictive power of the different combiner schemes, we will perform a series of experiments using real datasets following the *bagging* procedure. We denote by $C = (C_{ij})_{i,j=1,\ldots,r}$ a general confusion matrix, with C_{ij} being the number of instances in the testing dataset that belong to class j and have been assigned to class i by the classifier, and we compare the goodness of the ensembles as classifiers on unseen data in the validation process using two different performance measures that can be applied both in the binary and in the multi-class setting:

- **Accuracy:** this performance measure is one of the most intuitive and appealing, and is defined from C in this way:

$$\text{Accuracy} = \frac{\sum_{i=1}^{r} C_{ii}}{\sum_{i=1}^{r} \sum_{j=1}^{r} C_{ij}} .$$

Accuracy ranges between 0 and 1, the latter corresponding to perfect classification.

- **Matthews Correlation Coefficient (MCC):** is a more subtle performance measure, which was first introduced in the binary case by B.W. Matthews [21] as a measure of association obtained by discretization of the Pearson's correlation coefficient for two binary vectors, and was generalized in [15] to multi-class classification. MCC has proven to be more reliable as a metric for classification than Cohen's Kappa, which has been used in many works for a long time (see [9]). Its definition is as follows:

$$\text{MCC} = \frac{\sum_{k,\ell,m=1}^{r} (C_{kk}\, C_{\ell m} - C_{mk}\, C_{k\ell})}{\sqrt{\sum_{k=1}^{r} \left(\left(\sum_{\ell=1}^{r} C_{k\ell} \right) \left(\sum_{u,v=1,\, u\neq k}^{r} C_{uv} \right) \right)} \sqrt{\sum_{k=1}^{r} \left(\left(\sum_{\ell=1}^{r} C_{\ell k} \right) \left(\sum_{u,v=1,\, u\neq k}^{r} C_{vu} \right) \right)}}$$

MCC also assumes its theoretical maximum value of 1 when classification is perfect, but ranges between -1 and 1.

In both cases, the larger the metric value, the better the classifier performance.

1.5 Description of the sections

The remainder of the paper is structured as follows. After introducing the CL-MV combiner scheme in Section 2, in Section 3 we compare the accuracies of CL-MV and the *majority vote* in the binary case, and in Section 4 we investigate the sensitivity of the *average*, *product* and CL-MV schemes to probability estimation errors, in the general multi-class setting. The Modified Confidence Level MCL is introduced in Section 5 as a *degree of support* in classification alternative to the confidence level CL, showing some properties in Appendix A. Without diving into computationally demanding experiments, Section 6 describes the used datasets, the experimental design aimed to compare the predictive power and the computational complexity of the

considered combiner schemes for *bagging*, and the obtained results (of which some complementary tables are allocated in Appendix B). We finish with a discussion in Section 7, and some words by way of conclusion in Section 8.

2 The CL-MV combiner scheme

We build a novel ensemble classifier from M base classifiers $\mathcal{C}_1, \ldots, \mathcal{C}_M$, by introducing a modification of the simple *majority vote* scheme, and we name it the *Confidence Level based Majority Vote*, CL-MV. This combiner scheme uses the classifications given by the base classifiers themselves along with their corresponding estimated probability distributions, in this way:

If p_{jk} denotes the probability that classifier j assigns to class k, the predicted class by this classifier following the *maximum a posteriori probability* (MAP) criterium, is that with the largest assigned probability. That is, the predicted class by the j-th probabilistic classifier is

$$y_j^* = y_\ell \text{ if } \ell = \arg\max_{k=1,\ldots,r} p_{jk}\,, \quad \text{and} \quad \mathrm{CL}_j = \max_{k=1,\ldots,r} p_{jk}$$

is said to be the *confidence level* of the prediction, which is interpreted as a *degree of support* to it. (It is understood that if the point where the maximum is reached is not unique, one of them is chosen by a tie-breaking rule.) In general, a combiner scheme predicts in the following way:

$$y^*_{ensemble} = y_\ell \quad \text{with} \quad \ell = \arg\max_{k=1,\ldots,r} g_k \tag{1}$$

for some function g_k. Consider the following particular cases:

1. *Majority vote:* $g_k = \sum_{j=1}^M d_{jk}$, where $d_{jk} = \begin{cases} 1 & \text{if } y_j^* = y_k \\ 0 & \text{otherwise.} \end{cases}$
   Weighted majority vote: $g_k = \sum_{j=1}^M \omega_j\, d_{jk}$, where d_{jk} is as in the simple *majority vote*, and ω_j is the weight assigned to classifier $\mathcal{C}_j$, usually obtained from its estimated accuracy (trainable combiner).
2. *Average:* $g_k = \frac{1}{M} \sum_{j=1}^M p_{jk}$ (the *Sum* combiner is equivalent, with the same g_k but without dividing by M).
3. *Product:* $g_k = \prod_{j=1}^M p_{jk}$
4. *Minimum:* $g_k = \min_{j=1,\ldots,M} p_{jk}$
5. *Maximum:* $g_k = \max_{j=1,\ldots,M} p_{jk}$

We propose the following non-trainable weighted version of the *majority vote* combiner scheme, based on the confidence level CL as it degree of support:

6. CL-MV: $g_k = \sum_{j=1}^M \omega_j\, d_{jk}$, with d_{jk} as in the *majority vote* and $\omega_j = \mathrm{CL}_j$.

Note that ω_j needs no other separate training algorithm to be learned, making CL-MV a **non-trainable** combiner scheme, and that for each classifier, it only uses the maximum of its probability distribution, unlike the *average*, which uses all the values of the probability distribution. Also note that by definition of d_{jk}, g_k only depends on the value of the weight ω_j for those j such that $d_{jk} = 1$, that is, when $y_j^* = y_k$, and we can assume without loss of generality that otherwise $\omega_j = 0$. For that, we introduce the notation $y(k) = \{j = 1, \ldots, M : y_j^* = y_k\}$ for any $k = 1, \ldots, r$, and then, with this notation, we can rewrite:

1. *Majority vote:* $g_k = \#\, y(k)$.
   Weighted majority vote: $g_k = \sum_{j \in y(k)} \omega_j$ where ω_j is the weight assigned to classifier $\mathcal{C}_j$ by a separate learning algorithm.
6. CL-MV: $g_k = \sum_{j \in y(k)} \omega_j$ where $\omega_j = \mathrm{CL}_j$.

(Here and in the sequel, we use the convention that a sum over an empty set, is zero.)

Remark 1 On the binary case ($r = 2$), we see in Proposition 1 below that under certain circumstances, CL-MV matches the *average* scheme.

Proposition 1 *In binary classification ($r = 2$), suppose that $y^*_{CL\text{-}MV} = y_k$. Therefore, if $\#\, y(k) \leq M/2$, we can ensure that $y^*_{Average} = y_k$, that is, CL-MV and the average schemes give the same prediction.*

Before giving the proof, let's look at two examples in Table 1, with $M = 5$ classifiers and $r = 2$ classes, where the probabilities p_{jk} are listed: in example a), the hypothesis and the thesis of Proposition 1 are fulfilled, in example b), neither the hypothesis nor the thesis are (it is a counterexample that without the hypothesis, the thesis is no longer true).

Table 1 Illustrative examples of Proposition 1

	Example a)		Example b)	
Classifier	y_1	y_2	y_1	y_2
$\mathcal{C}_1$	**0.55**	0.45	**0.55**	0.45
$\mathcal{C}_2$	**0.60**	0.40	**0.60**	0.40
$\mathcal{C}_3$	**0.65**	0.35	**0.65**	0.35
$\mathcal{C}_4$	0.05	**0.95**	0.20	**0.80**
$\mathcal{C}_5$	0.10	**0.90**	0.15	**0.85**
Average	0.39	**0.61**	0.43	**0.57**
CL-MV	1.80	**1.85**	**1.80**	1.65

The sum of the CL-MV weights for each example is the sum of the probabilities in boldface for each class, which are the confidence levels CL

Indeed, example a) in Table 1 shows that $y^*_{Average} = y_2 = y^*_{\text{CL-MV}}$ and $\#\, y(2) = 2 \leq 5/2$, while example b) gives $y^*_{Average} = y_2$ and $y^*_{\text{CL-MV}} = y_1$, which is possible since $\#\, y(1) = 3 > 5/2$.

Proof of Proposition 1 Without loss of generality we can assume that $y^*_{\text{CL-MV}} = y_1$. This means that

$$\sum_{j \in y(1)} \text{CL}_j \geq \sum_{j \in y(2)} \text{CL}_j \tag{2}$$

Moreover, we assume that $\#\, y(1) \leq M/2$, which implies that $\#\, y(1) \leq \#\, y(2)$ since $M = \#\, y(1) + \#\, y(2)$. Then, from (2) we have that

$$\sum_{j \in y(1)} \left(1 - \text{CL}_j\right) \leq \sum_{j \in y(2)} \left(1 - \text{CL}_j\right) \tag{3}$$

and then, adding term to term (2) and (3), we obtain

$$\sum_{j \in y(1)} \text{CL}_j + \sum_{j \in y(2)} \left(1 - \text{CL}_j\right) \geq \sum_{j \in y(2)} \text{CL}_j + \sum_{j \in y(1)} \left(1 - \text{CL}_j\right),$$

that is, $\sum_{j=1}^{M} p_{j1} \geq \sum_{j=1}^{M} p_{j2}$, which implies that $y^*_{Average} = y_1$.

(If CL-MV classifies without ties, that is, $\sum_{j \in y(1)} \text{CL}_j > \sum_{j \in y(2)} \text{CL}_j$, therefore, $\sum_{j=1}^{M} p_{j1} > \sum_{j=1}^{M} p_{j2}$, that is, the *average* scheme does too.) □

In words, unlike what happens with the simple *majority vote*, which chooses the class with the most votes from among the base classifiers (for each classifier, the counter of each class adds 1 if the classifier predicts that class, and 0 otherwise, and the combiner chooses the class with the highest counter), with the CL-MV combiner scheme, the counter of each class adds the measure of the degree of support (CL_j) if the classifier $\mathcal{C}_j$ predicts that class, and 0 otherwise. Let us see the toy example in Table 2 below, in which we have $M = 5$ classifiers and $r = 3$ classes, and the probability distributions provided by each classifier. The corresponding predictions with any of the base classifiers and with the non-trainable ensembles are also given.

As can be seen in Table 2, the predictions with the ensembles are:

$y^*_{\text{Sum}} = y^*_{\text{Average}} = y^*_{\text{Product}} = y^*_{\text{Minimum}} = y_3$,
$y^*_{\text{Majority}} = y_1$, $y^*_{\text{CL-MV}} = y^*_{\text{Maximum}} = y_2$.

The main highlights of the CL-MV combiner scheme are:

1. It is a fusion not of the predictions but of the *degree of support* given to the predictions, where this *degree of support* is defined from the probability distribution over the classes assigned by the classifier, as the *confidence level*.
2. It is **non-trainable**, that is, no extra parameters need to be trained and it is ready to run as soon as the base classifiers are available.
3. It can provide different predictions both from those provided by the *majority vote* and by the *average* criteria (see the toy example in Table 2).
4. In the binary case, we found a scenario where it gives the same prediction as the *average* (Proposition 1), and we will show that its accuracy is greater than that of the *majority vote* (see Section 3 below).
5. From the point of view of the sensitivity to probability estimation errors, under some reasonable hypotheses, CL-MV is more resilient than both the *product* and the *average* schemes (see Section 4 underneath).

The pseudo-code for the algorithm of classification corresponding to the ensemble given by the novel CL-MV combiner scheme is **Algorithm 1** below, and presents the benefit of being simple and easy to implement. For the

Table 2 In boldface CL in the probability distributions, and also the maximum of the *sum*, the *average*, the *product*, the *minimum* and the *maximum* of the predicted probabilities, and the maximum of the sum of the weights for both the *majority vote* and the CL-MV combiners

	Distribution			Majority vote			CL-MV		
Classifier	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3
$\mathcal{C}_1$	**0.55**	0.00	0.45	1	0	0	0.55	0	0
$\mathcal{C}_2$	**0.50**	0.05	0.45	1	0	0	0.50	0	0
$\mathcal{C}_3$	**0.50**	0.02	0.48	1	0	0	0.50	0	0
$\mathcal{C}_4$	0.00	**0.85**	0.15	0	1	0	0	0.85	0
$\mathcal{C}_5$	0.05	**0.75**	0.20	0	1	0	0	0.75	0
Sum	1.60	1.67	**1.73**	**3**	2	0	1.55	**1.60**	0
Average	0.32	0.334	**0.346**						
Product	0.00	0.00	**0.002916**						
Minimum	0.00	0.00	**0.15**						
Maximum	0.55	**0.85**	0.48						

sake of completeness, the pseudo-code for the standard *majority vote* and the *average* schemes are also included as **Algorithm 2** and **Algorithm 3**, respectively. Those for the *product*, *minimum* and *maximum* combiner schemes are analogous to **Algorithm 3**, simply modifying the definition of g_k in line 4 properly, and then omitted.

Algorithm 1 `CL-MV combiner scheme.`

Input evidence E, classifiers $\mathcal{C}_1, \ldots, \mathcal{C}_M$
Output the predicted class $y^*_{\text{CL-MV}}$

1: **for** j in $1 : M$ **do**
2: **for** k in $1 : r$ **do**
3: **compute** p_{jk}, the probability assigned by $\mathcal{C}_j$ to class k, given the evidence E
4: $\ell = \arg\max_{k=1,\ldots,r} p_{jk}$
5: $y^*_j = y_\ell$
6: **for** k in $1 : r$ **do**
7: $y(k) = \{j = 1, \ldots, M : y^*_j = y_k\}$
8: **if** $y(k) = \emptyset$ **then**
9: $g_k = 0$
10: **else**
11: **for** j in $y(k)$ **do**
12: $\text{CL}_j = \max_{k=1,\ldots,r} p_{jk}$
13: $g_k = \sum_{j \in y(k)} \text{CL}_j$
14: $\ell = \arg\max_{k=1,\ldots,r} g_k$
15: $y^*_{\text{CL-MV}} = y_\ell$
return $y^*_{\text{CL-MV}}$

Algorithm 2 `(simple) Majority vote combiner scheme.`

Input evidence E, classifiers $\mathcal{C}_1, \ldots, \mathcal{C}_M$
Output the predicted class y^*_{Majority}

1: **for** j in $1 : M$ **do**
2: **for** k in $1 : r$ **do**
3: **compute** p_{jk}, the probability assigned by $\mathcal{C}_j$ to class k, given the evidence E
4: $\ell = \arg\max_{k=1,\ldots,r} p_{jk}$
5: $y^*_j = y_\ell$
6: **for** k in $1 : r$ **do**
7: $y(k) = \{j = 1, \ldots, M : y^*_j = y_k\}$
8: **if** $y(k) = \emptyset$ **then**
9: $g_k = 0$
10: **else**
11: $g_k = \#\, y(k)$
12: $\ell = \arg\max_{k=1,\ldots,r} g_k$
13: $y^*_{\text{Majority}} = y_\ell$
return y^*_{Majority}

Algorithm 3 `Average combiner scheme.`

Input evidence E, classifiers $\mathcal{C}_1, \ldots, \mathcal{C}_M$
Output the predicted class $y^*_{Average}$

1: **for** k in $1 : r$ **do**
2: **for** j in $1 : M$ **do**
3: **compute** p_{jk}, the probability assigned by $\mathcal{C}_j$ to class k, given the evidence E
4: $g_k = \frac{1}{M} \sum_{j=1,\ldots,M} p_{jk}$
5: $\ell = \arg\max_{k=1,\ldots,r} g_k$
6: $y^*_{Average} = y_\ell$
return $y^*_{Average}$

Remark 2 As we can verify empirically in Section 6.3, from a computational and saving of storage space point of view, the *majority vote* (hard voting) presents the advantage that once we know the prediction of any of the base classifiers, y^*_j, we do not need to store any other information about the probability distributions of the predictions. At the other extreme, the *average* scheme (soft voting) needs to store and use all the values of the distribution, making it more computationally and storage-demanding. The CL-MV combiner is halfway between them, using only the maximum of the distribution. That is why we say that CL-MV is a *semi-hard* voting combiner scheme.

3 Accuracy in the binary case

Majority vote is one of the most popular combiner schemes, if not the most. In Section 4.2 [19], to find out why that is so, the author studies in deep its *accuracy*, assuming that

i) the number of classes is $r = 2$ (binary case),
ii) the number of classifiers M is odd, say $M = 2L + 1$ with $L \geq 1$,
iii) the base classifiers outputs are independent (this condition may seem unrealistic, but for many applications it holds, at least approximately),
iv) the probability for each base classifier to give correct class label is $p \in (0, 1)$.

The *majority vote* will predict an accurate class label if the simple majority of the base classifiers vote for it, that is, if at least $\lfloor M/2 + 1 \rfloor$ of them give the correct prediction (where $\lfloor x \rfloor$ denotes the integer part or *floor* of x). Therefore, the accuracy of the ensemble based on the *majority vote* combiner is:

$$Acc_{majority} = \sum_{\ell=\lfloor M/2 \rfloor+1}^{M} \binom{M}{\ell} p^\ell (1-p)^{M-\ell} \tag{4}$$

Then, it can be seen (Condorcet Jury Theorem, 1785 [24]) that

a) if $p > 0.5$, $\lim_{M\to\infty} Acc_{majority} = 1$ and it is monotonically increasing,
b) if $p = 0.5$, $Acc_{majority} = 0.5$ for all M,
c) if $p < 0.5$, $\lim_{M\to\infty} Acc_{majority} = 0$ and it is monotonically decreasing.

And this result supports the intuition that we can expect improvement over the individual accuracy p only if $p > 0.5$.

In the same scenario, what is the formula for the accuracy of the ensemble based on CL-MV? This combiner scheme gives the correct prediction if the number of classifiers that predict correctly is at least α, with α being the minimum integer such that

$$p\,\alpha > (1-p)\,(M-\alpha),$$

which is equivalent to say that α is the minimum integer such that $\alpha > (1-p)\,M$. Then, $\alpha = \lfloor (1-p)\,M \rfloor + 1$, and in consequence, the accuracy is:

$$Acc_{\text{CL-MV}} = \sum_{\ell=\lfloor (1-p)\,M\rfloor+1}^{M} \binom{M}{\ell} p^{\ell}\,(1-p)^{M-\ell} \tag{5}$$

We can easily prove the next result:

Proposition 2 *In the binary case and with an odd number of base classifiers M, we have that*

$$\begin{cases} \text{If } p > \frac{M+1}{2M}, & \text{then } Acc_{\text{CL-MV}} > Acc_{majority}, \\ \text{If } \frac{M-1}{2M} \le p \le \frac{M+1}{2M}, & \text{then } Acc_{\text{CL-MV}} = Acc_{majority}, \\ \text{If } p < \frac{M-1}{2M}, & \text{then } Acc_{\text{CL-MV}} < Acc_{majority}. \end{cases}$$

In particular, since $(M+1)/(2M) > 1/2$, Proposition 2 implies that if $p > (M+1)/(2M)$, CL-MV strictly improves accuracy with respect to the *majority vote* combiner scheme, and by the Condorcet Jury Theorem, $Acc_{\text{CL-MV}}$ is monotonically increasing and

$$\lim_{M\to\infty} Acc_{\text{CL-MV}} = 1\,.$$

Proof By (4) and (5) we see that $Acc_{\text{CL-MV}} > Acc_{majority}$ if and only if

$$\lfloor M/2 \rfloor \ge \lfloor (1-p)\,M \rfloor + 1 \Leftrightarrow \lfloor (1-p)\,M \rfloor \le L-1 \Leftrightarrow (1-p)\,M < L \Leftrightarrow p > \frac{M+1}{2M}.$$

On the other hand, $Acc_{\text{CL-MV}} < Acc_{majority}$ if and only if

$$\lfloor M/2 \rfloor + 1 \le \lfloor (1-p)\,M \rfloor \Leftrightarrow (1-p)\,M \ge L+1 \Leftrightarrow p < \frac{M-1}{2M}.$$

Otherwise, the accuracies are equal. □

4 Error sensitivity

In this section we investigate the sensitivity of the *product*, the *average* (equivalently, the sum) and the CL-MV combiner schemes to probability estimation errors, by following the approach of [18].

Assume for a while that probabilities p_{jk} for $j = 1,\ldots,M$, $k = 1,\ldots,r$ are not computed correctly, rather they suffer from an estimation error. Denote by $\hat{p}_{jk}$ the obtained estimates of the probabilities, which are those used by the combination rules to obtain their predictions. As the model of additive errors is the most popular error model in statistics, we assume that the estimation error is additive, that is, for any j and k,

$$\widehat{p}_{jk} = p_{jk} + \varepsilon_{jk}\,,$$

where errors ε_{jk} are small (in absolute value). In particular, we assume that errors do not affect the individual prediction of any base classifier (y_j^* is not affected by errors). In other words, we assume that for any $j = 1,\ldots,M$,

$$\arg\max_{k=1,\ldots,r} p_{jk} = \arg\max_{k=1,\ldots,r} \widehat{p}_{jk}\,. \tag{6}$$

This implies that $y(k)$ is also unaffected by the probability estimation errors.

We are concerned about the effect that those probability estimation errors will have on the predictions obtained by the ensembles following the different combination rules. By (1), now $y^*_{ensemble} = y_\ell$ with $\ell = \arg\max_{k=1,\ldots,r} \widehat{g}_k$, where $\widehat{g}_k$ denotes the estimate of function g_k obtained substituting probabilities p_{jk} by their estimates $\widehat{p}_{jk}$. In what follows, we concentrate on the *product*, the *average* and the CL-MV combiner schemes.

a. *The product scheme*: $g_k = \prod_{j=1}^{M} p_{jk}$.
Following [18], we can write

$$\widehat{g}_k = \prod_{j=1}^{M} \widehat{p}_{jk} = \prod_{j=1}^{M} (p_{jk}+\varepsilon_{jk}) = \left(\prod_{j=1}^{M} p_{jk}\right) \prod_{i=1}^{M} \left(1+\frac{\varepsilon_{ik}}{p_{ik}}\right)$$
$$\approx \left(\prod_{j=1}^{M} p_{jk}\right)\left(1+\sum_{i=1}^{M}\frac{\varepsilon_{ik}}{p_{ik}}\right) = g_k\left(1+\sum_{i=1}^{M}\frac{\varepsilon_{ik}}{p_{ik}}\right),$$

where we have made a linear approximation (we neglect higher order terms since the errors ε_{jk} are small and therefore, the product of two or more of them is of a very small order). That is, if the probabilities p_{jk} are affected by the additive estimation errors ε_{jk}, then g_k is affected by a multiplicative error Ψ_k^{prod}, where

$$\Psi_k^{prod} = 1 + \sum_{i=1}^{M}\frac{\varepsilon_{ik}}{p_{ik}}.$$

b. *The average scheme*: $g_k = \frac{1}{M}\sum_{j=1}^{M} p_{jk}$.

Following [18] again,

$$\widehat{g}_k = \frac{1}{M}\sum_{j=1}^{M}\widehat{p}_{jk} = \frac{1}{M}\sum_{j=1}^{M}(p_{jk}+\varepsilon_{jk}) = \left(\frac{1}{M}\sum_{j=1}^{M} p_{jk}\right)\left(1+\frac{\sum_{i=1}^{M}\varepsilon_{ik}}{\sum_{i=1}^{M} p_{ik}}\right) = g_k\left(1+\frac{\sum_{i=1}^{M}\varepsilon_{ik}}{\sum_{i=1}^{M} p_{ik}}\right).$$

In this case, if the probabilities p_{jk} are affected by the additive estimation errors ε_{jk}, then g_k is affected by a multiplicative error Ψ_k^{aver}, where

$$\Psi_k^{aver} = 1+\frac{\sum_{i=1}^{M}\varepsilon_{ik}}{\sum_{i=1}^{M} p_{ik}}.$$

c. *The CL-MV scheme*: $g_k = \sum_{j\in y(k)} \mathrm{CL}_j$ (assume that k is such that $y(k)\neq\emptyset$), where $\mathrm{CL}_j = \max_{\ell=1,\ldots,r} p_{j\ell}$.

Therefore,

$$\widehat{g}_k = \sum_{j\in y(k)}\widehat{\mathrm{CL}}_j = \sum_{j\in y(k)}\max_{\ell=1,\ldots,r}\widehat{p}_{j\ell} = \sum_{j\in y(k)}\max_{\ell=1,\ldots,r}\left(p_{j\ell}+\varepsilon_{j\ell}\right). \tag{7}$$

We make the following assumption:

$$(\mathrm{H}_1)\qquad \text{for all } k=1,\ldots,r,\quad k=\arg\max_{\ell=1,\ldots,r}\varepsilon_{j\ell}\quad \text{for all } j\in y(k),$$

that is, $\varepsilon_{jk} = \max_{\ell=1,\ldots,r}\varepsilon_{j\ell}$. In words, for each classifier the error is maximum when predicting the highest probability. Under (H_1) we have that for any $j\in y(k)$,

$$\max_{\ell=1,\ldots,r}\left(p_{j\ell}+\varepsilon_{j\ell}\right) = \mathrm{CL}_j+\varepsilon_{jk}.$$

Then, substituting into (7), we have that under (H_1),

$$\widehat{g}_k = \sum_{j\in y(k)}\left(\mathrm{CL}_j+\varepsilon_{jk}\right) = \left(\sum_{j\in y(k)}\mathrm{CL}_j\right)\left(1+\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)}\mathrm{CL}_i}\right) = g_k\left(1+\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)}\mathrm{CL}_i}\right).$$

Then, if the probabilities p_{jk} are affected by the additive estimation errors ε_{jk}, function g_k is affected by a multiplicative error $\Psi_k^{\text{CL-MV}}$, with

$$\Psi_k^{\text{CL-MV}} = 1+\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)}\mathrm{CL}_i} = 1+\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)} p_{ik}}. \tag{8}$$

Now we can compare the error factors to see which combiner scheme is more resilient to probability estimation errors, and prove the following result (Proposition 3 below). First, we introduce a hypothesis:

$$(\mathrm{H}_2)\text{for all } k=1,\ldots,r,\quad \sum_{j\in y(k)}\varepsilon_{jk}\sum_{j\notin y(k)}\widehat{p}_{jk} \le \sum_{j\notin y(k)}\varepsilon_{jk}\sum_{j\in y(k)}\widehat{p}_{jk}.$$

The rationale for this assumption is that fixed k, for the classifiers $\mathcal{C}_j$ with j in $y(k)$, k is the most probable class, and then $\sum_{j\in y(k)}\widehat{p}_{jk}$ is likely to be sufficiently greater than $\sum_{j\notin y(k)}\widehat{p}_{jk}$ to compensate that $\sum_{j\notin y(k)}\varepsilon_{jk}$ could be less than $\sum_{j\in y(k)}\varepsilon_{jk}$, since the errors are assumed to be small. This is precisely the situation that we will see, as an illustration, for the example in Table 2 (see Table 3 underneath). Proposition 3 states that the *average* rule is much less affected by the probability estimation errors than the *product* rule, and that under reasonable conditions, the CL-MV rule is still less affected by errors than the *average*.

Proposition 3 *With the previous notations, for any* $k=1,\ldots,r$,

a) *If* $\sum_{i=1}^{M}\widehat{p}_{ik} \ge 1+\sum_{i=1}^{M}\varepsilon_{ik}$, *then* $\Psi_k^{aver}\le\Psi_k^{prod}$.
b) *If* $\sum_{i\in y(k)}\widehat{p}_{ik} \ge 1+\sum_{i\in y(k)}\varepsilon_{ik}$, *then* $\Psi_k^{CL\text{-}MV}\le\Psi_k^{prod}$.
c) *Under* (H_2), $\Psi_k^{CL\text{-}MV}\le\Psi_k^{aver}$.

Proof The proof of the statement a) can be found in Section 6 [18], but we reproduce it here for the sake of completeness. Indeed, as $p_{ik}\le 1$, each error ε_{ik} is amplified in Ψ_k^{prod} by dividing it into p_{ik}, while for the average, the errors are not amplified, in such a way that

$$\Psi_k^{prod} = 1+\sum_{i=1}^{M}\frac{\varepsilon_{ik}}{p_{ik}} \ge 1+\sum_{i=1}^{M}\varepsilon_{ik} \ge 1+\frac{\sum_{i=1}^{M}\varepsilon_{ik}}{\sum_{i=1}^{M} p_{ik}} = \Psi_k^{aver},$$

where the second inequality is due to the fact that we are assuming that $\sum_{i=1}^{M}\widehat{p}_{ik} \ge 1+\sum_{i=1}^{M}\varepsilon_{ik}$, which is equivalent to $\sum_{i=1}^{M} p_{ik}\ge 1$. This assumption is likely to happen for the most probable class(es).

The proof of b) is similar for k such that $\sum_{i\in y(k)}\widehat{p}_{ik} \ge 1+\sum_{i\in y(k)}\varepsilon_{ik}$, which is equivalent to $\sum_{i\in y(k)} p_{ik}\ge 1$:

$$\Psi_k^{prod} = 1+\sum_{i=1}^{M}\frac{\varepsilon_{ik}}{p_{ik}} \ge 1+\sum_{i\in y(k)}\varepsilon_{ik} \ge 1+\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)} p_{ik}} = \Psi_k^{\text{CL-MV}}.$$

To see that statement c) holds, we only have to prove that for any $k=1,\ldots,r$ verifying (H_2),

$$\frac{\sum_{i\in y(k)}\varepsilon_{ik}}{\sum_{i\in y(k)} p_{ik}} \le \frac{\sum_{i=1}^{M}\varepsilon_{ik}}{\sum_{i=1}^{M} p_{ik}}. \tag{9}$$

Indeed, this holds since

$$
\begin{aligned}
(9) \Leftrightarrow & \sum_{i\in y(k)} \varepsilon_{ik} \left(\sum_{i\in y(k)} p_{ik} + \sum_{i\notin y(k)} p_{ik} \right) \\
& \leq \sum_{i\in y(k)} p_{ik} \left(\sum_{i\in y(k)} \varepsilon_{ik} + \sum_{i\notin y(k)} \varepsilon_{ik} \right) \\
\Leftrightarrow & \sum_{i\in y(k)} \varepsilon_{ik} \sum_{i\notin y(k)} p_{ik} \leq \sum_{i\in y(k)} p_{ik} \sum_{i\notin y(k)} \varepsilon_{ik} \\
\Leftrightarrow & \sum_{i\in y(k)} \varepsilon_{ik} \sum_{i\notin y(k)} (\widehat{p}_{ik} - \varepsilon_{ik}) \leq \sum_{i\in y(k)} (\widehat{p}_{ik} - \varepsilon_{ik}) \sum_{i\notin y(k)} \varepsilon_{ik} \\
\Leftrightarrow & \sum_{i\in y(k)} \varepsilon_{ik} \sum_{i\notin y(k)} \widehat{p}_{ik} \leq \sum_{i\in y(k)} \widehat{p}_{ik} \sum_{i\notin y(k)} \varepsilon_{ik},
\end{aligned}
$$

which exactly is (H_2). □

To illustrate hypotheses made to establish Proposition 3, we will return to the toy example in Table 2 in Table 3 beneath.

Observe that in Table 3 some of the estimation errors are negative. Indeed, as for the true probabilities $\sum_{k=1}^{r} p_{jk} = 1$ for any $j = 1, \ldots, M$, and we also assume that the same happens for the predicted probabilities, $\sum_{k=1}^{r} \widehat{p}_{jk} = 1$, we have that necessarily, the sum of the prediction errors for any classifier must be zero, that is, $\sum_{k=1}^{r} \varepsilon_{jk} = 0$. In other words, the sum of the prediction errors by row in Table 3 must be 0. Although we will not give the details, we can see that for $\delta < 1$ big enough and $\varepsilon > 0$ small enough, for instance,

$$\delta > 0.20 \quad \text{and} \quad \varepsilon < 0.08\,,$$

all assumptions are met:

0) The basic assumption that $p_{jk} = \widehat{p}_{jk} - \varepsilon_{jk} \in [0, 1]$.
i) Assumption (6).
ii) $\sum_{i=1}^{M} \widehat{p}_{ik} \geq 1 + \sum_{i=1}^{M} \varepsilon_{ik}$ for any $k = 1, 2, 3$.
iii) $\sum_{i\in y(k)} \widehat{p}_{ik} \geq 1 + \sum_{i\in y(k)} \varepsilon_{ik}$ for for $k = 1, 2$ ($y(3) = \emptyset$ and then the condition does not make sense for $k = 3$).

Table 3 Predicted probabilities $\widehat{p}_{jk}$ and estimation errors ε_{jk} in brackets, for the example in Table 2, where $\varepsilon > 0$ is small, and $\delta \in (0, 1)$

	Estimated probability $\widehat{p}_{jk}$ (error ε_{jk})		
Classifier	y_1	y_2	y_3
$\mathcal{C}_1$	**0.55** (ε)	0.00 $(-\delta\,\varepsilon)$	0.45 $(-(1-\delta)\,\varepsilon)$
$\mathcal{C}_2$	**0.50** (ε)	0.05 $(\delta\,\varepsilon)$	0.45 $(-(1+\delta)\,\varepsilon)$
$\mathcal{C}_3$	**0.50** (ε)	0.02 $(\delta\,\varepsilon/2)$	0.48 $(-(1+\delta/2)\,\varepsilon)$
$\mathcal{C}_4$	0.00 $(-\delta\,\varepsilon/2)$	**0.85** (ε)	0.15 $(-(1-\delta/2)\,\varepsilon)$
$\mathcal{C}_5$	0.05 $(\delta\,\varepsilon)$	**0.75** (ε)	0.20 $(-(1+\delta)\,\varepsilon)$

In boldface the confidence levels CL

iv) Hypotheses (H_1) and (H_2).

5 The modified confidence level (MCL)

Although CL quantifies the uncertainty associated with class prediction and it is the usual measure of *degree of support*, it suffers from a shortcoming. Indeed, information provided by CL is very valuable but it could be insufficient to compare predictions made by classifiers in the multi-class setting. Suffice a toy example as argument. Imagine that we are in the 3-class setting and for two classifiers, the probability distributions associated to their predictions are, respectively:

$$(0.6,\ 0.4,\ 0.0) \quad \text{and} \quad (0.2,\ 0.6,\ 0.2)\,.$$

For the first classifier, the predicted class will be y_1, with a confidence level of CL$=$ 0.6, while the second classifier will provide y_2, with the same confidence level. Can we choose objectively between them? Prediction with the second classifier seems more "reliable" (in the intuitive sense of being more dependable). Then, if we had to choose, intuitively we would choose y_2 as class prediction, that is, we will prefer classification provided by the second classifier.

To formalize this intuition, since CL has proven unable to distinguish between the two predictions in the example, we will introduce a modification that can do it, and we name it the Modified Confidence Level (MCL). Then, MCL could also be used alternatively to CL in order to compare and choose among predictions made by different classifiers, and therefore to construct a combiner scheme that we name MCL-MV, which is like the already introduced CL-MV but substituting CL by MCL.

The Modified Confidence Level MCL is formally introduced as follows: consider a classifier that produces a r-dimensional vector $(p_1, \ldots, p_r)$ where p_k is the probability the classifier adjudges to class y_k ($p_k \geq 0$ for all $k = 1, \ldots, r$ and $\sum_{k=1}^{r} p_k = 1$). Then, the class predicted by the classifier is

$$y^* = y_\ell \text{ if } \ell = \arg\max_{k=1,\ldots,r} p_k\,, \text{ with confidence level CL} = \max_{k=1,\ldots,r} p_k\,.$$

Definition 1 In this setting, the *Modified Confidence Level* (MCL) is defined by:

$$\boxed{\text{MCL} = \text{CL} + (1 - \text{CL})\,\Delta} \tag{10}$$

where $\Delta = \text{CL} - \widetilde{\text{CL}}$ is the *margin of confidence*, being $\widetilde{\text{CL}} = \max_{k=1,\ldots,r\,:\,y_k \neq y^*} p_k$ the second maximum.

Justification: Note that $\text{CL} + (1 - \text{CL})\,\Delta = (1 - \Delta)\,\text{CL} + \Delta$. Formula (10) is obtained by searching for a weighted

sum of CL and Δ with respective weights $f(\Delta)$ and 1, that is, $\text{MCL} = f(\Delta)\,\text{CL} + \Delta$, such that

i) f is a linear function, that is, $f(\Delta) = a\,\Delta + b$,
ii) If $\text{CL} = \widetilde{\text{CL}}$ ($\Rightarrow \Delta = 0$), then MCL = CL,
iii) If $\text{CL} = 1$ ($\Rightarrow \Delta = 1$), then MCL = 1,
iv) $\text{CL} \leq \text{MCL} \leq 1$.

Indeed, by i) and ii), we obtain that $b = 1$, and then, by using iii) we have that $a = -1$, giving that $f(\Delta) = 1 - \Delta$. As by definition, $0 \leq \widetilde{\text{CL}} \leq \text{CL} \leq 1$, we have that $\Delta \in [0,\ 1]$ and then iv) holds. More specifically, MCL verifies:

$$0 < \frac{1}{r} \leq \text{CL} \leq \text{MCL} \leq 1$$

(see Appendix A for this and other properties of MCL).

Intuitively, we have introduced MCL as a modification of CL by adding a non-negative term Δ multiplied by 1−CL, and it can be interpreted as a *degree of support*, different from CL, to the prediction y^* given by the classifier. The idea is that this new measure incorporates more information than CL about how reliable the prediction is. This definition is motivated by the fact that when there are more than one class with a high probability (that is, when $\widetilde{\text{CL}}$ is close to CL), it seems reasonable to assume that we have less conviction when choosing the class with the highest probability, so to associate a *degree of support* with the prediction, we reward when it is done with a wide margin between the first and second candidates, and penalize otherwise.

For instance, a similar situation has already been considered in [17], where the second maximum is taken into account on the conditional probability distribution obtained from a Property Performance Bayesian Network, in order to select candidates that allow to find out the Virtual Machine with the minimal resource cost.

We can observe a parallelism between the MCL-MV combiner scheme and the *uncertainty sampling* method in the Active Learning setting (see [25]). Indeed, MCL-MV prefers the prediction of a base classifier which is less uncertain on how to label, uncertainty being measured through the margin of confidence. The higher the confidence margin with which the model predicts, the lower its uncertainty, then the more preferable the base classifier prediction will be.

Remark 3 Note that MCL coincides with CL if $\Delta = 0$, and also that if MCL is a monotonically non-decreasing function of CL, therefore, $y^*_{\text{MCL-MV}} = y^*_{\text{CL-MV}}$. By Proposition 4 below, we have then that CL-MV and MCL-MV schemes are equivalent combiners in the binary case, in the sense that they provide the same predictions.

Proposition 4 *In binary classification ($r = 2$), MCL is a monotonically increasing function of CL.*

Proof Increase in the binary case is consequence of the fact that MCL as function of x =CL is

$$f(x) = -2\,x^2 + 4\,x - 1$$

which is an increasing function of x in the interval where CL lives, [0.5, 1]. Indeed, if we denote CL by x, $\widetilde{\text{CL}}$ is then $1 - x$, and therefore

$$\begin{aligned}\text{MCL} &= \text{CL} + (1 - \text{CL})\,\Delta = \text{CL} + (1 - \text{CL})\,(\text{CL} - \widetilde{\text{CL}}) \\ &= x + (1 - x)\,(x - (1 - x)) = -2\,x^2 + 4\,x - 1,\end{aligned}$$

and $f'(x) = -4\,(x - 1) > 0$ if $x < 1$, which means that $f(x)$ is strictly increasing, while $f'(x) = 0$ if $x = 1$, case in which CL = MCL = 1. □

And what happens in the multi-class context? Although in the toy example of Table 2, CL-MV gives the same prediction as MCL-CV, as can be seen in Table 4 below, it doesn't have to be.

Indeed, in this example, $y^*_{\text{CL-MV}} = y^*_{\text{MCL-MV}} = y_2$, that match. However, this might not be the case, as we can see in this other toy example, from Table 5 in which, with respect to the toy example in Table 4, only the probability distribution corresponding to classifier $\mathcal{C}_4$ changes.

Table 4 Enlargement of the first toy example in Table 2

	Distribution			CL-MV			MCL-MV		
Classifier	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3
$\mathcal{C}_1$	**0.55**	0.00	0.45	0.55	0	0	0.5950	0	0
$\mathcal{C}_2$	**0.50**	0.05	0.45	0.50	0	0	0.5250	0	0
$\mathcal{C}_3$	**0.50**	0.02	0.48	0.50	0	0	0.5100	0	0
$\mathcal{C}_4$	0.00	**0.85**	0.15	0	0.85	0	0	0.9550	0
$\mathcal{C}_5$	0.05	**0.75**	0.20	0	0.75	0	0	0.8875	0
Sum				1.55	**1.60**	0	1.6300	**1.8425**	0

In boldface CL in the probability distributions, the maximum of the sum of weights both for the CL-MV and the MCL-MV combiners

Table 5 Second toy example

	Distribution			Majority vote			CL-MV			MCL-MV		
Classifier	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3	y_1	y_2	y_3
$\mathcal{C}_1$	**0.55**	0.00	0.45	1	0	0	0.55	0	0	0.5950	0	0
$\mathcal{C}_2$	**0.50**	0.05	0.45	1	0	0	0.50	0	0	0.5250	0	0
$\mathcal{C}_3$	**0.50**	0.02	0.48	1	0	0	0.50	0	0	0.5100	0	0
$\mathcal{C}_4$	0.15	**0.65**	0.20	0	1	0	0	0.65	0	0	0.8075	0
$\mathcal{C}_5$	0.05	**0.75**	0.20	0	1	0	0	0.75	0	0	0.8875	0
Sum	1.75	1.47	**1.78**	**3**	2	0	**1.55**	1.40	0	1.6300	**1.6950**	0
Average	0.350	0.294	**0.356**									
Product	0.00103125	0.00	**0.003888**									
Minimum	0.05	0.00	**0.20**									
Maximum	0.55	**0.75**	0.48									

In boldface CL in the probability distributions, the maximum of the *sum*, the *average*, the *product*, the *minimum* and the *maximum* of the predicted probabilities, and the maximum of the sum of weights both for the *majority vote*, the CL-MV and the MCL-MV combiners

In this second toy example, the predictions are:

$$y^*_{\text{Sum}} = y^*_{\text{Average}} = y^*_{\text{Product}} = y^*_{\text{Minimum}} = y_3,$$
$$y^*_{\text{Majority}} = y^*_{\text{CL-MV}} = y_1, \quad y^*_{\text{MCL-MV}} = y^*_{\text{Maximum}} = y_2,$$

and we can observe that CL-MV and MCL-MV provide different predictions.

6 Experimentation

It is worth mentioning here that "there is no single *best* classifier and that classifiers applied to different problems and trained by different algorithms perform differently" ([19]). For that, given the sources of variation which are imponderable when comparing classifiers, we follow the advice of [19], Section 1.4, and carry out the experiments with multiple training and validation sets, and with multiple runs, and perform statistical tests of hypotheses to compare Accuracy and MCC as behaviour metrics. More specifically, the experiments were carried out using different datasets to which we apply the *bagging* procedure, and we have validated and compared the obtained classifiers using the k-fold cross-validation procedure. Each experiment is repeated $N = 10$ times with different random seeds for the separation into the k folds of any dataset.

As decision trees are considered to be easy to implement state-of-the-art classifiers, their use is widely spread, and the cluster point of the methodology we introduce here does not lie in the type of base classifier used to ensemble, we decided to bag *decision trees* (through the C4.5 algorithm, using the function **J48** provided by the R library *RWeka*[1]). Note that in order to evaluate the predictive capacity of the ensembles, it is desirable that the base classifiers from which they are built have a not very high predictive power, which is achieved with *decision trees*.

First, we compare CL-MV and MCL-MV against the simple *majority vote*, and among them. Secondly, we compare CL-MV and MCL-MV against the usual *average* which has better behaviour than the *majority vote* and is computationally more demanding, and finally we compare them against the *decision tree* generated from the whole training dataset without *bagging*.

6.1 The datasets

Some datasets from small to moderate size have been considered in the experiments of this section, without claiming to be exhaustive. All the datasets are public and have been obtained from repositories of free access and proven prestige, such as the UCI machine learning repository [13] and Kaggle Inc[2]. See Table 6 for a brief list of the datasets and a summary of their main characteristics.

6.2 Experimental design

The experiments have been implemented by using programming with R language [22], with fixed seed for reproducibility purposes. The details are as follows: to avoid possible biases, the procedure is repeated $N = 10$ runs, with different seeds in the process of randomly splitting the dataset D into folds, and in any run the same strategy was used, which we break down into the following steps:

[1] https://CRAN.R-project.org/package=RWeka

[2] http://www.kaggle.com

Table 6 datasets used in the experimentation phase

Name	Source	# Cases	# Attr.	Marginal class distribution
Balance scale	UCI	625	4	0.46, 0.46, 0.08
Contraceptive method	UCI	1473	9	0.43, 0.35, 0.22
Horse colic	UCI	366	26	0.62, 0.24, 0.14
Car evaluation	UCI	1728	6	0.70, 0.22, 0.04, 0.04
Crime propensity [4]	Kaggle	463	23	0.30, 0.32, 0.38
Waveform	UCI	5000	21	0.33, 0.33, 0.33
Students alcohol [8]	Kaggle	1044	26	0.38, 0.23, 0.19, 0.13, 0.07
Students' API [2, 3]	Kaggle	480	16	0.44, 0.30, 0.26
Cardiotocography	UCI	2126	22	0.78, 0.14, 0.08
Yeast	UCI	1483	8	0.312, 0.003, 0.024, 0.030, 0.034, 0.110, 0.164, 0.289, 0.014, 0.020
Image segmentation	UCI	2310	19	0.143, 0.143, 0.143, 0.143, 0.143, 0.143, 0.143
Red wine quality	UCI	1599	11	0.01, 0.03, 0.43, 0.40, 0.12, 0.01
Ecoli	UCI	336	7	0.43, 0.23, 0.15, 0.10, 0.06, 0.02, 0.01, 0.01
Website phishing [1]	UCI	1353	9	0.52, 0.08, 0.40
Vehicle silhouettes	Kaggle	846	18	0.26, 0.25, 0.26, 0.24

Step 1: preparation of the training/validation sets.

For all the datasets we perform 10-fold cross-validation to evaluate the considered ensemble methods for *bagging*, as well as the decision tree algorithm C4.5 generated from the complete training dataset, denoted by DT in what follows. For that, we split the whole dataset D into 10 subsets or folds of length approximately equal, say $V_1, \ldots, V_{10}$. For each $\ell = 1, \ldots, 10$, let denote by T_ℓ the complementary of V_ℓ in D.

Step 2: learning DT.

For each $\ell = 1, \ldots, 10$, we use fold V_ℓ as validation set for the DT model constructed from T_ℓ as training set, from which we learn the decision tree algorithm C4.5. Then, for each ℓ we obtain the corresponding confusion matrix.

Step 3: bagging.

For each fold $\ell = 1, \ldots, 10$, and fixed a number of bags NBags (we have considered three possibilities: NBags = 5, 10, 100), we first choose from T_ℓ as many samples drawn at random with replacement, of length the number of cases in T_ℓ, as NBags, and denote them by $TB_1, \ldots, TB_{\text{NBags}}$. Then, for each $j = 1, \ldots, \text{NBags}$, we learn the decision tree algorithm C4.5 with TB_j as training dataset, and from them the ensembles using the combiner schemes: simple *majority vote*, CL-MV, MCL-MV and *average*. Note that all of them have been built using the same base classifiers. *Maximum*, *minimum* and *product* combiners have also been considered although they show a very poor behaviour. Since the other four combiner schemes (*majority vote*, CL-MV, MCL-MV and *average*) clearly outweigh them in all the settings with the two considered metrics (accuracy and MCC), we have not included the details to lighten the section devoted to the Results (Section 6.3).

Further step: Validation For each run $n = 1, \ldots, N$, and for each fold $\ell = 1, \ldots, 10$, we validate the ensembles of classifiers used for bagging with the validation set V_ℓ, obtaining the corresponding confusion matrices. From these matrices, and the matrices obtained for the DT classifier, we compute Accuracy and MCC metrics. In addition, for each model and for $n = 1, \ldots, N$, we compute the averages over $\ell = 1, \ldots, 10$ of Accuracy and MCC. So, finally, we have a vector of length N for each model, metric and dataset, which forms a statistical sample of size N that we use for carry on the comparison between models using the appropriate statistical tests of hypotheses: pairwise Student's t-tests or Wilcoxon signed-rank tests [28], its non-parametric counterpart, in case of lack of normality (after applying the Shapiro-Wilk test for normality [23] to discriminate whether we should apply a parametric or non-parametric methodology).

6.3 Results

The outcomes of the comparisons are in Tables 13 and 14 (Appendix B), where only significant results have been recorded. As usual, superscript * denotes statistical significance at 5%, ** at 1% and *** at 1‰, and · denotes weak significance at 10%. The alternative hypothesis, which is accepted for small p-values, is that reported in the table; for example, "MCL-MV $>$ *majority*" indicates that *bagging* procedure with the MCL-MV scheme is better than that with the simple *majority vote* in the

Table 7 Comparison summary of the predictive power: MCL-MV and CL-MV vs. *majority vote*

NBags	MCL-MV > *major.*	MCL-MV < *major.*	p-value	CL-MV > *major.*	CL-MV < *major.*	p-value
100	Contraceptive	Cardiotoc.	0.15625	Cardiotoc.	Ecoli	0.15625
	Car eval.			Car eval.		
	Waveform			Waveform		
	Students alco.			Students alco.		
10	Balance scale		0.003906**	Balance scale		0.003906**
	Car eval.			Car eval.		
	Students alco.			Students alco.		
	Cardiotoc.			Cardiotoc.		
	Yeast			Yeast		
	Crime			Crime		
	Horse colic			Horse colic		
	Vehicle			Vehicle		
5	Balance scale		0.0078125**	Balance scale		0.0078125**
	Contraceptive			Contraceptive		
	Waveform			Waveform		
	Students alco.			Students alco.		
	Image seg.			Image seg.		
	Red wine			Red wine		
	Ecoli			Ecoli		

Significative p-values are in favour that MCL-MV and CL-MV > *majority vote*

sense that the mean (or median, as appropriate) of any of the metrics is statistically significantly greater (the p-value of the corresponding statistical test is < 0.1), the accompanying p-value being the measure of statistical significance.

To facilitate interpretation of the results, a summary of Tables 13 and 14 is given in separate Tables: 7, 8 and 9, where p-values are one-sided exact Binomial p-values, which are obtained as follows: for example, in Table 7 below, the comparison between MCL-MV and *majority vote* with NBags $=$ 100 shows 4 datasets in favor of the first, and 1 in favor of the second, that is, of the 5 datasets for which there are significant differences between MCL-MV and *majority vote*, 4 are in favor of MCL-MV, resulting in a one-sided exact p-value of $P(B(n = 5, p = 0.5) = 4) = \binom{5}{4} \times 0.5^5 = 0.15625$, what

Table 8 Comparison summary of the predictive power: MCL-MV and CL-MV vs. *average*

NBags	MCL-MV > *average*	MCL-MV < *average*	p-value	CL-MV > *average*	CL-MV < *average*	p-value
100	Balance scale	Car eval.	0.2734375	Balance scale	Car eval.	0.24609375
	Contraceptive	Students alco.		Contraceptive	Students alco.	
	Website phishing	Cardiotoc.		Waveform	Red wine	
		Red wine		Cardiotoc	Vehicle	
				Website phishing		
10	Students' API	Car eval.	0.234375	Cardiotoc.	Car eval.	0.375
	Website phishing	Students alco.		Website phishing	Students alco.	
		Image seg.				
		Vehicle				
5	Cardiotoc.	Car eval.	0.3125	Cardiotoc.	Car eval.	0.21875
	Crime	Waveform		Crime	Waveform	
	Website phishing	Students alco.		Website phishing	Students alco.	
					Yeast	
					Image seg.	

Table 9 Comparison summary of the predictive power: *average* vs. *majority vote*, and MCL-MV vs. CL-MV

NBags	*average* > *majority*	*average* < *majority*	p-value	MCL-MV > CL-MV	MCL-MV < CL-MV	p-value
100	Car eval. Students alco. Red wine Horse colic	Balance scale Website phishing	0.234375	Balance Ecoli	Car eval. Students alco. Cardiotoc.	0.3125
10	Balance scale Car eval. Students alco. Yeast Ecoli Crime Vehicle		0.0078125**		Car eval. Students alco. Image seg. Vehicle	0.0625·
5	Balance scale Contraceptive Car eval. Waveform Students alco. Yeast Image seg. Red wine Ecoli	Cardiotoc. Website phishing	0.026855*	Balance Yeast Image seg. Ecoli	Car eval. Students' API	0.234375

Significative p-values are in favour that *average* > *majority vote*, and (slightly) in favour that CL-MV > MCL-MV

would be the probability of obtaining such a result if there were really no differences between the two combiner schemes.

We observe that the main hypothesis (that CL-MV and MCL-MV outperform the *majority vote*) is supported by results in Table 7, while Table 9 gives support to secondary hypothesis that in some cases MCL-MV shows better predictive power than CL-MV, while in others, the opposite, and Tables 8 and 9 sustain the secondary hypothesis that although the *average* rule outperforms the *majority vote*, it does not outperform CL-MV or MCL-MV.

Note that results corresponding to the comparison against DT are not recorded since for most of the datasets considered in this work, *bagging* with both combiner schemes CL-MV, MCL-MV, *majority vote* or *average*, are significantly better than DT, as expected.

Also note that we have used 5, 10 and 100 bootstrap replicates of the training datasets for *bagging* because it seemed reasonable to try a low number, a high number, and a middle number. We observe that for most data bases, increasing this number improves the predictive behavior of the classifiers obtained by *bagging* (see Table 16 in Appendix B, where we record the average over the runs of the averages over the folds for both Accuracy and MCC metrics), and also that the differences between them decrease.

Finally, to address the issue of the computational complexity, measured by the amount of time required to run the algorithm, we have measured running times of R code for all performed computations corresponding to *bagging* with the different combiner schemes (see Table 15 in Appendix B, whose information is summarized in

Table 10 Comparison summary of the running times: *average* vs. *majority vote*

NBags	*average* > *majority vote*	*average* < *majority vote*	p-value
100	14	0	6.10352×10^{-5}***
10	14	0	6.10352×10^{-5}***
5	13	0	1.22070×10^{-4}***

Significative p-values are in favour that times of *average* are greater than that of *majority vote*

Table 11 Comparison summary of the running times: MCL-MV and CL-MV vs. *average*

NBags	*aver.* > MCL-MV	*aver.* < MCL-MV	p-value	*aver.* > CL-MV	*aver.* < CL-MV	p-value
100	8	0	0.003906^{**}	10	2	0.016113^{*}
10	12	0	$2.44 \times 10^{-4***}$	14	0	$6.104 \times 10^{-5***}$
5	13	0	$1.221 \times 10^{-4***}$	13	0	$1.221 \times 10^{-4***}$

Significative p-values are in favour that times of *average* are greater than that of MCL-MV and that of CL-MV

Tables 10, 11 and 12 below where, as for Tables 7-9, p-values are one-sided exact Binomial p-values). For that, we use functions `tic` and `toc` of the library *tictoc*[3] to measure the run time of a chunk of code by taking the difference between the time at the start and at the end (elapsed times).

The results of these tables indicate that in terms of running times, *bagging* with the simple *majority vote* is less computationally demanding than with CL-MV or MCL-MV, which in turn is less computationally demanding than with the *average* combiner scheme, confirming the remaining part of the secondary hypotheses.

7 Discussion

"No free lunch" theorems (see [29]) tell us that for any algorithm, its performance being high in one context will be compensated for being low in a different one. Applied to ensembles of probabilistic classifiers, different combiner schemes are known to perform differently on different datasets, and therefore choosing an appropriate combiner scheme in each particular case is a major problem. To address this question, it is essential to understand the way in which classifier ensembles are combined.

In this paper we have used a common theoretical framework to ensemble multi-class probabilistic classifiers (see (1)), of which the most popular combiner schemes are a particular case. We have focused on the consideration of class-conscious combiner schemes, for which function g_k is defined from probabilities assigned by the different base classifiers to class k, $\{p_{jk},\ j = 1, \ldots, M\}$, and/or the predictions of these classifiers, $\{y_j^*,\ j = 1, \ldots, M\}$. Out of our scope are the class-indifferent combiners, such as the *decision templates* (see [19]), which use the set of all the probabilities $\{p_{j\ell},\ j = 1, \ldots, M,\ \ell = 1, \ldots, r\}$ (named there as "decision profile" DP) to define g_k. A difference between the former and the latter is that the class-conscious combiners are idempotent, that is, applied to M copies of the same classifier, give the same decision than that of the classifier; however, this is not the case for *decision templates*, which can give a different decision. This characteristic of *decision templates*, whose predictive capacity may be better or worse than that of a class-conscious combiner, is difficult to justify from the point of view of intuition and applications: In a team whose members fully agree and make the same decision, how to explain that the team as a whole makes a different decision than its members?

As many empirical studies have shown that simpler class-conscious combiner schemes often work remarkably well ([18]), we concentrate on this category and only consider non-trainable rules, which do not need a separate training algorithm to find weights for the base classifiers, usually as a function of their estimated accuracies. We distinguish between hard and soft voting. *Majority vote* is a hard-voting scheme (based in the label outputs of the classifiers), perhaps the oldest and simpler strategy for decision making in a group, which is still the most common today.

From among the soft-voting combiner schemes polling the continuous outputs of the base classifiers to reach a decision, we consider the most popular choices: *average*, *product*, *minimum* and *maximum*. The *minimum* is the most pessimistic choice: a class is supported by the ensemble with a certain degree if all the classifiers members of the ensemble give a support of at least as much as this degree to that class, while at the other extreme, the *maximum* rule is the most optimistic, since a support degree is assigned by the ensemble to a class if at least one of the members supports that class with this degree; the *average* is an intermediate case between pessimism and optimism, and in general it is preferred. Besides, the *product* and the *minimum* combiners are oversensitive to probabilities close to zero: the presence of such probabilities for a given class has the effect of veto on that particular class regardless of how large the probabilities assigned by the other classifiers to this class might be, which, in general, prevents against their use.

The introduction in this paper of a novel **class-conscious non-trainable** combiner scheme named CL-MV (and its counterpart MCL-MV), which is halfway between the *majority vote* and the *average* schemes, defined as a weighted version of the former by using the confidence levels as weights, seemed natural and convenient, since non-trainable rules are simpler, well behaved and have lower computing needs. Indeed, with its simple definition, this combiner achieves greater accuracy than the *majority vote* in

[3] https://CRAN.R-project.org/package=tictoc

Table 12 Comparison summary of the running times: MCL-MV and CL-MV vs. *majority*

NBags	MCL-MV > *maj.*	MCL-MV < *maj.*	p-value	CL-MV > *maj.*	CL-MV < *maj.*	p-value
100	14	0	$6.104 \times 10^{-5***}$	14	0	$6.104 \times 10^{-5***}$
10	8	1	0.03125^{*}	8	0	0.003906^{**}
5	2	3	0.3125	3	3	0.3125

Significative p-values are in favour that times of *majority vote* are less than that of MCL-MV and that of CL-MV

the binary case, and at the same time, in plausible scenarios, it is more resilient to probability estimation errors than both the *average* and the *product* schemes, hence it is preferable when probabilities assigned by classifiers to the classes are not computed correctly but they suffer from an estimation error.

Furthermore, comparing those combiner schemes through experimentation, we show by bagging that both CL-MV and MCL-MV are competitive with the *average*, being less computationally demanding, and outperform the rest of considered combiners, including the *majority vote*, so they are a good alternative to consider when making the choice of a combiner scheme.

8 Conclusions

The *majority vote* is an elementary combiner scheme that together with the *average*, it is still the most used in practice today to ensemble probabilistic classifiers. In this work we have introduced a non-trainable weighted version of the simple *majority vote* combiner scheme, CL-MV, that uses the confidence level that each base classifier gives to its prediction as weight (instead of use weights based on the accuracies of the base classifiers, what corresponds to a trainable weighted majority vote scheme), and can be thought as a *semi-hard voting* combiner scheme, halfway between the *majority vote* (hard voting) and the *average* (soft voting) combiners. From a theoretical point of view, we have proved the following results, which could be a plausible explanation of its good performance observed in the experimentation phase:

- In the binary case, CL-MV is more accurate than the *majority vote* scheme.
- In the multi-class setting, under reasonable hypotheses, both the *average* rule and the CL-MV are more resilient to estimation errors than the *product* rule, being CL-MV even more resilient than the *average* rule.

That is, CL-MV is a *semi-hard voting* rule that improves the accuracy of the *hard voting* and the resilience to estimation errors of the considered *soft voting* combiner schemes.

We also introduce another simple measure of the *degree of support* that each base classifier gives to its prediction, alternative to CL in the multi-class setting, and we name it MCL (by Modified Confidence Level), which embodies more information than the usual CL since it incorporates some knowledge about the information involved in the probability distribution over the classes. More specifically, MCL is based on the difference between the maximum and the second maximum of the probability distribution.

The results of our experiments, using fifteen datasets from the UCI machine learning repository, are very encouraging since with the meta-algorithm *bagging*, we have given heuristic support to the hypotheses that we had raised at the beginning of the work, reaching the following statements, with both *accuracy* and MCC as performance metrics:

- MCL-MV and CL-MV give similar performance results, and each of them is better for some of the databases used in our experimental work.
- Both MCL-MV and CL-MV improve the classifying power of the simple *majority vote*.
- With both MCL-MV and CL-MV, *bagging* outperforms the base classifiers (each a single decision tree algorithm C4.5).
- *Bagging* with the *average* rule outperforms that with the simple *majority vote*, but it is equivalent to use MCL-MV or CL-MV.

Moreover, we also evaluate the computational complexity of the algorithms, reaching the conclusion that time complexity is higher for *bagging* using the *average*, and lower for the *majority vote*, positioning MCL-MV and CL-MV combiners at an intermediate point.

Therefore, we can say from the experimental evidence that both CL-MV and MCL-MV are combiner schemes preferable for *bagging* to the simple *majority vote* (the usual) and also to the *average* (its natural competitor), conclusion that is in line with the theoretical results that have been proved.

Appendix A: Properties of MCL

Proposition 5 *For $r \geq 2$,*

a)

$$\frac{1}{r} \leq \mathrm{CL} \leq \mathrm{MCL} \leq \mathrm{CL} + (1 - \mathrm{CL})\,\frac{r\,\mathrm{CL} - 1}{r - 1} \leq 1\,.$$

b) *If $r > 2$, fixed $\widetilde{\mathrm{CL}}$,* MCL *is an increasing function of* CL*, while fixed* CL *it is a straight line of non-positive slope as function of $\widetilde{\mathrm{CL}}$, then decreasing. Moreover, if $\widetilde{\mathrm{CL}} \leq 1/r$,*

$$\mathrm{CL} \leq \mathrm{CL} + (1 - \mathrm{CL})\,\frac{r\,\mathrm{CL} - 1}{r} \leq \mathrm{MCL}$$

Proof a) The first inequality is evident by definition of CL. Second inequality is obvious, being strict except if CL$=1$ or $\widetilde{\mathrm{CL}}$ =CL. Third inequality is due to the fact that

$$\widetilde{\mathrm{CL}} \geq \frac{1 - \mathrm{CL}}{r - 1}\,,$$

which is obvious by definition of $\widetilde{\mathrm{CL}}$, since a total probability of $1 - \mathrm{CL}$ is divided among the $r - 1$ non-maximum values, from which $\widetilde{\mathrm{CL}}$ is defined, in turn, as the maximum.

Finally, we prove that $\mathrm{CL} + (1 - \mathrm{CL})\,\frac{r\,\mathrm{CL}-1}{r-1} \leq 1$. Indeed, simple algebraic manipulations show that this inequality is equivalent to $(1 - \mathrm{CL})^2 \geq 0$, what is obviously fulfilled. This inequality is strict except if CL$=1$.

b) Fixed $\widetilde{\mathrm{CL}}$, MCL as function of $x = \mathrm{CL}$ is $g(x) = -x^2 + (2 + \widetilde{\mathrm{CL}})\,x - \widetilde{\mathrm{CL}}$, which is strictly increasing since its first derivative is $g'(x) = -2\,x + (2 + \widetilde{\mathrm{CL}})$, which is > 0 for $0 < x \leq 1$. On the other hand, fixed CL, as function of $z = \widetilde{\mathrm{CL}}$ MCL is $h(z) = -(1 - \mathrm{CL})\,z + \mathrm{CL} + (1 - \mathrm{CL})\,\mathrm{CL}$. □

Corollary 1 *For $r \geq 2$,*

$$\mathrm{CL} = 1 \Longleftrightarrow \mathrm{MCL} = 1$$

Proof By definition of MCL, if CL$=1$ then MCL=CL$=1$.

The reverse implication is also true. Indeed, MCL$=1$ implies by Proposition 5 a) that

$$\mathrm{CL} + (1 - \mathrm{CL})\,\frac{r\,\mathrm{CL} - 1}{r - 1} = 1\,,$$

which is equivalent to $(1-\mathrm{CL})^2 = 0$, implying $\mathrm{CL} = 1$. □

Appendix B: Complementary tables

Table 13 Comparison between MCL-MV, CL-MV, *majority vote* and *average* combiner schemes, with the different choices for the number of bags in *bagging*, attending to Accuracy and MCC, for different datasets

dataset		NBags		
		100	10	5
Balance scale	Acc.	MCL-MV > CL-MV (0.0046^{**}) MCL-MV > *average* (0.0054^{**}) CL-MV > *average* ($0.063^{\cdot}$)		MCL-MV > CL-MV ($0.088^{\cdot}$)
			MCL-MV > *majority* (0.0025^{**}) CL-MV > *majority* (0.00012^{***})	MCL-MV > *majority* (0.011^{*}) CL-MV > *majority* (0.0049^{**})
		majority > *average* ($0.063^{\cdot}$)	*average* > *majority* (0.0039^{**})	*average* > *majority* (0.019^{*})
	MCC	MCL-MV > CL-MV (0.0029^{**}) MCL-MV > *average* (0.002^{**}) CL-MV > *average* (0.042^{*})		
			MCL-MV > *majority* (0.0032^{**}) CL-MV > *majority* (0.00017^{***})	MCL-MV > *majority* (0.0098^{**}) CL-MV > *majority* (0.0068^{**})
		majority > *average* ($0.053^{\cdot}$)	*average* > *majority* (0.0039^{**})	*average* > *majority* (0.024^{*})
Contraceptive	Acc.	MCL-MV > *average* (0.046^{*}) CL-MV > *average* (0.049^{*}) MCL-MV > *majority* ($0.069^{\cdot}$)		MCL-MV > *majority* ($0.051^{\cdot}$) CL-MV > *majority* ($0.097^{\cdot}$)

Table 13 (continued)

dataset		NBags		
		100	10	5
	MCC	MCL-MV > *average* (0.046*) CL-MV > *average* (0.049*) MCL-MV > *majority* (0.069·)		MCL-MV > *majority* (0.017*) CL-MV > *majority* (0.039*) *average* > *majority* (0.063·)
Horse colic	Acc.		MCL-MV > *majority* (0.064·) CL-MV > *majority* (0.052·)	
	MCC	*average* > *majority* (0.091·)	MCL-MV > *majority* (0.066·) CL-MV > *majority* (0.057·)	
Car eval.	Acc.	CL-MV > MCL-MV (0.071·) *average* > MCL-MV (0.0016**) *average* > CL-MV (0.0056**) MCL-MV > *majority* (0.067·) CL-MV > *majority* (0.0095**) *average* > *majority* (0.00054***)	CL-MV > MCL-MV (0.049*) *average* > MCL-MV (0.00084***) *average* > CL-MV (0.004**) MCL-MV > *majority* (0.0053**) CL-MV > *majority* (0.00021***) *average* > *majority* (7.7×10^{-6}***)	CL-MV > MCL-MV (0.096·) *average* > MCL-MV (0.00069***) *average* > CL-MV (0.00019***) *average* > *majority* (0.0027**)
	MCC	CL-MV > MCL-MV (0.078·) *average* > MCL-MV (0.002**) *average* > CL-MV (0.0098**) CL-MV > *majority* (0.019*) *average* > *majority* (0.0049**)	CL-MV > MCL-MV (0.065·) *average* > MCL-MV (0.002**) *average* > CL-MV (0.014*) MCL-MV > *majority* (0.019*) CL-MV > *majority* (0.00098***) *average* > *majority* (0.00098***)	CL-MV > MCL-MV (0.077·) *average* > MCL-MV (0.00087***) *average* > CL-MV (0.00032***) *average* > *majority* (0.0031**)
Crime	Acc.		MCL-MV > *majority* (0.04*) CL-MV > *majority* (0.04*) *average* > *majority* (0.044*)	MCL-MV > *average* > (0.053·) CL-MV > *average* (0.053·)
	MCC		MCL-MV > *majority* (0.035*) CL-MV > *majority* (0.035*) *average* > *majority* (0.038*)	MCL-MV > *average* > (0.057·) CL-MV > *average* (0.057·)
Waveform	Acc.	CL-MV > *average* (0.096·) MCL-MV > *majority* (0.0094**) CL-MV > *majority* (0.093·)		*average* > MCL-MV (0.0065**) *average* > CL-MV (0.026*) MCL-MV > *majority* (0.046*) CL-MV > *majority* (0.052·) *average* > *majority* (0.0012**)
	MCC	CL-MV > *average* (0.096·) MCL-MV > *majority* (0.0098**) CL-MV > *majority* (0.093·)		*average* > MCL-MV (0.0063**) *average* > CL-MV (0.025*) MCL-MV > *majority* (0.044*) CL-MV > *majority* (0.05·) *average* > *majority* (0.0011**)
Students alco.	Acc.	CL-MV > MCL-MV (0.00014***) *average* > MCL-MV (9.2×10^{-5}***)	CL-MV > MCL-MV (0.00019***) *average* > MCL-MV (2.0×10^{-5}***)	*average* > MCL-MV (0.0015**)

Table 13 (continued)

dataset		NBags		
		100	10	5
		average > CL-MV (0.00026***) MCL-MV > *majority* (0.0059**) CL-MV > *majority* (9.5×10^{-5}***) *average* > *majority* (7.6×10^{-5}***)	*average* > CL-MV (0.00022***) MCL-MV > *majority* (1.6×10^{-5}***) CL-MV > *majority* (3.1×10^{-7}***) *average* > *majority* (2.0×10^{-7}***)	*average* > CL-MV (0.0019**) MCL-MV > *majority* (0.00065***) CL-MV > *majority* (0.00027***) *average* > *majority* (2.5×10^{-5}***)
	MCC	CL-MV > MCL-MV (0.00018***) *average* > MCL-MV (0.00012****) *average* > CL-MV (0.00034***) MCL-MV > *majority* (0.0072**) CL-MV > *majority* (0.00011***) *average* > *majority* (9.1×10^{-5}***)	CL-MV > MCL-MV (0.00013***) *average* > MCL-MV (2.6×10^{-5}****) *average* > CL-MV (0.00027***) MCL-MV > *majority* (1.5×10^{-5}***) CL-MV > *majority* (3.2×10^{-7}***) *average* > *majority* (2.8×10^{-7}***)	*average* > MCL-MV (0.00074***) *average* > CL-MV (0.0012**) MCL-MV > *majority* (0.00056***) CL-MV > *majority* (0.00022***) *average* > *majority* (2.1×10^{-5}***)
Students' API	Acc.		MCL-MV > *average* (0.033*)	CL-MV > MCL-MV (0.074·)
	MCC		MCL-MV > *average* (0.022*)	CL-MV > MCL-MV (0.05·)
Cardiotoc.	Acc.	CL-MV > MCL-MV (0.0029**) *average* > MCL-MV (0.062·) CL-MV > *average* (0.00098***) *majority* > MCL-MV (0.029*) CL-MV > *majority* (0.00098***)	CL-MV > *average* (0.084·) MCL-MV > *majority* (0.053·) CL-MV > *majority* (0.032*)	MCL-MV > *average* (0.026*) CL-MV > *average* (0.0052**) *majority* > *average* (0.037*)
	MCC	CL-MV > MCL-MV (7.9×10^{-5}***) *average* > MCL-MV (0.07·) CL-MV > *average* (7.6×10^{-5}***) *majority* > MCL-MV (0.029*) CL-MV > *majority* (5.7×10^{-5}***)	MCL-MV > *majority* (0.064·) CL-MV > *majority* (0.04*)	MCL-MV > *average* (0.021*)

Only statistically significant differences have been recorded, with the corresponding one-sided p-value

Table 14 Continuation of Table 13

dataset		NBags		
		100	10	5
Yeast	Acc.			MCL-MV > CL-MV (0.097·) *average* > CL-MV (0.093·)
			MCL-MV > *majority* (0.0017**) CL-MV > *majority* (0.0078**) *average* > *majority* (0.0037**)	*average* > *majority* (0.0082**)
	MCC			*average* > CL-MV (0.098·)
			MCL-MV > *majority* (0.0019**) CL-MV > *majority* (0.0084**) *average* > *majority* (0.0041**)	*average* > *majority* (0.097·)
Image seg.	Acc.		CL-MV > MCL-MV (0.018*) *average* > MCL-MV (0.084·)	MCL-MV > CL-MV (0.036*)

Table 14 (continued)

dataset		NBags		
		100	10	5
				average > CL-MV (0.068$^{\cdot}$) MCL-MV > *majority* (0.029*) CL-MV > *majority* (0.048*) *average* > *majority* (0.016*)
	MCC		CL-MV > MCL-MV (0.019*) *average* > MCL-MV (0.088$^{\cdot}$)	MCL-MV > CL-MV (0.03*) MCL-MV > *majority* (0.032*) CL-MV > *majority* (0.042*) *average* > *majority* (0.032*)
Red wine	Acc.	*average* > MCL-MV (0.04*) *average* > CL-MV (0.028*) *average* > *majority* (0.039*)		MCL-MV > *majority* (0.04*) CL-MV > *majority* (0.053$^{\cdot}$) *average* > *majority* (0.023*)
	MCC	*average* > MCL-MV (0.035*) *average* > CL-MV (0.024*) *average* > *majority* (0.036*)		MCL-MV > *majority* (0.046*) CL-MV > *majority* (0.058$^{\cdot}$) *average* > *majority* (0.027*)
Ecoli	Acc.	MCL-MV > CL-MV (0.084$^{\cdot}$) *majority* > CL-MV (0.078$^{\cdot}$)		MCL-MV > CL-MV (0.099$^{\cdot}$) MCL-MV > *majority* (0.025*) CL-MV > *majority* (0.0089**) *average* > *majority* (0.016*)
	MCC		*average* > *majority* (0.097$^{\cdot}$)	MCL-MV > *majority* (0.013*) CL-MV > *majority* (0.0048**) *average* > *majority* (0.012*)
Website	Acc.	MCL-MV > *average* (0.03*) CL-MV > *average* (0.068$^{\cdot}$) *majority* > *average* (0.029*)	MCL-MV > *average* (0.097$^{\cdot}$) CL-MV > *average* (0.007**)	MCL-MV > *average* (0.021*) CL-MV > *average* (0.024*) *majority* > *average* (0.028*)
	MCC	MCL-MV > *average* (0.049*) CL-MV > *average* (0.062$^{\cdot}$) *majority* > *average* (0.062$^{\cdot}$)	MCL-MV > *average* (0.093$^{\cdot}$) CL-MV > *average* (0.0087**)	MCL-MV > *average* (0.024*) CL-MV > *average* (0.027*) *majority* > *average* (0.03*)
Vehicle	Acc.		CL-MV > MCL-MV (0.036*) *average* > MCL-MV(0.075$^{\cdot}$) MCL-MV > *majority* (0.08$^{\cdot}$) CL-MV > *majority* (0.041*) *average* > *majority* (0.041*)	
	MCC	*average* > CL-MV(0.097$^{\cdot}$)	CL-MV > MCL-MV (0.014*) MCL-MV > *majority* (0.08$^{\cdot}$) CL-MV > *majority* (0.042*) *average* > *majority* (0.042*)	

Table 15 Comparison between MCL-MV, CL-MV, *majority vote* and *average* combiner schemes, with the different choices for the number of bags in *bagging*, attending to the mean running times

dataset	NBags		
	100	10	5
Balance scale	MCL-MV < CL-MV ($0.097^{\cdot}$)		
	MCL-MV < *average* (0.002^{**})	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.0029^{**})
		CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.019^{*})
	majority < MCL-MV(0.00098^{***})		*majority* < MCL-MV ($0.053^{\cdot}$)
	majority < CL-MV(0.00098^{***})		
	majority < *average* (0.00098^{***})	*majority* < *average* (0.0029^{**})	*majority* < *average* (0.00098^{***})
Contraceptive	CL-MV < MCL-MV ($0.053^{\cdot}$)		
	MCL-MV < *average* ($0.065^{\cdot}$)	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.00098^{***})
	CL-MV < *average* ($0.024^{\cdot}$)	CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.00098^{***})
	majority < MCL-MV(0.0029^{**})	*majority* < MCL-MV (0.041^{*})	
	majority < CL-MV(0.002^{**})	*majority* < CL-MV ($0.097^{\cdot}$)	
	majority < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})
Horse colic			MCL-MV < *average* (0.0029^{**})
	CL-MV < *average* ($0.08^{\cdot}$)	CL-MV < *average* (0.033^{*})	CL-MV < *average* (0.0029^{**})
	majority < MCL-MV(0.042^{*})	*majority* < MCL-MV ($0.053^{\cdot}$)	
	majority < CL-MV(0.019^{*})	*majority* < CL-MV (0.028^{*})	
	majority < *average* (0.0072^{**})	*majority* < *average* (0.0029^{**})	*majority* < *average* (0.0029^{**})
Car eval.	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.0029^{**})
	CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.0029^{**})
	majority < MCL-MV(0.00098^{***})	*majority* < MCL-MV (0.024^{*})	
	majority < CL-MV(0.00098^{***})	*majority* < CL-MV (0.032^{*})	*majority* < CL-MV ($0.062^{\cdot}$)
	majority < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})
Crime			MCL-MV < CL-MV (0.021^{*})
		MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.0029^{**})
		CL-MV < *average* (0.0029^{**})	CL-MV < *average* (0.0046^{**})
	majority < MCL-MV(0.0029^{**})		
	majority < CL-MV(0.0029^{**})		
	majority < *average* (0.00098^{***})	*majority* < *average* (0.0029^{**})	*majority* < *average* (0.0029^{**})
Waveform	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.00098^{***})
	CL-MV < *average* (0.042^{*})	CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.00098^{***})
	majority < MCL-MV(0.042^{*})	*majority* < MCL-MV (0.00098^{***})	
	majority < CL-MV(0.026^{*})	*majority* < CL-MV (0.0029^{**})	*majority* < CL-MV (0.026^{*})
	majority < *average* (0.002^{**})	*majority* < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})
Students alco.	MCL-MV < *average* ($0.065^{\cdot}$)	MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.0046^{**})
	CL-MV < *average* ($0.065^{\cdot}$)	CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.00098^{***})
	majority < MCL-MV(0.00098^{***})	*majority* < MCL-MV ($0.084^{\cdot}$)	
	majority < CL-MV(0.002^{**})	*majority* < CL-MV ($0.07^{\cdot}$)	
	majority < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})	*majority* < *average* (0.00098^{***})
Students' API		CL-MV < MCL-MV (0.021^{*})	
		MCL-MV < *average* (0.00098^{***})	MCL-MV < *average* (0.0045^{**})
		CL-MV < *average* (0.00098^{***})	CL-MV < *average* (0.004^{**})

Table 15 (continued)

dataset	NBags		
	100	10	5
	majority < MCL-MV(0.0098**)	*majority* < MCL-MV (0.053·)	
	majority < CL-MV(0.0068**)	*majority* < CL-MV (0.062·)	
	majority < *average* (0.002**)	*majority* < *average* (0.0029**)	*majority* < *average* (0.0029**)
Cardiotoc.	*average* < CL-MV (0.096·)	CL-MV < *average* (0.042*)	
	majority < MCL-MV(0.00098***)		
	majority < CL-MV(0.0029**)		
	majority < *average* (0.00098***)	*majority* < *average* (0.0068**)	
Yeast	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)
	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)
	majority < MCL-MV(0.0029**)	MCL-MV < *majority* (0.016*)	MCL-MV < *majority* (0.084·)
	majority < CL-MV(0.00098***)		CL-MV < *majority* (0.0049**)
	majority < *average* (0.0029**)	*majority* < *average* (0.00098***)	*majority* < *average* (0.00098***)
Image seg.		MCL-MV < CL-MV (0.038*)	
	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)
	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)
	majority < MCL-MV(0.00098***)		MCL-MV < *majority* (0.057·)
	majority < CL-MV(0.0049**)		CL-MV < *majority* (0.054·)
	majority < *average* (0.00098***)	*majority* < *average* (0.00098***)	*majority* < *average* (0.00098***)
Red wine		MCL-MV < CL-MV (0.032*)	
	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)
	CL-MV < *average* (0.0029**)	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)
	majority < MCL-MV(0.0049**)		
	majority < CL-MV(0.00098***)		CL-MV < *majority* (0.049*)
	majority < *average* (0.0029**)	*majority* < *average* (0.00098***)	*majority* < *average* (0.00098***)
Ecoli	*average* < CL-MV (0.065·)		
			MCL-MV < *majority* (0.062·)
	majority < CL-MV(0.065·)	*majority* < CL-MV(0.062·)	
Website		MCL-MV < CL-MV (0.062·)	CL-MV < MCL-MV (0.046*)
		MCL-MV < *average* (0.00098***)	MCL-MV < *average* (0.00098***)
	CL-MV < *average* (0.097·)	CL-MV < *average* (0.002**)	CL-MV < *average* (0.004**)
	majority < MCL-MV(0.002**)	*majority* < MCL-MV(0.0029**)	*majority* < MCL-MV(0.026*)
		majority < CL-MV(0.002**)	*majority* < CL-MV(0.038*)
	majority < *average* (0.00098***)	*majority* < *average* (0.00098***)	*majority* < *average* (0.00098***)
Vehicle		MCL-MV < *average* (0.0039**)	MCL-MV < *average* (0.00098***)
	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)	CL-MV < *average* (0.00098***)
	majority < MCL-MV(0.0029**)		
	majority < CL-MV(0.00098***)		
	majority < *average* (0.00098***)	*majority* < *average* (0.00098***)	*majority* < *average* (0.0029**)

Only statistically significant differences have been recorded, with the corresponding one-sided p-value

Table 16 Average over the runs of the averages over the folds, for the metrics Accuracy and MCC, with the different combiner ensembles used for *bagging* purpose, for all the datasets

dataset	Combiner	Accuracy			MCC		
		NBags=100	NBags=10	NBags=5	NBags=100	NBags=10	NBags=5
Balance scale	Average	0.7740154	0.7513288	0.7288445	0.5883788	0.5487738	0.5072420
	Majority vote	0.7780356	0.7440587	0.7201709	0.5965023	0.5363291	0.4935601
	CL-MV	0.7769186	0.7527805	0.7258931	0.5941668	0.5515474	0.5029486
	MCL-MV	0.7794752	0.7514901	0.7272075	0.5988784	0.5492955	0.5051069
Contraceptive	Average	0.5353973	0.5254163	0.5193388	0.2774085	0.2611231	0.2537991
	Majority vote	0.5359469	0.5255374	0.5165075	0.2782439	0.2616225	0.2485659
	CL-MV	0.5367578	0.5244531	0.5188639	0.2796619	0.2598304	0.2534177
	MCL-MV	0.5369673	0.5250639	0.5192748	0.2798230	0.2605874	0.2539472
Horse colic	Average	0.7175397	0.7028571	0.6853175	0.4604409	0.4324652	0.4074338
	Majority vote	0.7164683	0.6977778	0.6857937	0.4554709	0.4223913	0.4074721
	CL-MV	0.7167460	0.7033730	0.6876984	0.4581209	0.4337365	0.4113487
	MCL-MV	0.7164683	0.7033730	0.6874603	0.4578716	0.4338120	0.4112308
Car eval.	Average	0.9374716	0.9325426	0.9291912	0.8680035	0.8552277	0.8480259
	Majority vote	0.9340491	0.9284806	0.9246111	0.8591384	0.8470087	0.8382690
	CL-MV	0.9354974	0.9309884	0.9247300	0.8619095	0.8520468	0.8386320
	MCL-MV	0.9345724	0.9302274	0.9241460	0.8601390	0.8504922	0.8372627
Crime	Average	0.9665040	0.9667081	0.9653904	0.9496098	0.9499838	0.9479535
	Majority vote	0.9667214	0.9651863	0.9660293	0.9499501	0.9476783	0.9489973
	CL-MV	0.9665040	0.9664907	0.9662467	0.9496098	0.9496911	0.9492570
	MCL-MV	0.9665040	0.9664907	0.9662467	0.9496098	0.9496911	0.9492570
Waveform	Average	0.8364200	0.8206000	0.8041600	0.7550795	0.7311470	0.7064618
	Majority vote	0.8362600	0.8203400	0.8033400	0.7548421	0.7307600	0.7052240
	CL-MV	0.8365600	0.8205400	0.8037400	0.7552898	0.7310510	0.7058331
	MCL-MV	0.8366600	0.8207400	0.8037200	0.7554365	0.7313542	0.7058048
Students alco.	Average	0.735260	0.6791667	0.6195513	0.6429574	0.5653656	0.4838058
	Majority vote	0.7190954	0.6524501	0.5973291	0.6206685	0.5276119	0.4519976
	CL-MV	0.7278134	0.6695833	0.6121724	0.6328053	0.5520377	0.4731271
	MCL-MV	0.7216845	0.6652849	0.6118803	0.6243004	0.5458373	0.4725444
Students' API	Average	0.7750000	0.7660417	0.7500000	0.6533812	0.6385312	0.6150958
	Majority vote	0.7741667	0.7650000	0.7506250	0.6520287	0.6371588	0.6166304
	CL-MV	0.7739583	0.7670833	0.7500000	0.6516134	0.6400713	0.6158577
	MCL-MV	0.7733333	0.7677083	0.7493750	0.6508280	0.6410259	0.6149154
Cardiotoc.	Average	0.9327064	0.9410728	0.9374823	0.8107983	0.8352823	0.8268416
	Majority vote	0.9324273	0.9405016	0.9380011	0.8101460	0.8337865	0.8282892
	CL-MV	0.8518415	0.8518415	0.8518415	0.7964556	0.7964556	0.7964556
	MCL-MV	0.9320512	0.9412602	0.9378609	0.8090851	0.8356427	0.8279561
Yeast	Average	0.6186299	0.5989162	0.5752443	0.5054024	0.4795878	0.4490664
	Majority vote	0.6184325	0.5950783	0.5726848	0.5053746	0.4747339	0.4460183
	CL-MV	0.6186352	0.5980446	0.5734835	0.5055393	0.4787068	0.4468311
	MCL-MV	0.6184974	0.5985824	0.5743633	0.5054534	0.4793597	0.4478329

Table 16 (continued)

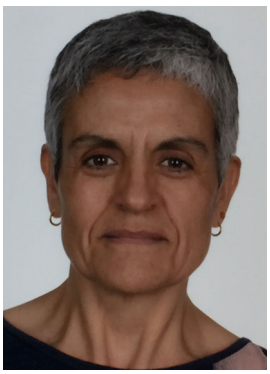
dataset	Combiner	Accuracy			MCC		
		NBags=100	NBags=10	NBags=5	NBags=100	NBags=10	NBags=5
Image seg.	Average	0.9699134	0.9669697	0.9642857	0.9649614	0.9615164	0.9584254
	Majority vote	0.9699567	0.9671429	0.9634199	0.9650143	0.9617162	0.9574192
	CL-MV	0.9698268	0.9669697	0.9640693	0.9648596	0.9615169	0.9581705
	MCL-MV	0.9699567	0.9667965	0.9642424	0.9650121	0.9613172	0.9583763
Red wine	Average	0.6932615	0.6681390	0.6473372	0.5071194	0.4689767	0.4367043
	Majority vote	0.6915173	0.6670339	0.6426797	0.5041135	0.4674161	0.4297496
	CL-MV	0.6918250	0.6675767	0.6465656	0.5046150	0.4681904	0.4357777
	MCL-MV	0.7917060	0.6676426	0.6467475	0.5043858	0.4682266	0.4359762
Ecoli	Average	0.8527040	0.8442424	0.8342657	0.7977340	0.7857791	0.7718838
	Majority vote	0.8527972	0.8403963	0.8274359	0.7978752	0.7803340	0.7629759
	CL-MV	0.8518415	0.8419580	0.8329371	0.7633132	0.7825715	0.7701328
	MCL-MV	0.8524476	0.8427273	0.8319814	0.7972553	0.7834106	0.7689298
Website	Average	0.9076151	0.9015507	0.9011836	0.8374854	0.8263294	0.8252411
	Majority vote	0.9085733	0.9024412	0.9029630	0.8390778	0.8276560	0.8285420
	CL-MV	0.9082770	0.9028035	0.9031095	0.8386970	0.8285752	0.8288027
	MCL-MV	0.9084976	0.9024364	0.9031836	0.8390036	0.8279560	0.8289320
Vehicle	Average	0.7432857	0.7427857	0.7338175	0.6603490	0.6592262	0.6471487
	Majority vote	0.7429206	0.7400952	0.7327381	0.6598246	0.6556408	0.6457060
	CL-MV	0.7422381	0.7430238	0.7337143	0.6588553	0.6595597	0.6471186
	MCL-MV	0.7427143	0.7419444	0.7333571	0.6595494	0.6581548	0.6466841

Acknowledgments The author is supported by Ministerio de Ciencia, Innovación y Universidades, Gobierno de España, project ref. PGC2018-097848-B-I0.

References

1. Abdelhamida N, Ayesha A, Thabtah F (2014) Phishing detection based associative classification data mining. Expert Syst Appl 41:5948–5959
2. Amrieh EA, Hamtini T, Aljarah I (2015) Preprocessing and analyzing educational dataset using X-API for improving student's performance. In: IEEE jordan conference on applied electrical engineering and computing technologies (AEECT), Amman, 1–5
3. Amrieh EA, Hamtini T, Aljarah I (2016) Mining educational data to predict student's academic performance using ensemble methods. Int J Database Theor Appl 9(8):119–136
4. Benjamin Fredrick David H, Suruliandi A, Raja SP (2019) Preventing crimes ahead of time by predicting crime propensity in released prisoners using Data Mining techniques. Int J Appl Decis Sci 12(3):307–336
5. Bi Y, Guan J, Bell D (2008) The combination of multiple classifiers using an evidential reasoning approach. Artif Intell 172(15):1731–1751
6. Breiman L (1996) Bagging predictors. Mach Learn 26(2):123–140
7. Breiman L (2001) Random forests. Machine Learn 45:5–32
8. Cortez P, Silva A (2008) Using data mining to predict secondary school student performance. In: Brito A, Teixeira J (eds) Proceedings of 5th FUture BUsiness TEChnology conference (FUBUTEC 2008) pp. 5–12, Porto, Portugal, April, 2008, EUROSIS ISBN 978-9077381-39-7,
9. Delgado R, Tibau XA (2019) Why Cohen's Kappa should be avoided as performance measure in classification. PLoSONE 14(9):e0222916. https://doi.org/10.1371/journal.pone.0222916
10. Dettling M, Bühlmann P (2003) Boosting for tumor classification with gene expression data. Bioinformatics 19(9):1061–1069
11. Dietterich TG (1997) Machine-learning research: Four current directions. The AI Magazine 8(4):97–136
12. Dietterich TG (2000) An experimental comparison of three methods for constructing ensembles of decision trees: bagging, boosting, and randomization. Mach Learn 40:139–157
13. Dua D, Graff C (2019) UCI Machine Learning Repository. [Http://archive.ics.uci.edu/ml]. irvine, CA: University of California School of Information and Computer Science
14. Freund Y, Schapire RE (1996) Experiments with a new boosting algorithm. In: ICML'96: Proceedings of the thirteenth international conference on international conference on machine learning, vol 96, pp 148–156
15. Gorodkin J (2004) Comparing two k-category assignments by a k-category correlation coefficient. Computat Biol Chemist 28(5-6):367–374
16. Hansen LK, Salamon P (1990) Neural networs ensembles. IEEE Trans Pattern Anal Machine Intell 12(10):993–1001
17. Hao J, Zhang B, Yue K, Wang J, Wu HSun X, Chao HC, You X, Bertino E (eds) (2017) Performance measurement and configuration optimization of virtual machines based on the bayesian network, vol 10603. Springer, Cham

18. Kittler J, Hatef M, Duin RPW, Matas J (1998) On combining classifiers. IEEE Trans Pattern Anal Machine Intell 20(3):226–239
19. Kuncheva LI (2004) Combining pattern classifiers. Wiley, New York
20. Liu Q, Lu J, Chen S, Zhao K (2014) Multiple naïve Bayes Classifiers Ensemble for Traffic Incident Detection, Mathematical Problems in Engineering, vol. 2014, Article ID 383671, 16 pages Hidawi Publishing Corporation
21. Matthews BW (1975) Comparison of the predicted and observed secondary structure of t4 phage lysozyme. Biochim Biophys Acta (BBA)-Protein Struct 405(2):442–451
22. R Core Team R (2013) A language and environment for statistical computing. R Foundation for Statistical Computing, Vienna, Austria http://www.R-project.org/
23. Shapiro SS, Wilk MB (1965) An analysis of variance test for normality (complete samples). Biometrika 52(3-4):591–611
24. Shapley L, Grofman B (1984) Optimizing group judgement accuracy in the presence of interdependencies. Public Choice 43:329–343
25. Sharma M, Bilgic M (2017) Evidence-based uncertainty sampling for active learning. Data Minning Knowl Discov 31:164–202
26. Svetnik V, Liaw A, Tong C, Culberson JC, Sheridan RP, Feuston BP (2003) Random forest: A classification and regression tool for compound classification and QSAR modeling. J Chem Inform Comput Sci 43(6):1947–1958
27. Twala B (2010) Multiple classifier application to credit risk assessment. Expert Syst Appl 37(4):3326–3336
28. Wilcoxon F (1945) Individual comparisons by ranking methods. Biomet Bull 1(6):80–83
29. Wolpert DH, Macready WG (1997) No free lunch theorems for optimization. IEEE Trans Evol Comput 1(1):67–82
30. Xu L, Krzyzak A, Suen CY (1992) Methods of combining multiple classifiers and their application to handwriting recognition. IEEE Trans Syst Man Cybern 22:418–435



Rosario Delgado is Professor of Statistics and Operational Research at the Department of Mathematics of the University Autonomous of Barcelone (UAB) since 1998 and belong to the research group of the Department of Mathematics on Stochastic Processes. Degree in Mathematics from the Universtity of Barcelone (UB) in year 1987, obtained a Ph D. in Mathematics from the same university in 1994.