An Empirical Comparison of Supervised Ensemble Learning Approaches

Mohamed Bibimoune[1,2], Haytham Elghazel[1], Alex Aussem[1]

[1] Université de Lyon, CNRS
Université Lyon 1, LIRIS UMR 5205, F-69622, France
mohamed.bibimoune@univ-lyon1.fr, haytham.elghazel@univ-lyon1.fr,
alexandre.aussem@univ-lyon1.fr
[2] ProbaYes,
82 allée Galilée, F-38330 Montbonnot, France

Abstract. We present an extensive empirical comparison between twenty prototypical supervised ensemble learning algorithms, including Boosting, Bagging, Random Forests, Rotation Forests, Arc-X4, Class-Switching and their variants, as well as more recent techniques like Random Patches. These algorithms were compared against each other in terms of threshold, ranking/ordering and probability metrics over nineteen UCI benchmark datasets with binary labels. We also examine the influence of two base learners, CART and Extremely Randomized Trees, and the effect of calibrating the models via Isotonic Regression on each performance metric. The selected datasets were already used in various empirical studies and cover different application domains. The experimental analysis was restricted to the hundred most relevant features according to the SNR filter method with a view to dramatically reducing the computational burden involved by the simulation. The source code and the detailed results of our study are publicly available.

Key words: Ensemble learning, classifier ensembles, empirical performance comparison.

1 Introduction

The ubiquity of ensemble models in Machine Learning and Pattern Recognition applications stems primarily from their potential to significantly increase prediction accuracy over individual classifier models [25]. In the last decade, there has been a great deal of research focused on the problem of boosting their performance, either by placing more or less emphasis on the hard examples, by constructing new features for each base classifier, or by encouraging individual accuracy and/or diversity within the ensemble. While the actual performance of any ensemble model on a particular problem is clearly dependent on the data and the learner, there is still much room for improvement as the comparison between all the proposals provide valuable insight into understanding their respective benefit and their differences.

There are few comprehensive empirical studies comparing ensemble learning algorithms [1, 9]. The study performed by Caruana and Niculescu-Mizil [9] is perhaps the best known study however it is restricted to small subset of well established ensemble methods like random forests, boosted and bagged trees, and more classical models (e.g., neural networks, SVMs, Naive Bayes). On the other had, many authors have compared their ensemble classifier proposal with others. For instance, Zhang et al. compared in [29] RotBoost against Bagging, AdaBoost, MultiBoost and Rotation Forest using decision tree-based estimators, over 36 data sets from the UCI repository. In [23], Rodriguez et al. examined the Rotation Forest ensemble on a selection of 33 data sets from the UCI repository and compared it with Bagging, AdaBoost, and Random Forest with decision trees as the base classifier. More recently, Louppe et al. investigated a very simple, yet effective, ensemble framework called *Random Patches* that builds each individual model of the ensemble from a random patch of data obtained by drawing random subsets of both instances and features from the whole dataset. With respect to AdaBoost and Random Forest, these experiments on 16 data sets showed that the proposed method provides on par performance in terms of accuracy while simultaneously lowering the memory needs, and attains significantly better performance when memory is severely constrained. Despite these attempts that have emerged to enhance the capability and efficiency, we believe an extensive empirical evaluation of most of the ensemble proposal algorithms can shed some light into the strength and weaknesses.

We briefly review these algorithms and describe a large empirical study comparing several ensemble method variants in conjunction with two types of unpruned decision trees : the standard CART decision tree and another randomized variant called Extremely Randomized Tree (ET) proposed by Geurts et al in [13] as base classifier, both using the Gini splitting criterion. As noted by Caruana et al. [9], different performance metrics are appropriate for each domain. For example Precision/Recall measures are used in information retrieval; medicine prefers ROC area; Lift is appropriate for some marketing tasks, etc. The different performance metrics measure different tradeoffs in the predictions made by a classifier. One method may perform well on one metric, and worse on another, hence the importance to gauge their performance on several performance metrics to get a broader picture. We evaluate the performance of Boosting, Bagging, Random Forests, Rotation Forests, and their variants including LogitBoost, VadaBoost, RotBoost, and AdaBoost with stumps. For the sake of completeness, we added more recent techniques like Random Patches and less conventional techniques like Class-Switching and Arc-X4. All these voting algorithms can be divided into two types: those that adaptively change the distribution of the training set based on the performance of previous classifiers (as in boosting methods) and those that do not (as in Bagging). Our purpose was not to cover all existing methods, and we have restricted ourselves to well performing methods that have been presented in the literature, without claiming exhaustivity, but trying to cover a wide range of implementation ideas.

The data sets used in the experiments were all taken from the UCI Machine Learning Repository. They represent a variety of problems but do not include high-dimensional data sets owing to the computational expense of running Rotation Forests. The comparison is performed based on three performance metrics: accuracy, ROC Area and squared error. For each algorithm we examine common parameters values. Following [9] and [22], we also examine the effect that calibrating the models via Isotonic Regression has on their performance.

The paper is organized as follows. In Section 2, we begin with basic notation and follow with a description of the base inducers that build classifiers. We use two variants of decision tree inducers: unlimited depth, and extremely randomized tree. We then describe three performance metrics and the Isotonic calibration method that we use throughout the paper. In Section 3, we describe our set of experiments with and without calibration and report the results. We raise several issues and for future work in Section 4 and conclude with a summary of our contributions.

2 Ensemble Learning Algorithms & Parameters

Before discussing the ensemble algorithms chosen in this comprehensive study, we would like to mention that, contrary to [9] which attempted to explore the space of parameters for each learning algorithm, we decided to fix the parameters to their common value except for a few data dependent extra parameters that have to be finely pretuned. The number of trees was fixed to 200 in accordance with a recent empirical study [15] which tends to show that ensembles of size less or equal to 100 are too small for approximating the infinite ensemble prediction. Although it is shown that for some datasets the ensemble size should ideally be larger than a few thousands, our choice for the ensemble size tries to balance performance and computation cost. This shall now summarize the parameters used for each learning algorithm below.

Bagging (Bag) [4]: Practically, Bag has many advantages. It is fast, simple and easy to program. It has no parameters to tune. Bag is sometimes proposed with an optimization of the bootstraps samples size to perform better. However we fixed the default size equal to the size of the initial dataset.

Random Forests (RF) [7]: the number of feature selected at each node for building the trees was fixed to the root square of the total number of features.

Random Patches (RadP) [19]: this method was proposed very recently to tackle the problem of insufficient memory w.r.t. the size of the data set. The idea is to build each individual model of the ensemble from a random patch of data obtained by drawing random subsets of both instances and features from the whole dataset; p_s and p_f are hyper-parameters that control the number of samples and features in a patch. These parameters are tuned using an independent validation dataset. It is worth mentioning that RadP was initially designed to overcome some shortcomings of the existing ensemble techniques in the context of huge data sets. As such, they were not meant to outperform the other methods

on small data sets or without an memory limitation. We chosed, however, this algorithm as an interesting alternative to Bag and RF.

AdaBoost (Ad) [11]: we used the standard algorithm proposed by Freund and Schapire.

AdaBoost Stump (AdSt): in this particular version of Ad, the base learner is replaced by a stump. A stump is a decision tree with only one node. While the base learner is highly biased, when combined with AdaBoost, it is believed to compete with the best methods while providing a serious computational advantage.

VadaBoost (Vad) [26]: this is another ensemble method called Variance Penalizing AdaBoost that appeared recently in the literature. VadaBoost is similar to AdaBoost except that the weighting function tries to minimize both empirical risk and empirical variance. This modification is motivated by the recent empirical bound which relates the empirical variance to the true risk. Vad depends on a hyper-parameter, λ, that will be tuned on a validation set.

Arc-X4 (ArcX4) [5]: the method belongs to the family of Arcing (Adaptive Resampling and Combining) algorithms. It started out as a simple mechanism for evaluating the effect of Ad.

LogitBoost (Logb) [12]: LogitBoost is a boosting algorithm formulated by Friedman et al. Their original paper [12] casts the Ad algorithm into a statistical framework. When regarded as a generalized additive model, the Logb algorithm is derived by applying the cost functional of logistic regression. Note that there is no final vote as each base classifier is not an independent classifier but rather a correction for the whole model.

Rotation Forests (Rot) [23]: this method builds multiple classifiers on randomized projections of the original dataset The feature set is randomly split into K subsets (K is a parameter of the algorithm) and PCA is applied to each subset in order to create the training data for the base classifier. The idea of the rotation approach is to encourage simultaneously individual accuracy and diversity within the ensemble. The size of each subsets of feature was fixed to 3 as proposed by Rodriguez. The number of sub classes randomly selected for the PCA was fixed to 1 as we focused on binary classification. The size of the bootstrap sample over the selected class was fixed to 75% of its size.

RotBoost (Rotb) [29]: this method combines Rot and Ad. As the main idea of Rot is to improve the global accuracy of the classifiers while keeping the diversity through the projections, the idea here is to replace the decision tree by Ad. This can be seen as an attempt to improve Rot by increasing the base learner accuracy without affecting the diversity of the ensemble. The final decision is the vote over every decision made by the internal Ad. The parameter setup for Rotb is the same as for Rot. In order to be fair in term of ensemble size, we construct an ensemble consisting of 40 Rotation Forests which are learned by Ad during 5 iterations. Hence the total number of trees is 200. This ratio has been shown to be approximatively the empirically best in [29].

Class-Switching (Swt) [6]: Swt is a variant of the output flipping ensembles proposed by Martinez-Munoz and Suarez in [21]. The idea is to randomly

switch the class labels at a certain user defined rate p. The decision of the final classifier is again the plurality vote over these base classifiers. p will be tuned on a validation set.

Considering the four data dependent parameters mentioned above (i.e., p_s, p_f,p and λ), we randomly split each dataset into two parts, 80% for training and 20% for validation, The later is used to search the best hyper-parameters and is not used afterwards for training or comparison purposes (it will be discarded from the whole data set). We then construct the ensemble on the training set by increasing each parameters from 0.1 to 1.0. The parameters yielding the best accuracy on the validation set are retained. It is worth noting that the other two performance metrics (i.e., mean square error and AUC) could also be applied for parametrization. All the above methods were implemented in Matlab - except the CART algorithm in the Matlab statistics toolbox and the ET algorithm in the regression tree package [13] -, in order to make fair comparisons and also because some algorithms are not publicly available (e.g., random patches, output switching). To make sure our Matlab implementations were correct, we did a sanity check against previous papers on ensemble algorithms.

2.1 The decision tree inducers

As mentioned above, we use two distinct decision tree inducers: a decision tree (CART) and a so-called Extremely Randomized Tree (ET) proposed in [13]. In [19], Louppe and Geurts found out that every sub-sampling (samples and/or feature) ensemble method they experimented with was improved when ET was used as base learner instead of a standard decision tree. ET is a variant of decision tree which aims to reduce even more the variance of ensemble methods by reducing the variance of the tree as base learner. At each node, instead of cutting at the best threshold among every possible ones, the method selects an attribute and a threshold at random. To avoid very bad cuts, the score-measure of the selected cut must be higher than a user-defined threshold otherwise it has to be re-selected. This process is repeated until a convenient threshold is found or until it does not remain any attribute to pick up (The algorithm uses one threshold per attribute). According to the authors, the reducing variance strength of his algorithm arises from the fact that threshold are selected totally at random, contrary to preceding methods proposed by Kong and Dietterich in [18] which select at random a threshold among the best ones or by Ho in [16] which select the best one among a fixed number of thresholds. Therefore, we used both unpruned DT and ET as base learners. For ET, we used he regression tree package proposed in [13]. To distinguish ensemble with DT and ET, we added 'ET' at the end of the algorithm names to indicate that extremely randomized trees are used.

2.2 Performance Metrics & Calibration

The performance metrics can be splitted into three groups: threshold metrics, ordering/rank metrics and probability metrics [8]. For thresholded metrics, like

accuracy (ACC), it makes no difference how close a prediction is to a threshold, usually 0.5, what matters whether it is above or below the threshold. In contrast, the ordering/rank metrics, like the area under the ROC curve (AUC), depend only on the ordering of the instances, not the actual predicted values, while the probability metrics, like the squared error (RMS), interpret the predicted value of each instance as the conditional probability of the output label being in the positive class given the input.

In many applications it is important to predict well calibrated probabilities; good accuracy or area under the ROC curve are not sufficient. Therefore, all the algorithms were run twice, with and without post calibration, in order to compare the effects of calibrating ensemble methods on the overall performance. The idea is not new, Niculescu-Mizil and Caruana have investigated in [9] the benefit of two well known calibration methods, namely Platt Scaling and Isotonic Regression [28], on the performance of several classifiers. They concluded that AdaBoost and good ranking algorithms in general are those which draw the most benefits from calibration. As expected, these benefits are the most noticeable on the root mean squared error metric. In this paper, we only focus on Isotonic Regression because it was originally designed for decision trees model although Platt Scaling could also applied to decision trees. To this purpose, we use the pair-adjacent violators (PAV) algorithm described in [28, 9] that finds a piecewise constant solution in linear time.

2.3 Data sets

We compare the algorithms on nineteen binary classification problems of various sizes and dimensions. Table 1 summarizes the main characteristics of these data sets utilized in our empirical study. This selection includes data sets with different characteristics and from a variety of fields. Among them, we find some data sets with thousands of features. As explained by Liu in [17], if Rot or Rotb are applied to classify such datasets, a rotation matrix with thousands of dimensions is required for each tree, which entails a dramatic increase in computational complexity. To keep the running time reasonable, we had no choice but to resort to a dimension reduction technique for these problems; the same strategy was adopted in several works [29, 23, 17]. Based on Liu's comparison, we took the best of the three proposed filter methods for rotation forest, the signal to noise ratio [27]. SNR was used to rank all the features; we kept the 100 top relevant features and discarded the others. Of course this choice necessarily entails some compromises as there will generally be some loss of information. So the reader shall bear in mind that the actual size of the data sets is limited to 100 features in the experiments.

3 Performances analysis

In this section, we report the results of the experimental evaluation. For each test problem, we use 5-fold cross validation (CV) on 80% of the data (recall

Table 1. Characteristics of the nineteen problems used in this study

Data sets	*#inst*	*#feat*	*#labels*	*Reference*
Basehock	1993	4862	2	[30]
Breast-cancer	699	9	2	[3]
Cleve	303	13	2	[3]
Colon	62	2000	2	[2]
Ionosphere	351	34	2	[3]
Leukemia	73	7129	2	[14]
Madelon	2600	500	2	[3]
Musk	476	166	2	[3]
Ovarian	54	1536	2	[24]
Parkinson	195	22	2	[3]
PcMac	1943	3289	2	[30]
Pima	768	8	2	[3]
Promoters	106	57	2	[3]
Relathe	1427	4322	2	[30]
Smk-Can	187	19993	2	[30]
Spam	4601	57	2	[3]
Spect	267	22	2	[3]
Wdbc	569	30	2	[3]
Wpbc	194	33	2	[3]

that 20% of each data set is used to calibrate the models and to select the best parameters). In order to get reliable statistics over the metrics, the experiments were repeated 10 times. So the results obtained are averaged over 50 iterations which allows us to apply statistical tests in order to discern significant differences between the 20 methods.

Detailed average performances of the 20 methods for all 19 data sets using the protocol described above are reported in Tables 1-6 of the supplementary material[1]. For each evaluation metric, we present and discuss the critical diagrams from the tests for statistical significance using all data sets.

Table 2 shows the normalized score for each algorithm on each of the three metrics. Each entry in the table averages these scores across the fifty trials and nineteen test problems. The table is divided into two blocks to separately illustrate the performances for both calibrated and uncalibrated models. The last column per block, Mean, is the mean (only for illustration purposes, not for statistical analysis) over the three metrics ($ACC, AUC, 1 - RMS$) and nineteen problems, and fifty trials. In the table, higher scores always indicate better performance.

Considering all three metrics together, it appears that the strongest models among the uncalibrated ones are Rotation Forest (Rot), Rotation Forest using extremely randomized tree (RotET), Rotboost (Rotb) and its ET-based variant

[1] `http://perso.univ-lyon1.fr/haytham.elghazel/copem2013-supplementary.pdf`

Table 2. Average normalized scores by metric for each learning algorithm obtained over nineteen test problems. We give complete results over all evaluation metrics in supplementary material.

Approach	Uncalibrated Models				Calibrated Models			
	ACC	AUC	1-RMS	Mean	ACC	AUC	1-RMS	Mean
Rot	0,865	0,903	0,700	0,823	0,837	0,864	0,673	0,791
Bag	0,823*	0,875*	0,660*	0,786	0,820*	0,844	0,649*	0,771
Ad	0,857	0,893	0,668*	0,806	0,836	0,863	0,669	0,789
RF	0,864	0,896	0,689	0,816	0,835	0,857	0,669	0,787
Rotb	0,865	0,897	0,702	0,821	0,841	0,861	0,676	0,793
ArcX4	0,852*	0,892*	0,686	0,810	0,829	0,853	0,659*	0,780
AdSt	0,833*	0,874*	0,598*	0,769	0,817*	0,845	0,653	0,771
CART	0,811*	0,809*	0,617*	0,746	0,808*	0,806*	0,622*	0,745
Logb	0,845	0,884	0,635*	0,788	0,823	0,854	0,660	0,779
Swt	0,859	0,888*	0,638*	0,795	0,829*	0,848*	0,660*	0,779
RadP	0,850	0,889	0,669*	0,803	0,836	0,851	0,662	0,783
Vad	0,858	0,894	0,684	0,812	0,839	0,864	0,671	0,791
RotET	0,871	0,901	0,698	0,823	0,843	0,858	0,675	0,792
BagET	0,836*	0,893	0,673*	0,800	0,833	0,852	0,663	0,783
AdET	0,862	0,898	0,667*	0,809	0,838	0,861	0,674	0,791
RotbET	**0,866**	0,900	**0,704**	0,824	**0,844**	0,859	0,678	0,794
ArcX4ET	0,868	0,901	0,693	0,821	0,842	0,859	0,673	0,791
SwtET	0,866	0,890	0,649*	0,802	0,841	0,850	0,673	0,788
RadPET	0,861	**0,908**	0,680	0,816	0,844	**0,867**	**0,678**	0,797
VadET	0,864	0,899	0,681	0,815	0,841	0,864	0,678	0,794

(RotbET), and ArcX4ET. Among calibrated models, the best models overall are Rotation Forest (Rot) and its ET-based variant (RotET), Rotboost (Rotb) and its ET-based variant (RotbET), boosted extremely randomized trees (AdET), ArcX4ET, Vadaboost (Vad) and its ET-based variant (VadET), and Random Patch using extremely randomized tree (RadPET). With or without calibration, the poorest performing models are decision trees (CART), bagged trees (Bag), and AdaBoost Stump (AdSt). Looking at individual metrics, calibration generally slightly degrades the results on accuracy and AUC and is remarkably effective at obtaining excellent performance on the RMS score (probability metric) for especially boosting-based algorithms. Indeed, calibration improves the performance (in terms of RMS) of boosted stumps (AdSt), LogitBoost (Logb), Class-Switching with or without extremely randomized trees (Swt and SwtEt), and provides a small, but noticeable improvement for boosted trees with or without extremely randomized trees (Ad and AdET), and a single tree (CART). If we consider only large data sets in Tables 1-6 of the supplementary materials (i.e. Ovarian, Smk-Can, Leukemia), reported results show that RMS values decrease with calibration when boosting-based approaches are used, while their AUC and ACC are not affected.

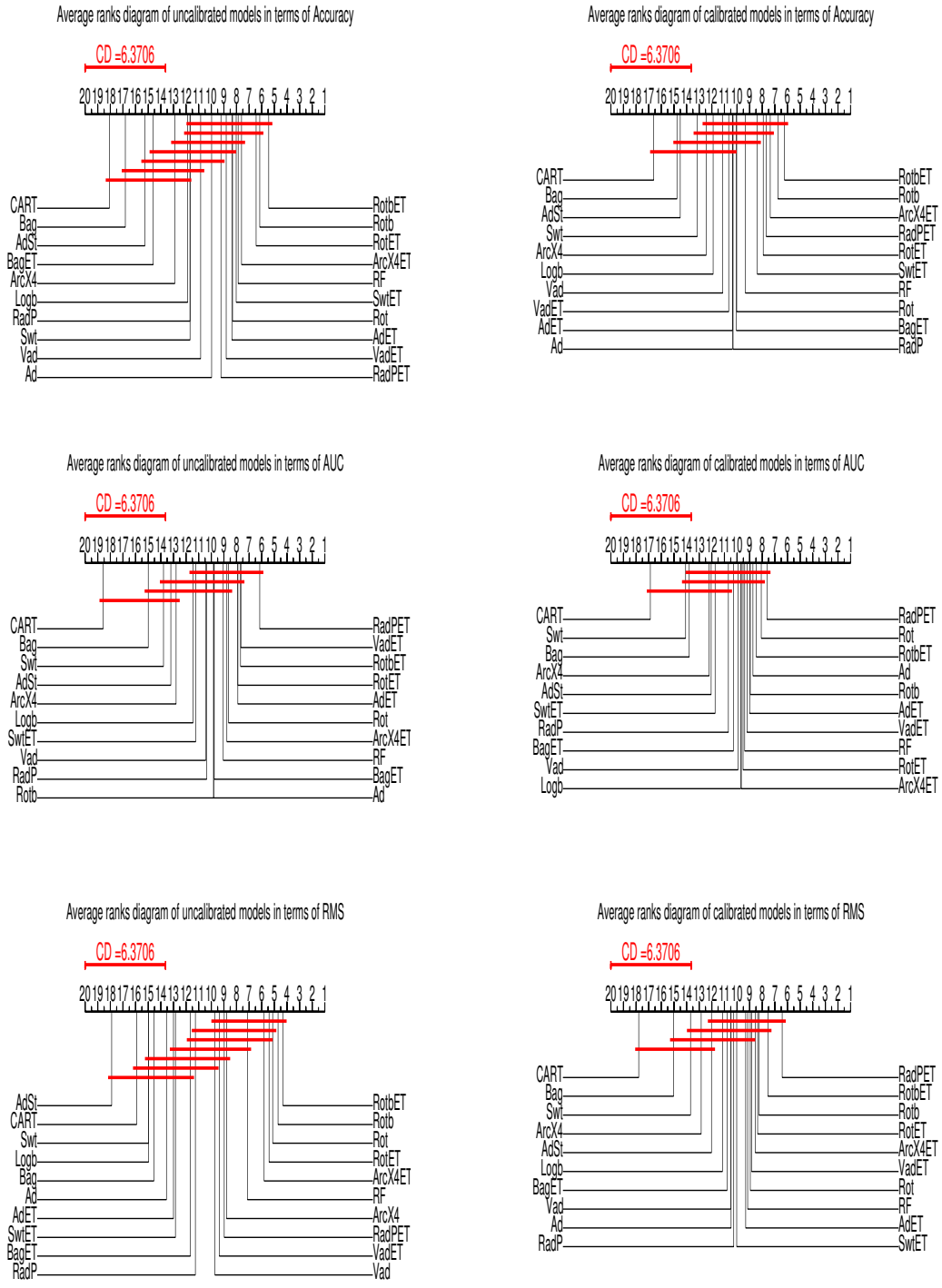
Regarding now the performances of ET-based variants of the algorithms, across all three metrics, with or without calibration, it is observed that each ensemble method with ET always outperforms ensembles of standard DT. This observation confirms the results obtained in [19] and clearly suggests that using random split thresholds, instead of optimized ones like in DT, pays off in terms of generalization error, especially for small data sets.

In order to better assess the results obtained for each algorithm on each metric, we adopt in this study the methodology proposed by [10] for the comparison of several algorithms over multiple datasets. In this methodology, the non-parametric Friedman test is firstly used to evaluate the rejection of the hypothesis that all the classifiers perform equally well for a given risk level. It ranks the algorithms for each data set separately, the best performing algorithm getting the rank of 1, the second best rank 2 etc. In case of ties it assigns average ranks. Then, the Friedman test compares the average ranks of the algorithms and calculates the Friedman statistic. If a statistically significant difference in the performance is detected, we proceed with a *post hoc* test. The Nemenyi test is used to compare all the classifiers to each other. In this procedure, the performance of two classifiers is significantly different if their average ranks differ more than some critical distance (CD). The critical distance depends on the number of algorithms, the number of data sets and the critical value (for a given significance level p) that is based on the Studentized range statistic (see [10] for further details). In this study, the Friedman test reveals statistically significant differences ($p < 0.05$) for each metric with and without calibration. As seen in table 2, the algorithm performing best on each metric is boldfaced. Algorithms performing significantly worse than the best algorithm at $p = 0.1$ (CD=6.3706) using the Nemenyi posthoc test are marked with '$\star$' next to them. Furthermore, we present the result from the Nemenyi posthoc test with average rank diagrams as suggested by Demsar [10]. These are given on Figure 1. The ranks are depicted on the axis, in such a manner that the best ranking algorithms are at the rightmost side of the diagram. The algorithms that do not differ significantly (at $p = 0.1$) are connected with a line. The critical difference CD is shown above the graph.

As may be observed in Figure 1, ET-based variant of Rotboost (RotbET) performs best in terms of accuracy. In the average ranks diagrams corresponding to accuracy, two groups of algorithms could be separated. The first consists of all algorithms which have seemingly similar performances with the best method (*i.e.* RotbET). The second contains the methods that performs significantly worse than RotbET, including Bagging (Bag) and its ET-based variant (BagET); ArcX4, Boosted stumps (AdS) and single tree (CART).

The statistical tests we use are conservative and the differences in performance for methods within the first group are not significant. To further support these rank comparisons, we compared the 50 accuracy values obtained over each dataset split for each pair of methods in the first group by using the paired t-test (with $p = 0.05$) as done [19]. The results of these pairwise comparisons are depicted (see the supplementary material) in terms of "Win-Tie-Loss" sta-

Fig. 1. Average ranks diagram comparing the 20 algorithms in terms of three metrics (Accuracy, AUC and RMS)



tuses of all pairs of methods; the three values in each cell (i, j) respectively indicate how times many the approach i is significantly better/not significantly different/significantly worse than the approach j. Following [10], if the two algorithms are, as assumed under the null-hypothesis, equivalent, each should win on approximately $N/2$ out of N data sets. The number of wins is distributed according to the binomial distribution and the critical number of wins at $p = 0.1$ is equal to 13 in our case. Since tied matches support the null-hypothesis we should not discount them but split them evenly between the two classifiers when counting the number of wins; if there is an odd number of them, we again ignore one.

In the Table 7 (see the supplementary material), each pairwise comparison entry (i, j) for which the approach i is significantly better than j is boldfaced. The analysis of this table reveals that the approaches that are never beaten by any other approach are: all the Rotation Forest-based methods (Rot, Rotb, RotET and RotbET), AdET and ArcX4ET. We may also notice from Figure 1 and Table 8 (see the supplementary material) for accuracy on calibrated models the following. First, the calibration is beneficial to Random Patch algorithms (RadP and RadPET) and Bagged trees (BagET) in terms of ranking. It hurts the ranking of boosted trees but does not affect the performances of Rotation Forest-based methods and ArcX4ET. Overall, RotbET is ranked first, then come Rotb, ArcX4ET and RadPET. Looking at Table 8 (see the supplementary material), the dominating approaches include again all Rotation Forest-based methods and ArcX4ET, as well as RadPET and VadET (*c.f.* Table 3). Another interesting observation upon looking at the average rank diagrams is that ensembles of ET lie mostly on the right side of the plot compared to their DT counterparts, hence their superior performance.

As far as the AUC is concerned (*c.f.* Figure 1), RadPET ranks first. However, its performance is not statistically distinguishable from the performance of five other algorithms: RotET, RotbET, Ad, AdET and VadET (*c.f.* Table 9 in supplementary material). In our experiments, ET improved the ranking of all ensemble approaches by at least 10% on average when compared to DT. This corroborate our previous finding, namely that ET should be preferred to DT in the ensembles. Figure 1 and Table 10 (see the supplementary material) indicate that calibration reduces the ranking of some approaches, especially VadET and RotET (among the best uncalibrated approaches in terms of AUC) but slightly improves the ranks of the approaches that adaptively change the distribution (Logb, AdSt, Ad, Vad, Rotb, ArcX4) and Rot. This explain why equally performing methods like RadPET are, after calibration, judged insignificant (*c.f.* Table 3).

Regarding the RMS results reported in Figure 1 and Table 11 (see the supplementary material). Rot, Rotb and RotbET significantly outperforms the other approaches. Here again, ET-based method outperforms the DT ones by a noticeable margin. We found calibration to be remarkably effective at improving the ranking of boosting-based algporithms in terms of RMS values, especially Ad, AdET, AdSt, Logb and VadET. This is the reason why that algorithms

Table 3. List of dominating approaches per metric, with and without calibration

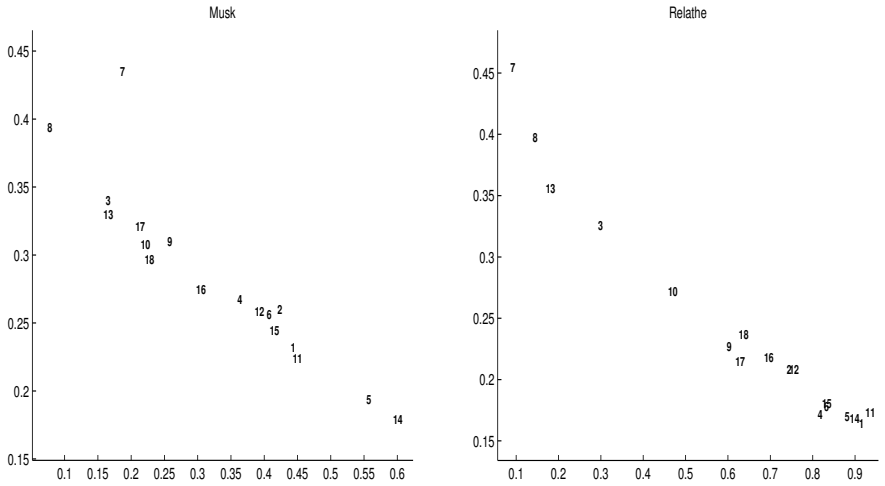
Metric	Without calibration	With calibration
ACC	AdET, ArcX4ET, Rot, Rotb, RotET, RotbET	ArcX4ET, Rot, Rotb, RotbET, RotET, RadPET, VadET
AUC	Ad, AdET, RotET, RotbET, RadPET, VadET	Ad, AdET, ArcX4ET, Logb, Rot, Rotb, RotbET, RadPET, Vad, VadET
RMS	Rot, Rotb, RotbET	Ad, AdET, Logb, Rot, Rotb, RotET, RotbET, RadPET, Vad, VadET

that adaptively change the distribution have integrated the list of dominating approaches (*c.f.* Table 3).

3.1 Diversity-error diagrams

To achieve higher prediction accuracy than individual classifiers, it is crucial that the ensemble consists of highly accurate classifiers which at the same time disagree as much as possible. To illustrate the diversity-accuracy patterns of the ensemble, we use the kappa-error diagrams proposed in [20]. The latter are scatterplots with $L \times (L-1)/2$ points, where L is the committee size. Each point corresponds to a pair of classifiers. On the x-axis is a measure of diversity between the pair, κ. On the y-axis is the averaged individual error of the classifiers in the pair, $e_{i,j} = (e_i + e_j)/2$. As small values of κ indicate better diversity and small values of $e_{i,j}$ indicate better performance; the diagram of an ideal ensemble should be filled with points in the bottom left corner. Since we have a large number of algorithms to compare and due to space limitation, we only plot the distance between their corresponding centroids in Figure 2 for the 18 ensemble methods (Logb and CART are excluded), for the "Musk" and "Relathe" data sets only. The following is observed: (1) Rot-based algorithms outperform the others in terms of accuracy; (2) ArcX4, Bag and RF exhibit equivalent patterns, they are slightly more diverse but slightly less accurate than Rot-based algorithms; (3) while boosting-based methods (AdSt, Ad, AdET) and switching are more diverse, their accuracies are lower than the others, except SwtET as ET is generally able to increase the individual accuracy, and (4) no clear picture emerged when one examines Random Patch-based algorithms. Not surprinsingly, as the classifiers become more diverse, they become less accurate and vice versa. Furthermore, according to the results in the previous subsection, it seems that the more accurate the base classifiers are, the better the performance. This corroborates the conclusion drawn in [23], namely that individual accuracy is probably the more crucial component of the tandem diversity-accuracy, contrary to the diversifying strategies.

Fig. 2. Centroids of κ-Error Diagrams of different ensemble approaches for two data sets. x-axis= κ, y-axis= $e_{i,j}$ (average error of pair of classifiers). (01) Rot; (02) Bag; (03) Ad; (04) RF; (05) Rotb; (06) ArcX4; (07) AdSt; (08) Swt; (09) RadP; (10) Vad; (11) RotET; (12) BagET; (13) AdET; (14) RotbET; (15) ArcX4ET; (16) SwtET; (17) RadPET; (18) VadET.



The kappa-error relative movement diagrams in Figure 3 display the difference between the κ and accuracy of the DT-based method and the ET-based one. There are as many points as data sets. Points in the upper-right corner represent datasets for which the ET-based method outperformed the standard DT-based algorithm in terms of both diversity and accuracy, points up-left indicate that ET-based method improved the accuracy but degrades diversity. We may notice that ET as a base learner improves one criteria at the expense of the other. Furthermore, according to the resulting win/tie/loss counts for each ET-based approach against the DT-based one summarized in Table 4, we find that the approaches for which the ET-variant is significantly superior to the standard one are those for which the accuracy (*i.e.* Swt) or the diversity (*i.e.* Bag, ArcX4 and RadP) is significantly better.

Before we conclude, we would like to mention that some of the above findings need to be regarded with caution. We list a few caveats and our comments on these.

– The experimental analysis was restricted to the 100 most relevant features with a view to dramatically reducing the computational burden required to run Rotation Forest-based methods. Thus, the results reported here are valid for data sets of small to moderate sizes. The data sets used in the experiments did not include very large-scale data sets. Moreover, the complexity

Fig. 3. Centroids of κ-Error relative movement diagrams of different ensemble approaches

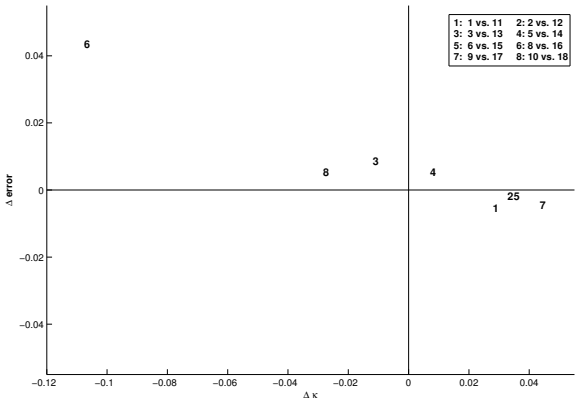


Table 4. The win/tie/loss results for ET-based ensembles vs. DT-based ensembles. Bold cells indicate significant differences at $p = 0.1$

Approaches	Uncalibrated Models			Calibrated Models			In Total
	ACC	AUC	RMS	ACC	AUC	RMS	
RotET/Rot	8/8/3	11/2/6	7/6/6	6/11/2	7/8/4	8/7/4	47/42/25
BagET/Bag	**11/6/2**	**13/4/2**	**13/3/3**	**13/5/1**	**12/5/2**	**12/6/1**	74/29/11
AdET/Ad	7/10/2	7/10/2	**11/4/4**	6/11/2	4/8/7	6/12/1	41/55/18
RotbET/Rotb	3/12/4	6/10/3	5/11/3	3/13/3	3/11/5	4/10/5	24/67/23
ArcX4ET/ArcX4	**14/5/0**	**13/2/4**	**13/1/5**	**10/9/0**	9/7/3	**14/4/1**	73/28/13
SwtET/Swt	**10/8/1**	9/5/5	**13/2/4**	**14/3/2**	**10/6/3**	**13/4/2**	69/28/17
RadPET/RadP	**9/10/0**	**10/7/2**	**14/1/4**	**10/7/2**	**12/4/3**	**13/4/2**	68/33/13
VadET/Vad	**10/7/2**	**9/9/1**	9/5/5	6/9/4	3/11/5	7/9/3	44/50/20

issue should be addressed to balance the computation cost with the obtained performance in a real scenario.

– We used the same ensemble size $L = 200$ for all methods. It is known that bagging fares better for large L. On the other hand, AdaBoost would benefit from tuning L. It is not clear what the outcome would be if L was treated as hyperparameter and tuned for all ensemble methods compared here. We acknowledge that a thorough experimental comparison of a set of methods needs tuning each of the methods to its best for every data set and every performance metric. Interestingly, while VadaBoost, Class-Switiching and Random Patches were slightly favored as we tuned some of their parameters on an independent validation set, these methods were not found to compare favorably with Rotation Forest and its variants.

– The comparison was performed on binary classification problems solely. Mutliclass and multi-label classification problems were not investigated. These can, however, be turned into binomial classifiers by a variety of strategies.

4 Discussion & Conclusion

We described an extensive empirical comparison between twenty prototypical supervised ensemble learning algorithms over nineteen UCI benchmark datasets with binary labels and examined the influence of two variants of decision tree inducers (unlimited depth, and extremely randomized tree) with and without calibration. The experiments presented here support the conclusion that the Rotation Forest family of algorithms (Rotb, RotbET, Rot and RotET) outperforms all other ensemble methods with or without calibration by a noticeable margin, which is much in line with the results obtained in [29]. It appears that the success of this approach is closely tied to its ability to simultaneously encourage diversity and individual accuracy via rotating the feature space and keeping all principal components. Not surprinsingly, the worse performing models are single decision trees, bagged trees, and AdaBoost Stump. Another conclusion we can draw from these observations is that building ensembles of extremely randomized trees is very competitive in terms of accuracy even for small sized data sets. This confirms the effectiveness of using random split thresholds, instead of optimized ones like in decision trees. We found calibration to be remarkably effective at lowering the RMS values of boosting-based methods.

References

1. Eric Bauer and Ron Kohavi. An empirical comparison of voting classification algorithms: Bagging, boosting, and variants. In *Machine Learning*, pages 105–139, 1998.
2. Amir Ben-Dor, Laurakay Bruhn, Agilent Laboratories, Nir Friedman, Miche'l Schummer, Iftach Nachman, U. Washington, U. Washington, and Zohar Yakhini. Tissue classification with gene expression profiles. *Journal of Computational Biology*, 7:559–584, 2000.
3. C.L Blake and C.J Merz. Uci repository of machine learning databases, 1998.
4. Leo Breiman. Bagging predictors. In *Machine Learning*, pages 123–140, 1996.
5. Leo Breiman. Bias, variance, and arcing classifiers. Technical report, 1996.
6. Leo Breiman. Randomizing outputs to increase prediction accuracy. *Machine Learning*, 40(3):229–242, 2000.
7. Leo Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.
8. Rich Caruana and Alexandru Niculescu-Mizil. Data mining in metric space: an empirical analysis of supervised learning performance criteria. In *KDD*, pages 69–78, 2004.
9. Rich Caruana and Alexandru Niculescu-Mizil. An empirical comparison of supervised learning algorithms. In *ICML*, pages 161–168, 2006.
10. Janez Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

11. Yoav Freund and Robert E. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting, 1997.
12. Jerome Friedman, Trevor Hastie, and Robert Tibshirani. Additive logistic regression: a statistical view of boosting. *Annals of Statistics*, 28:2000, 1998.
13. Pierre Geurts, Damien Ernst, and Louis Wehenkel. Extremely randomized trees. *Machine Learning*, 63(1):3–42, 2006.
14. T.R. Golub, Slonim, D.K., P. Tamayo, C. Huard, M. Gaasenbeek, J.P. Mesirov, and H. Coller. Molecular classication of cancer: Class discovery and class prediction by gene expression monitoring. *Science*, 286:531–537, 1999.
15. Daniel Hernández-Lobato, Gonzalo Martínez-Muñoz, and Alberto Suárez. How large should ensembles of classifiers be? *Pattern Recognition*, 46(5):1323–1336, 2013.
16. Tin Kam Ho. The random subspace method for constructing decision forests. *IEEE Trans. Pattern Anal. Mach. Intell.*, 20(8):832–844, 1998.
17. Kun hong Liu and De-Shuang Huang. Cancer classification using rotation forest. *Comp. in Bio. and Med.*, 38(5):601–610, 2008.
18. Eun Bae Kong and Thomas G. Dietterich. Error-correcting output coding corrects bias and variance. In *ICML*, pages 313–321, 1995.
19. Gilles Louppe and Pierre Geurts. Ensembles on random patches. In *ECML/PKDD (1)*, pages 346–361, 2012.
20. Dragos D. Margineantu and Thomas G. Dietterich. Pruning adaptive boosting. In *International Conference on Machine Learning (ICML)*, pages 211–218, 1997.
21. Gonzalo Martínez-Muñoz and Alberto Suárez. Switching class labels to generate classification ensembles. *Pattern Recognition*, 38(10):1483–1494, 2005.
22. Alexandru Niculescu-Mizil and Rich Caruana. Predicting good probabilities with supervised learning. In *ICML*, pages 625–632, 2005.
23. Juan José Rodríguez, Ludmila I. Kuncheva, and Carlos J. Alonso. Rotation forest: A new classifier ensemble method. *IEEE Trans. Pattern Anal. Mach. Intell.*, 28(10):1619–1630, 2006.
24. M. Schummer, W. V. Ng, and R. E. Bumgarnerd. Comparative hybridization of an array of 21,500 ovarian cdnas for the discovery of genes overexpressed in ovarian carcinomas. *Gene*, 238(2):375–385, 1999.
25. Friedhelm Schwenker. Ensemble methods: Foundations and algorithms [book review]. *IEEE Comp. Int. Mag.*, 8(1):77–79, 2013.
26. Pannagadatta K. Shivaswamy and Tony Jebara. Variance penalizing adaboost. In *NIPS*, pages 1908–1916, 2011.
27. Donna K. Slonim, Pablo Tamayo, Jill P. Mesirov, Todd R. Golub, Eric S. Lander, and Eric S. L. Class prediction and discovery using gene expression data. pages 263–272. Press, 2000.
28. Bianca Zadrozny and Charles Elkan. Obtaining calibrated probability estimates from decision trees and naive bayesian classifiers. In *ICML*, pages 609–616, 2001.
29. Chun-Xia Zhang and Jiang-She Zhang. Rotboost: A technique for combining rotation forest and adaboost. *Pattern Recognition Letters*, 29(10):1524–1536, 2008.
30. Zheng Zhao, Fred Morstatter, Shashvata Sharma, Salem Alelyani, and Aneeth Anand. Feature selection, 2011.