Combining Ensembles

Ulf Johansson*, Henrik Linusson†, Tuve Löfström†, Cecilia Sönströd†
*Department of Computer Science and Informatics, Jönköping University, Sweden
Email: ulf.johansson@ju.se
†Department of Information Technology
University of Borås, Sweden
Email: {henrik.linusson, tuve.lofstrom, cecilia.sonstrod}@hb.se

***Abstract*—Ensemble techniques are the standard choice for obtaining robust and high predictive performance. Two of the most popular ensemble techniques are random forests and bagged neural networks. Although often having similar accuracy, it is observed that a neural network ensemble and a random forest often disagree, i.e., produce different predictions, on a substantial number of instances, thus indicating an opportunity to further improve predictive performance by somehow actively selecting one of the ensembles for each prediction. In this study, several straightforward ways of utilizing two different ensembles are suggested and empirically evaluated, using 24 publicly available data sets. The results show that the simple method of selecting the ensemble which is most certain on its prediction achieved the best predictive performance, clearly outperforming both underlying ensembles.**

***Keywords*—Predictive modeling, Ensembles, Bagging, Random forest, Neural networks**

I. Introduction.

In the data mining task predictive modeling, the aim is to build a model capturing the relationship between an input vector $\mathbf{x}$ and a target concept y; in effect approximating the function $f(\mathbf{x}, y)$. When the target concept is a continuous variable, the task is called *regression*, while *classification* concerns categorical target variables, taking their values from a pre-determined set of class labels. For predictive models, the overriding concern is predictive performance, i.e., how well the model predicts the target value based on an input vector, for previously unseen examples. For classification, the simplest measure for this property is *accuracy*, which states the percentage of correct predictions on novel data instances. A wealth of machine learning techniques for producing classification models exist, ranging from transparent rule or decision tree models built using relatively simple algorithms, to opaque models such as artificial neural networks (ANNs) or support vector machines. Generally, opaque predictive models achieve better predictive performance than transparent models. It is firmly established in machine learning research (see e.g., [1] or [2]), that predictive performance can be further improved by combining several (different) models into an *ensemble*. In an ensemble, multiple *base models*, for classification called *base classifiers*, are combined into a composite model, where the ensemble prediction is a function of all included base model predictions. The basic idea behind this scheme is that a collection of models, which make their errors on different instances, will perform better than any single one of the models, since aggregating several models, using averaging or majority voting, will eliminate uncorrelated base model errors; see e.g. [3]. Based on their robustly high predictive performance, ensembles models, such as *random forests* [4] constitute the state-of-the-art in predictive modeling and are available in most commercial data analysis packages. Studying how errors are distributed between base classifiers in an ensemble, it becomes apparent that some instances are intrinsically difficult to predict, i.e., almost all base classifiers in the ensemble get them wrong. However, for a substantial proportion of ensemble errors, a number of base classifiers actually make a correct prediction. This reasoning can also be applied when considering an arbitrary pair of predictive models; the fact that they have similar predictive performance does not entail that they make their errors on the same instances. This implies that there is a possibility to obtain a correct prediction, if one could only choose which model to trust. Obviously, for this to be workable in practice, model predictions must be combined automatically to produce a single prediction for each instance.

The purpose of this study is to conduct a thorough empirical investigation of how the above observation, regarding differences on the instance level, between two models with similar predictive performance, can be exploited to increase overall predictive performance. In short, two different ensemble classifiers, both highly accurate and with similar predictive performance over a large number of data sets, are built using off-the-shelf techniques. Based on observations of model characteristics regarding errors and error distributions, different ways of combining ensemble predictions are empirically evaluated over a large number of data sets.

II. Background.

Simply put, an ensemble is constructed by training a set of L base classifiers and combining their predictions. When performing classification, the predictions of the base classifiers can be combined using, e.g., majority voting. For the ensemble to be effective, the base classifiers must commit their errors on different instances, which is the informal meaning of the term *diversity*. Arguably the simplest and most intuitive diversity measure, *disagreement* calculates the disagreement between all pairs of classifiers as the ratio between the number of instances on which one classifier is correct and the other incorrect and the total number of instances [5]:

$$\mathrm{D}_{j,k} = \frac{N^{01} + N^{10}}{N} \quad (1)$$

This work was supported by the Swedish Foundation for Strategic Research through the project High-Performance Data Mining for Drug Effect Detection (IIS11-0053) and the Knowledge Foundation through the project Data Analytics for Research and Development (20150185).

where N^{01} and N^{10} measure the number of instances on which the two models j and k are correct (1) and incorrect (0). N is the total number of instances. The disagreement measure is the average over all pairs:

$$Dis = \frac{2}{L(L-1)} \sum_{j=1}^{L-1} \sum_{k=j+1}^{L} D_{j,k} \tag{2}$$

Brown and Kuncheva showed that the ensemble error when using majority voting can be decomposed into the average base classifier error and diversity [6]. The diversity of a base classifier $h_l(x) \in \{1, -1\}$ is, for this decomposition, defined as the disagreement between the classifier and the ensemble prediction $H(x) \in \{1, -1\}$: $d_l(x) = \frac{1}{2}(1 - h_l(x)H(x))$.

The decomposition of the ensemble error e_{maj} into the average error of the base classifiers e_{ind} and the diversity d_l for an individual instance is:

$$e_{\text{maj}}(x) = e_{\text{ind}}(x) - yH(x)\frac{1}{L}\sum_{l=1}^{L} d_l(x) \tag{3}$$

where $y \in Y = \{1, -1\}$. It is evident that the ensemble error is composed of the average base classifier error and a combination of the true target value, the ensemble prediction and the average diversity among the base classifiers. The major difference between the decomposition for majority voting and a similar decomposition for regression ensembles [7] is the inclusion of the true target value in equation (3).

Equation (3) calculates the majority vote error of a single instance. To calculate the majority vote error over all instances, E_{maj}, taking advantage of the fact that $yH(x) = 1$ when correct and $yH(x) = -1$ when incorrect, the integration with respect to the probability density function becomes

$$\begin{aligned} E_{\text{maj}} &= \int_x e_{\text{ind}}(x) - \int_x yH(x)\frac{1}{L}\sum_{l=1}^{L} d_l(x) \\ &= \int_x e_{\text{ind}}(x) \\ &\quad - \underbrace{\int_{x+} \frac{1}{L}\sum_{l=1}^{L} d_l(x)}_{\text{good diversity}} \\ &\quad + \underbrace{\int_{x-} \frac{1}{L}\sum_{l=1}^{L} d_l(x)}_{\text{bad diversity}} \end{aligned} \tag{4}$$

where $x+$ refers to instances on which the ensemble is correct and $x-$ refers to instances on which the ensemble is wrong. What the decomposition essentially shows is that increasing the disagreement is beneficial for instances where the ensemble is correct and detrimental for instances where the ensemble is wrong, hence the labels 'good' and 'bad' diversity.

Brown et al. [8] introduced a taxonomy of methods for creating diversity. The most important distinction made is between explicit methods, where some metric of diversity is directly optimized, and implicit methods, where the method is likely to produce diversity without actually targeting it. A common procedure in many explicit methods is to first train a set of classifiers and then select, based on some diversity metric, a subset of classifiers to combine when predicting [9]. The selection can either be static, when the same subset is always used, or dynamic, when a new subset is selected for each new instance. However, several studies have shown that diversity is not likely to be useful as the optimization criterion for this procedure [10]–[12].

Bagging [13] is an implicit method which achieves diversity by training each base classifier using different emulated training sets obtained using resampling with replacement. Each training set (called a *bootstrap*) generally consists of the same number of instances as the entire data set available for training, with the result that approximately 63.2 % of available instances are included in each bootstrap. Since only a subset of the instances are used to train each classifier, there is always a proportion of the available instances, called *out-of-bag* (OOB) instances, not used to train a classifier. By forming an ensemble composed of only the classifiers for which an instance is out-of-bag, it is possible to get an unbiased estimate of the bagging ensemble's performance, while still using all instances for training. Since the out-of-bag ensemble is approximately $1/3$ of the size of the entire bagging ensemble, the out-of-bag error estimate tends to overestimate the actual error made by an ensemble, simply because a larger ensemble is normally a stronger model.

Implicitly targeting diversity is inherent in random forests. Random forests consist of decision trees and achieve their diversity by introducing randomness in both instance and feature selection; bagging is used to select which instances to use when training each tree and each split is based on only a randomized subset of the attributes. Similarly to standard bagging, out-of-bag estimates can be used to estimate the predictive performance of a random forest.

Diversity can also be implicitly targeted when constructing ensembles of neural networks. In an evaluation of different ways of implicitly creating diversity in ensembles of neural networks [14], it was shown that bagging is a reliable method for consistently achieving diversity.

Since ensembles of neural networks and random forests are both state-of-the-art techniques with high predictive performance, it might be expected that two of these ensemble models would agree to a large degree in their predictions. To explore how they perform in relation to each other, Figs. 1-3 show the proportion of base classifiers correctly predicting each instance, for a sample data set. In Fig. 1, all instances are plotted for both random forests and ensembles of neural networks, sorted on the proportion of base classifiers being correct. In Fig. 2, only instances on which the opposing ensemble is correct are plotted. This means that only instances on which the random forest (RF) is correct are plotted in the plot with the ensemble of neural networks (Ens) and vice versa. Fig. 2 plots only instances on which a majority of the base classifiers of the opposing ensemble are incorrect.

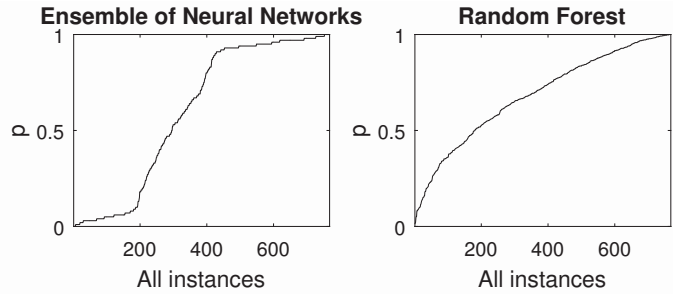

Fig. 1. Proportions of models correctly predicting instances on the Liver data set. The left plot shows the proportions for the ANN-ensemble and the right plot for the random forest. Instances are sorted on the proportion of the models being correct.

It is evident from Fig 1 that random forests and ensembles of neural networks, while being accurate to approximately the same degree, distribute their base classifiers errors quite differently. The neural networks tend, to a much larger degree than the trees in the random forests, to agree in their predictions. In other words, the neural networks are less diverse but more accurate than the trees, on average. The question remains, however, if the models agree on individual instances.

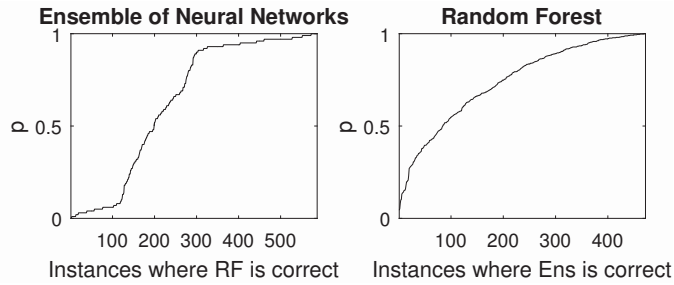

Fig. 2. The same setup as in Fig. 1 with the difference that only instances correctly predicted by the opposing ensemble are included.

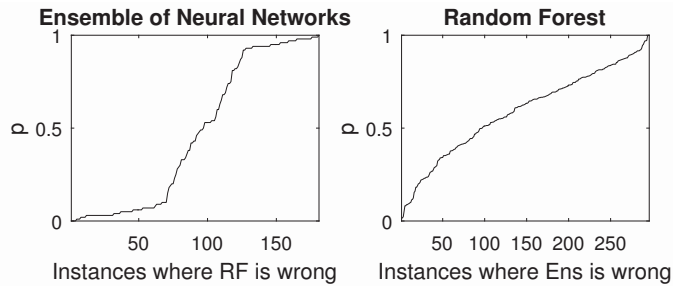

Fig. 3. The same setup as in Fig. 1 with the difference that only instances incorrectly predicted by the opposing ensemble are included.

As is clearly seen in Figs. 2 and 3, the proportion of instances on which these two highly accurate ensemble models disagree is fairly large. As expected, there are obviously many instances on which both models agree but there are also many instances on which they strongly disagree. The neural networks in particular strongly agree on the correct prediction on many instances that the random forest has failed to predict correctly. The decision trees in the random forest, on the other hand, agree to a much lower degree on the instances the ensemble of neural networks had failed to predict correctly. See Table I for results on the performance of individual models on all data sets.

A. Related work.

The goal of finding the most suitable learning algorithm, be it in general, with regard to a specific decision problem, or with regard to specific subspaces of a certain problem domain, is central to the predictive modeling task. Early approaches, still commonly used today, include the cross-validation [15], [16], generalized cross-validation [17] and bootstrapping [15] methods. These winner-takes-all approaches allow an analyst to select, from a pool of candidate algorithms, the most accurate one, either for a specific classification problem or for a set of classification problems.

The idea of stacked generalization [18]–[20] builds on the cross-validation method, but, rather than selecting the single most accurate learning algorithm, a composite ensemble model is created using a combination function. The goal of stacking learners is, of course, to create a composite model that is more accurate than the models created by any of the individual candidate learning algorithms. Closely related to the idea of stacking is the field of classifier fusion, in which heterogeneous ensemble models (containing models induced from different learning algorithms) are found through different combination methods [21]–[25].

Another related field of study is the overproduce-and-select scheme sometimes used in ensemble learning [9]. Here, a large pool of learners is first created, either by inducing, with randomness, multiple models using the same learning algorithm, e.g., [26], or by utilizing multiple learning algorithms, e.g., [24] (the overproduction step). Second, rather than utilizing the entire pool of learners as an ensemble model (using, e.g., voting), a subset of the candidate models are selected, using some heuristic, in order to attempt to construct a sub-ensemble more accurate than the model represented by the full model pool (the selection step).

All methods described thus far are *static*, i.e., once a certain model (ensemble or single model) has been selected, that model is used to predict the outputs of all future test patterns. With the dynamic overproduce-and-select scheme [27] (often called *dynamic ensemble selection*), sub-ensembles are selected on-line to predict the output of each test pattern, typically based on some characteristics of the particular test pattern or the (sub-)ensemble predictions. Examples include estimating and optimizing sub-ensemble performance locally in an explicit manner, i.e., by assessing the performance of ensemble members using the nearest-neighbor rule (based on performance on training or validation data) and creating an ensemble from locally well-performing models [28]–[30], as well as selecting sub-ensembles that are confident in their predictions, i.e., selecting sub-ensembles where there is little discrepancy amongst the voting members [31].

III. Method.

As described in the introduction, the overall purpose is to investigate whether it is possible to increase predictive performance by using two separate ensembles with different inductive biases. As an alternative to the plethora of highly

technical and very specialized ensemble schemes, we in this paper evaluate only straightforward approaches.

For this purpose, we start out with one ensemble of bagged neural nets and one random forest. For the neural network ensemble, we use 100 base classifiers, while the random forest has 500 trees. We believe these to be reasonable choices, specifically because the ensembles are large enough to allow for accurate out-of-bag estimates. In addition, adding more models would probably only lead to marginal improvements, making it interesting to look for other solutions; in other words, had we used significantly smaller ensembles, the most obvious method to increase accuracy would have been to simply add more base classifiers.

In the experimentation, a total of eight different setups are evaluated:

- **ANN**: The ensemble consisting of 100 bagged ANNs.
- **RF**: The random forest with 500 trees.
- **Prob**: The probability estimates of each model are used in this setup. When classifying a test instance, the ensemble with the highest probability estimate is used, i.e., if the two ensembles disagree, the one that is most certain is used. This is in fact very similar to averaging all models of both ensembles, but using a weighting of 5 for each ANN base classifier.
- **Rank**: This setup is identical to *Prob*, with one exception. Since ANN ensembles tend to be less diverse, there is a risk that *Prob* might favor the ANN ensemble. With this in mind, this setup instead first ranks all test instances, according to how certain the ensemble is, based on the probability estimates. This ranking gives the lowest rank to the highest probability estimate, and it is performed individually for both ensembles. When predicting a test instance, the ensemble with the lowest rank for this particular test instance is used, with the motivation that the instance is considered to be relatively easier for that ensemble. It must be noted that this setup requires a set of test instances, i.e., it can not be used on individual, e.g., streaming, test instances.
- **OOB-g**: Here the ensemble with the highest out-of-bag accuracy on the training set (on each fold) is used for predicting all test instances.
- **OOB-l**: In this setup, which ensemble to use is determined for each test instance individually. More specifically, the out-of-bag error over the ten neighboring instances is calculated, and the ensemble with the lowest error is used for predicting the test instance. If there is a tie, the technique with the lowest global out-of-bag error is used.
- **Gated**: Here, an additional feed-forward ANN (one hidden layer with ten units) is trained to learn the difference in out-of-bag errors for the two ensembles, using the training instances as inputs. When predicting a test instance, this additional ANN is used to determine which of the two ensembles that should be used. Naturally, if the ANN predicts the random forest to be the most accurate, the random forest is used for that test instance, and vice versa.
- **Tie-breaker**: In this setup too, an additional ANN is used. But here, the ANN is trained on only those training instances where the two ensembles disagree in their out-of-bag predictions. When predicting a test instance, this additional model is used for the actual prediction when the two ensembles output different classes.

All experimentation was performed in MatLab, in particular using the Neural network and the Statistics tool boxes. For the ensemble models, the functions *patternnet* and *treebagger* were used for neural networks and random forests, respectively. All ANNs had one hidden layer and, in order to introduce some additional diversity, the number of hidden units was, for each ANN, randomized from the uniform distribution $[\lfloor \frac{a}{2} \rfloor, a]$, where a is the number of attributes. As mentioned above, the number of random trees was set to 500, and the number of ANNs trained was 100. For the network training, resilient backpropagation (*rprop* in MatLab) was used. All other parameter values were left at their default values.

In all 24 publicly available data sets are used in the experimentation. All data sets are from the UCI [32] or KEEL [33] repositories. For the evaluation, predictive performance is measured using both accuracy and area-under-the-ROC-curve (AUC). While accuracy is based only on the classifications, AUC measures the ability to rank instances according to how likely they are to belong to a certain class; see e.g., [34]. AUC can be interpreted as the probability of ranking a true positive instance ahead of a false positive; see [35]. Since standard 10-fold cross-validation was used, all reported performance measures are averaged over the 10 folds.

IV. Results.

Table I below shows the results for the two ensembles. Looking first at ensemble accuracies, the mean results, when averaged over all data sets are very similar. In a head-to-head comparison, the random forest wins 13 data sets and the bagged ANNs 9. We see that individual neural networks are substantially more accurate than the corresponding random trees but, on the other hand, that diversity, measured as disagreement (*Dis*), is much higher in the random forest. The last column shows the proportion of all test instances where the two ensembles predict different class labels. Interestingly enough, the two ensembles disagree on almost 10% of all instances. For individual data sets, this number can be much higher, sometimes 20% or more.

TABLE I. ENSEMBLE RESULTS

	ANN			RF			
	ensAcc	bAcc	Dis	ensAcc	bAcc	Dis	Diff
colic	.832	.787	.165	.846	.723	.308	.059
creditA	.861	.845	.101	.880	.780	.243	.057
diabetes	.772	.758	.120	.764	.687	.310	.112
german	.691	.663	.209	.667	.609	.355	.108
haberman	.710	.721	.097	.686	.650	.291	.223
heartC	.841	.800	.149	.821	.732	.281	.066
heartH	.850	.811	.138	.823	.752	.245	.068
heartS	.826	.788	.150	.826	.726	.286	.074
hepati	.845	.810	.143	.845	.781	.235	.090
iono	.909	.885	.087	.934	.871	.156	.066
je4042	.722	.677	.242	.756	.675	.296	.204
je4243	.631	.595	.270	.656	.598	.368	.273
kc1	.758	.751	.060	.765	.673	.306	.104
kc2	.810	.787	.093	.789	.731	.245	.103
liver	.674	.609	.316	.733	.609	.395	.199
mw	.923	.914	.044	.910	.872	.119	.029
pc1req	.731	.640	.323	.683	.624	.326	.144
pc4	.900	.890	.047	.901	.854	.164	.050
sonar	.813	.771	.198	.832	.687	.369	.163
spect	.881	.844	.119	.885	.803	.199	.005
spectf	.824	.786	.164	.813	.733	.285	.094
ttt	.983	.976	.023	.991	.831	.254	.009
wbc	.955	.939	.039	.961	.910	.099	.019
vote	.867	.836	.139	.874	.799	.211	.054
Mean	**.815**	**.785**	**.140**	**.816**	**.735**	**.266**	**.099**

Table II below shows the accuracy results for the suggested ways of combining ensemble predictions, with ensemble accuracies from Table I repeated for comparison.

TABLE II. ACCURACY RESULTS FOR ENSEMBLE COMBINATIONS

	ANN	RF	Prob	Rank	OOB-g	OOB-l	Gated	Tie-br.
colic	.832	.846	.838	.838	.829	.818	.843	.832
creditA	.861	.880	.870	.867	.875	.878	.872	.867
diabetes	.772	.764	.777	.773	.770	.773	.764	.766
german	.691	.667	.679	.677	.691	.689	.663	.698
haberman	.710	.686	.707	.703	.710	.689	.700	.703
heartC	.841	.821	.848	.844	.838	.838	.841	.834
heartH	.850	.823	.840	.840	.829	.826	.843	.843
heartS	.826	.826	.822	.837	.807	.811	.819	.822
hepati	.845	.845	.858	.845	.839	.826	.832	.858
iono	.909	.934	.934	.934	.934	.929	.931	.903
je4042	.722	.756	.744	.744	.756	.733	.756	.722
je4243	.631	.656	.672	.672	.656	.669	.653	.661
kc1	.758	.765	.768	.770	.758	.758	.763	.755
kc2	.810	.789	.808	.802	.808	.797	.786	.802
liver	.674	.733	.739	.716	.724	.721	.739	.748
mw	.923	.910	.921	.918	.921	.921	.913	.921
pc1req	.731	.683	.750	.721	.731	.692	.654	.692
pc4	.900	.901	.908	.909	.899	.901	.908	.897
sonar	.813	.832	.832	.832	.803	.808	.827	.832
spect	.881	.885	.881	.885	.885	.885	.885	.881
spectf	.824	.813	.816	.820	.824	.816	.809	.794
ttt	.983	.991	.986	.991	.991	.991	.984	.982
wbc	.955	.961	.955	.961	.955	.957	.955	.952
vote	.867	.874	.870	.872	.874	.876	.870	.870
Mean	**.815**	**.816**	**.824**	**.822**	**.819**	**.814**	**.815**	**.816**
Rank	**4.81**	**4.52**	**3.42**	**3.46**	**4.29**	**4.94**	**5.21**	**5.35**

Here, it is seen that only three of the suggested combination methods produce higher mean accuracies than the random forest ensemble. Interestingly, it is in fact the most straightforward setups, i.e., *Prob*, *Rank* and *OOB-g* that all achieve higher accuracy, averaged over all data sets, than any of the underlying ensemble models obtain on their own. The three most elaborate schemes for combining ensemble predictions, i.e., *OOB-l*, *Gated* and *Tie-breaker*, on the other hand, all fail to improve overall accuracy. The highest mean accuracy is actually obtained by the simplest method (*Prob*), which performs better than both underlying ensembles on eight data sets, and worse than both of them on only a single data set. The mean ranks reflect the accuracy results, with the top three methods accuracy-wise having the lowest three means ranks.

In order to determine any statistically significant differences, we used the procedure recommended in [36] and performed a Friedman test [37], followed by Bergmann-Hommel's [38] dynamic procedure to establish all pairwise differences. Unfortunately, although the Friedman test shows that there are statistically significant differences, the n vs. n *post-hoc* test does not reveal any pairwise differences. One explanation is the fact that we evaluate as many as eight techniques on relatively few data sets. With this in mind, we also performed a 1 vs. n *post-hoc* test, treating *Prob* as the control. Here Hochberg's procedure [39] was used, and the resulting adjusted p-values are shown in Table III below. According to this test, *Prob* is indeed significantly more accurate (for $\alpha = 0.05$) than *Tie-breaker*, but no other differences are statistically significant.

TABLE III. ADJUSTED p-VALUES FOR ACCURACY RESULTS. *Prob* VS. OTHER TECHNIQUES USING HOCHBERG'S PROCEDURE.

Technique	p-value
Tie-br.	0.043
Gated	0.068
OOB-l	0.169
ANN	0.194
RF	0.355
OOB-g	0.410
Rank	0.953

Table IV below shows the ranking ability of the different techniques, given as AUC.

TABLE IV. AUC RESULTS

	ANN	RF	Prob	Rank	OOB-g	OOB-l	Gated	Tie-br.
colic	.879	.882	.881	.880	.882	.881	.877	.877
creditA	.923	.933	.933	.924	.932	.932	.931	.933
diabetes	.840	.829	.843	.836	.838	.835	.833	.836
german	.633	.625	.638	.632	.633	.640	.617	.638
haberman	.692	.674	.686	.701	.692	.684	.683	.686
heartC	.911	.905	.910	.914	.908	.908	.915	.895
heartH	.914	.896	.912	.903	.905	.900	.905	.912
heartS	.892	.894	.903	.904	.888	.891	.894	.903
hepati	.881	.900	.894	.904	.886	.883	.869	.887
iono	.974	.982	.984	.983	.982	.982	.978	.976
je4042	.786	.809	.815	.805	.809	.804	.812	.815
je4243	.676	.712	.723	.718	.712	.711	.705	.710
kc1	.684	.713	.720	.698	.698	.695	.707	.718
kc2	.839	.835	.848	.842	.834	.834	.842	.848
liver	.728	.754	.769	.744	.745	.740	.761	.764
mw	.797	.841	.838	.829	.802	.802	.786	.838
pc1req	.740	.743	.755	.750	.740	.708	.697	.730
pc4	.906	.947	.946	.930	.933	.937	.945	.946
sonar	.914	.957	.940	.919	.948	.945	.929	.915
spect	.699	.699	.703	.715	.699	.699	.715	.703
spectf	.852	.839	.852	.848	.852	.845	.838	.852
ttt	.999	1.000	.999	.996	1.000	1.000	.998	.999
wbc	.983	.980	.981	.984	.982	.983	.984	.978
vote	.924	.917	.927	.921	.914	.916	.918	.927
Mean	**.830**	**.839**	**.845**	**.839**	**.836**	**.834**	**.834**	**.840**
Rank	**5.63**	**4.44**	**2.60**	**4.08**	**4.56**	**5.25**	**5.38**	**4.06**

For AUC, the mean results differ slightly from the mean accuracies obtained. First of all, it should be noted that the ANN ensemble performs much worse on mean AUC than the random forest. Turning to the proposed combination methods, *Prob* once again performs best, and is the clear winner, both on mean AUC and on mean ranks. No other proposed combination technique outperforms random forests on mean AUC, although *Rank* and *Tie-breaker* achieve slightly lower mean ranks. For the AUC results, the differences are substantial enough for the n vs. n Bergmann-Hommel's *post-hoc* test to identify that *Prob* obtained significantly higher AUC than *ANN*, *OOB-l* and *Gated*. Using Hochberg's procedure for a 1 vs. n *post hoc* test, *Prob* is, in fact, seen to be significantly better than all other techniques, shown by the modified p-values in Table V below.

TABLE V. ADJUSTED p-VALUES FOR AUC RESULTS. *Prob* VS. OTHER TECHNIQUES USING HOCHBERG'S PROCEDURE.

Technique	p-value
ANN	1.4E-04
Gated	5.3E-04
OOB-l	9.1E-04
OOB-g	0.022
RF	0.029
Rank	0.039
Tie-br.	0.039

The overall result regarding predictive performance is thus that the most straightforward combination method, *Prob*, emerges as the clear winner, having both the highest mean accuracy and AUC and lowest mean rank of all setups. This implies that a simple combination procedure is available for increasing predictive performance in ensemble learning; build two highly accurate ensembles with different techniques and utilize their probability estimates to select the predicting ensemble for each new test instance. None of the more complicated setups perform consistently well, as seen by e.g. *Tie-breaker* having the second best results on AUC, but being worst on accuracy ranks.

In order to gain insights into how the suggested setups function in practice, it is useful to investigate how the two ensemble models are used to perform the actual classifications. Table VI below shows the proportion of instances predicted by the random forest, for each data set.

TABLE VI. PROPORTION OF THE PREDICTIONS MADE BY THE RANDOM FOREST

	Prob	Rank	OOB-g	OOB-l	Gated
colic	.213	.510	.697	.627	.437
creditA	.348	.504	.900	.803	.423
diabetes	.573	.525	.201	.310	.503
german	.591	.553	.000	.158	.467
haberman	.647	.523	.000	.131	.643
heartC	.301	.520	.199	.315	.325
heartH	.485	.560	.201	.259	.348
heartS	.333	.552	.400	.430	.281
hepati	.413	.594	.606	.542	.510
iono	.560	.563	1.000	.897	.546
je4042	.778	.522	1.000	.837	.881
je4243	.738	.474	1.000	.658	.865
kc1	.624	.542	.451	.442	.857
kc2	.672	.561	.100	.209	.797
liver	.748	.519	.900	.651	.924
mw	.660	.551	.100	.111	.478
pc1req	.731	.596	.000	.250	.663
pc4	.658	.519	.700	.640	.649
sonar	.135	.553	.803	.663	.438
spect	.560	.592	1.000	.986	.390
spectf	.356	.543	.000	.184	.363
ttt	.005	.529	1.000	.994	.054
wbc	.585	.553	.400	.400	.335
vote	.375	.497	.899	.805	.337
Mean	**.508**	**.540**	**.520**	**.510**	**.535**

The immediate impression from the mean values is that the two ensembles contribute more or less equally to the final prediction, albeit with a slight bias towards the random forest. Since this ensemble has slightly better predictive performance than the bagged ANNs, this is an encouraging indication that the basic idea works in practice. Closer inspection of the results does, however, reveal dramatic variations, both within each setup and each data set. Strikingly, for some data sets, e.g., *ttt*, the proportion of instances classified by the random forest can vary over the different setups from almost 0% to 100%. Similarly, for specific setups like *OOB-g*, the variation between data sets is also from zero to all instances. Again, this is an indication that the different setups successfully select the ensemble most suited to perform the prediction. Finally, it is noteworthy that *Rank* is the most stable setup, and also has the highest mean proportion of instances predicted by the random forest.

V. CONCLUDING REMARKS.

We have in this paper evaluated procedures combining two ensembles, one random forest and one ensemble of bagged neural networks, for predictive classification. The underlying assumption is that although both ensembles have similar and very high accuracy, they still disagree on a fairly large proportion of all test instances. With this in mind, it should be possible to increase the accuracy, compared to using either of the ensembles, by somehow combining them. In the experimentation, a number of straightforward approaches was evaluated, and the results show that it is indeed possible to devise combination strategies capable of outperforming the standard ensembles. The best combination strategy turned out to be one of the most straightforward, i.e., simply using the ensemble that is the most certain, for each test instance. This strategy obtained the best ranking for both accuracy and AUC, with the differences to all other methods being statistically significant for AUC.

REFERENCES

[1] T. G. Dietterich, "Ensemble methods in machine learning," in *Multiple Classifier Systems*, 2000, pp. 1–15.

[2] D. W. Opitz and R. Maclin, "Popular ensemble methods: An empirical study," *J. Artif. Intell. Res. (JAIR)*, vol. 11, pp. 169–198, 1999.

[3] T. G. Dietterich, "Machine-learning research: Four current directions," *The AI Magazine*, vol. 18, no. 4, pp. 97–136, 1998.

[4] L. Breiman, "Random forests," *Machine Learning*, vol. 45, no. 1, pp. 5–32, 2001.

[5] T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Trans. Pattern Anal. Mach. Intell.*, vol. 20, no. 8, pp. 832–844, 1998.

[6] G. Brown and L. I. Kuncheva, ""good" and "bad" diversity in majority vote ensembles," in *Multiple Classifier Systems*. Springer, 2010, pp. 124–133.

[7] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," *Advances in Neural Information Processing Systems*, vol. 2, pp. 231–238, 1995.

[8] G. Brown, J. Wyatt, R. Harris, and X. Yao, "Diversity creation methods: a survey and categorisation," *Information Fusion*, vol. 6, no. 1, pp. 5–20, 2005.

[9] L. I. Kuncheva, "That elusive diversity in classifier ensembles," in *Pattern Recognition and Image Analysis*. Springer, 2003, pp. 1126–1138.

[10] L. Saitta, "Hypothesis diversity in ensemble classification," in *Foundations of Intelligent Systems*. Springer, 2006, pp. 662–670.

[11] E. K. Tang, P. N. Suganthan, and X. Yao, "An analysis of diversity measures," *Machine Learning*, vol. 65, no. 1, pp. 247–271, 2006.

[12] U. Johansson, T. Lofstrom, and H. Bostrom, "Overproduce-and-select: The grim reality," in *Computational Intelligence and Ensemble Learning (CIEL), 2013 IEEE Symposium on*. IEEE, 2013, pp. 52–59.

[13] L. Breiman, "Bagging predictors," *Machine learning*, vol. 24, no. 2, pp. 123–140, 1996.

[14] U. Johansson and T. Lofstrom, "Producing implicit diversity in ann ensembles," in *Neural Networks (IJCNN), The 2012 International Joint Conference on*. IEEE, 2012, pp. 1–8.

[15] B. Efron, "Computers and the theory of statistics: thinking the unthinkable," *SIAM review*, vol. 21, no. 4, pp. 460–480, 1979.

[16] M. Stone, "An asymptotic equivalence of choice of model by cross-validation and akaike's criterion," *Journal of the Royal Statistical Society. Series B (Methodological)*, pp. 44–47, 1977.

[17] K.-C. Li, "From stein's unbiased risk estimates to the method of generalized cross validation," *The Annals of Statistics*, pp. 1352–1377, 1985.

[18] D. H. Wolpert, "Stacked generalization," *Neural networks*, vol. 5, no. 2, pp. 241–259, 1992.

[19] S. Džeroski and B. Ženko, "Is combining classifiers with stacking better than selecting the best one?" *Machine learning*, vol. 54, no. 3, pp. 255–273, 2004.

[20] K. M. Ting and I. H. Witten, "Issues in stacked generalization," *J. Artif. Intell. Res.(JAIR)*, vol. 10, pp. 271–289, 1999.

[21] G. Tsoumakas, I. Katakis, and I. Vlahavas, "Effective voting of heterogeneous classifiers," in *Machine Learning: ECML 2004*. Springer, 2004, pp. 465–476.

[22] G. Tsoumakas, L. Angelis, and I. Vlahavas, "Selective fusion of heterogeneous classifiers," *Intelligent Data Analysis*, vol. 9, no. 6, pp. 511–525, 2005.

[23] M. Woźniak, M. Graña, and E. Corchado, "A survey of multiple classifier systems as hybrid systems," *Information Fusion*, vol. 16, pp. 3 – 17, 2014.

[24] R. Caruana, A. Niculescu-Mizil, G. Crew, and A. Ksikes, "Ensemble selection from libraries of models," in *Proceedings of the twenty-first international conference on Machine learning*. ACM, 2004, p. 18.

[25] D. Ruta and B. Gabrys, "Classifier selection for majority voting," *Information fusion*, vol. 6, no. 1, pp. 63–81, 2005.

[26] G. Giacinto and F. Roli, "Design of effective neural network ensembles for image classification purposes," *Image and Vision Computing*, vol. 19, no. 9, pp. 699–707, 2001.

[27] A. S. Britto, R. Sabourin, and L. E. Oliveira, "Dynamic selection of classifiers—a comprehensive review," *Pattern Recognition*, vol. 47, no. 11, pp. 3665–3680, 2014.

[28] G. Giacinto and F. Roli, "Dynamic classifier selection based on multiple classifier behaviour," *Pattern Recognition*, vol. 34, no. 9, pp. 1879–1881, 2001.

[29] L. Didaci, G. Giacinto, F. Roli, and G. L. Marcialis, "A study on the performances of dynamic classifier selection based on local accuracy estimation," *Pattern Recognition*, vol. 38, no. 11, pp. 2188–2191, 2005.

[30] A. H. Ko, R. Sabourin, and A. S. Britto Jr, "From dynamic classifier selection to dynamic ensemble selection," *Pattern Recognition*, vol. 41, no. 5, pp. 1718–1731, 2008.

[31] E. M. Dos Santos, R. Sabourin, and P. Maupin, "A dynamic overproduce-and-choose strategy for the selection of classifier ensembles," *Pattern Recognition*, vol. 41, no. 10, pp. 2993–3009, 2008.

[32] K. Bache and M. Lichman, "UCI machine learning repository," 2013. [Online]. Available: http://archive.ics.uci.edu/ml

[33] J. Alcalá-Fdez, A. Fernández, J. Luengo, J. Derrac, and S. García, "Keel data-mining software tool: Data set repository, integration of algorithms and experimental analysis framework," *Multiple-Valued Logic and Soft Computing*, vol. 17, no. 2-3, pp. 255–287, 2011.

[34] T. Fawcett, "An introduction to roc analysis," *Pattern Recognition Letters*, vol. 27, no. 8, pp. 861–874, 2006.

[35] A. P. Bradley, "The use of the area under the roc curve in the evaluation of machine learning algorithms," *Pattern Recognition*, vol. 30, no. 7, pp. 1145–1159, 1997.

[36] S. Garcıa and F. Herrera, "An extension on statistical comparisons of classifiers over multiple data sets for all pairwise comparisons," *Journal of Machine Learning Research*, vol. 9, no. 2677-2694, p. 66, 2008.

[37] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *Journal of American Statistical Association*, vol. 32, pp. 675–701, 1937.

[38] B. Bergmann and G. Hommel, "Improvements of general multiple test procedures for redundant systems of hypotheses," in *Multiple Hypotheses Testing*. Springer, 1988, pp. 100–115.

[39] Y. Hochberg, "A Sharper Bonferroni Procedure for Multiple Tests of Significance," *Biometrika*, vol. 75, no. 4, pp. 800–802, 1988.