

A New Ensemble Diversity Measure Applied to Thinning Ensembles

Robert E. Banfield¹, Lawrence O. Hall¹, Kevin W. Bowyer², W. Philip Kegelmeyer³

¹Department of Computer Science and Engineering, University of South Florida
4202 E. Fowler Ave, Tampa, Florida 33620, USA
{rbanfiel,hall}@csee.usf.edu

²Department of Computer Science and Engineering, University of Notre Dame
384 Fitzpatrick Hall, Notre Dame, IN 46556, USA
kwb@cse.nd.edu

³Sandia National Labs, Biosystems Research Department, PO Box 969, MS 9951
Livermore, CA 94551-0969, USA
wpk@ca.sandia.gov

Abstract. We introduce a new way of describing the diversity of an ensemble of classifiers, the Percentage Correct Diversity Measure, and compare it against existing methods. We then introduce two new methods for removing classifiers from an ensemble based on diversity calculations. Empirical results for twelve datasets from the UC Irvine repository show that diversity is generally modeled by our measure and ensembles can be made smaller without loss in accuracy.

1 Introduction

Multiple classifier systems have become the subject of attention because they can provide significant boosts in accuracy for many datasets [1-6]. They can be created in a variety of ways. Classifiers of the same type that perform differently may be created by modifying the training set through randomly re-sampling with replacement, as in bagging [2], or successively choosing training sets based on errors made by the previous set of classifiers, as in Ivoting [3]. They can also be created by approaches that exploit randomization within the learner; for example, different initial random weights in neural networks or a random choice of the test at a node in a decision tree from among the top n choices [5]. The boost in performance from using an ensemble is at least partially due to diversity [4,7] – examples that are incorrectly classified by some classifiers are correctly classified by others, in such a way that the voted accuracy is greater than that of any single classifier. This paper considers the concept of diversity, develops a new approach to describing it, and then uses this to create a better ensemble by removing the less useful classifiers.

2 Diversity

Diversity is a property of an ensemble of classifiers with respect to a set of data. Diversity is greater when, all other factors being equal, the classifiers that make incorrect decisions for a given example spread their decisions more evenly over the possible incorrect decisions. The more uniformly distributed the errors are, the greater the diversity, and vice versa.

The Kappa statistic from Dietterich [4], which measures the degree of similarity between two classifiers, serves as an illustrative starting point for examining diversity. Referring to Figure 1, the Kappa value can be plotted on the x-axis for each pair of classifiers, against the mean error for the pair on the y-axis. A broad scatter of points along the x-axis indicates that the pairs of classifiers have significantly different levels of agreement. Ideally, the best classifiers would be both individually accurate and comparatively diverse. The average of all paired Kappa values can be used as a measure of ensemble diversity. One of the drawbacks of using Kappa diagrams is the computational complexity associated with calculating all of the pair-wise combinations which would be required to generate a single number for the overall diversity.

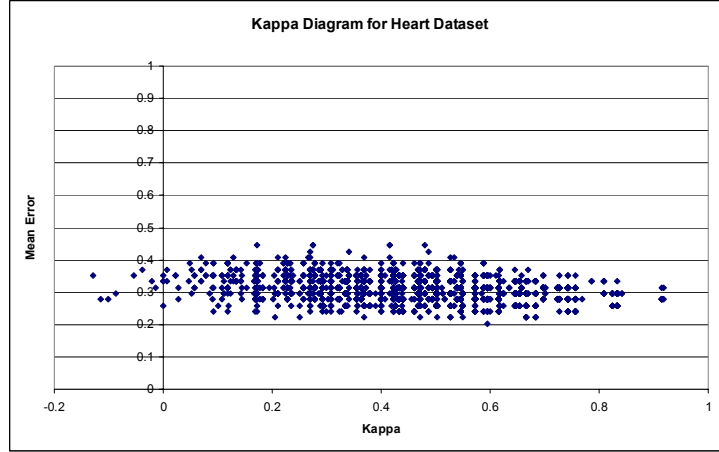


Fig. 1. A Kappa Diagram showing a large spread of Kappa and mean error scores

Kuncheva and Whitaker [7] compare ten statistics that can measure diversity among binary classifier outputs. They looked at four statistics that are averaged pair-wise results, and six that are non-pair-wise results. Since they found the importance of diversity to be unclear, they recommend the pair-wise Q statistic [11] based on the criteria that it is understandable and relatively simple to implement. As every set of paired classifiers produces a Q value, the average, Q_{av} , is used for the diversity value of the ensemble as shown in Figure 2. In this algorithm, classifications are compared as a function of correctness or incorrectness with regard to a validation or test set. This differs from Dietterich’s Kappa algorithm where classifications are compared based solely on the class they represent.

$$Q_{av} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^L \frac{N^{11}N^{00} - N^{01}N^{10}}{N^{11}N^{00} + N^{01}N^{10}}$$

L = Number of classifiers
 N^{00} = Classifier_i is incorrect, Classifier_k is incorrect
 N^{01} = Classifier_i is incorrect, Classifier_k is correct
 N^{10} = Classifier_i is correct, Classifier_k is incorrect
 N^{11} = Classifier_i is correct, Classifier_k is correct

Fig. 2. The averaged Q statistic

$$\kappa = 1 - \frac{\frac{1}{L} \sum_{j=1}^N l(z_j)(L - l(z_j))}{N(L-1)p(1-p)}$$

N = Number of examples
 L = Number of classifiers
 p = Average classifier accuracy
 $l(z_j)$ = Correct classifications for classifier j

Fig. 3. The Inter-rater Agreement function

Dietterich’s Kappa statistic is a variant of the Inter-rater Agreement function (also referred to as Kappa). In this algorithm, the rate of coinciding classifications is generated while taking into account the probability that the agreement is based solely upon chance. Like the Q statistic, it does not take into account the actual classification but rather whether the classification was correct or incorrect. However the Inter-rater Agreement function is not pair-wise and hence is polynomially faster than either aforementioned method. The Inter-rater Agreement function is defined in Figure 3.

The approach closest to our new diversity metric is the “measure of difficulty” [8]. This measure looks at the proportion of classifiers that correctly classify an example. One can consider plotting a histogram of the proportions. The variance of this histogram is considered to be a measure of diversity. Our approach measures the proportion of classifiers getting each example right. However, rather than building a histogram of proportions, we examine the percent correct per example.

We propose the percentage correct diversity measure (PCDM) algorithm shown in Figure 4. It works by finding the test set examples for which between 10% and 90% of the individual classifiers in the ensemble are correct. In this way, examples for which there is general consensus are not considered to be useful in the determination of ensemble diversity. Rather, if an example’s classification is ambiguous, as indicated

by having only the aforementioned percentages of classifiers vote correctly, then the classifiers, for at least that example, are said to be diverse. The bounds of ten and ninety percent were chosen empirically because they cause the algorithm to yield a somewhat uniform distribution of PCDM values over a wide array of ensemble creation techniques. Tighter bounds would place greater strictness on the examples deemed difficult. The use of tighter bounds might be appropriate if comparing two extremely diverse ensembles.

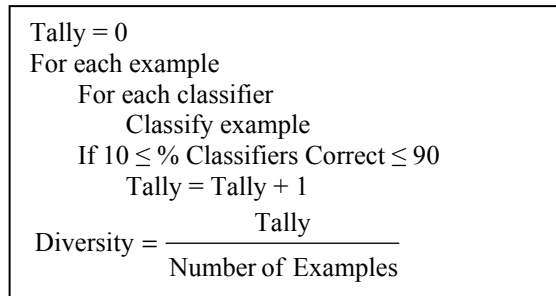


Fig. 4. The Percentage Correct Diversity Measure algorithm

For visualization purposes, let $f(x_i)$ be the percent of classifiers voting correctly on example $x_i \in \{x_1, \dots, x_N\}$ where N is the number of examples. Sorting the list of $N, f(x)$ values and plotting them on a graph generates a monotonically increasing function showing the “spread” of diversity for different examples. A single classifier, or a multiple classifier system where every classifier returns identical classifications, generates the graph shown in 5A which appears similar to a digital signal (0 or 1) with zero classifiers in between the 10% and 90% bounds. Multiple classifiers outputting diverse classifications on the other hand cause different percentages of correct classifications to appear relative to the number of classifiers. Diversity, in a sense, transforms the line from a discrete to a continuous function as in Figure 5B. Greater numbers of examples appearing between .1 and .9 equate to greater PCDM values.

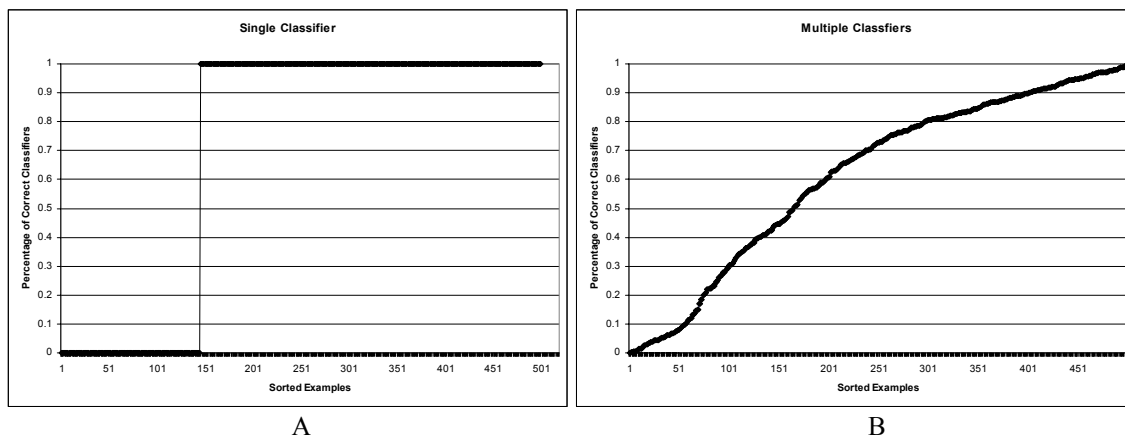


Fig. 5. After sorting the x-axis based on $f(x)$ to create a non-decreasing graph, it is easy to visualize the number of examples that are diverse. (A) shows a single classifier and (B) shows multiple classifiers having diversity

3 Diversity Experiments

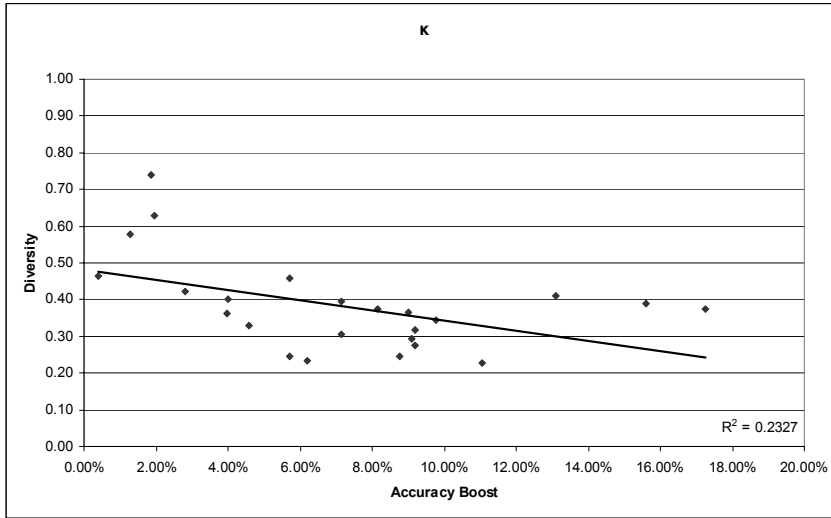
Breiman introduced the concept of creating ensembles of trees that he called random forests [5]. He discussed several methods of creating trees for the forests, one of which was to use bagging and randomly choose an attribute for a test at each node in the decision tree. The best split possible for that attribute would then be chosen. The resultant tree is called a random tree. An ensemble of them, 100 in his experiments, is a random forest. He found that this approach was comparable in accuracy to AdaBoost [6].

We modified C4.5 release 8 to produce random trees by randomly choosing a single attribute to test on at every node. We chose to build 1000 trees in our random forest so that the resultant ensemble is almost certainly larger than necessary and we can better evaluate using diversity to remove trees. Our experiments use a ten-fold cross validation. We build 10,000 trees (1000 per fold) for each of twelve experimental datasets from the UC Irvine Repository [12]. The accuracy of unpruned and pruned ensembles (using the default certainty factor of 25 for pruning) is calculated for each dataset. Table 1 shows the experimental results of the diversity algorithms in measuring the diversity of ensembles as well as the boost in accuracy when comparing average single classifier accuracy and voted accuracy. Decreasing values of Q and Kappa correspond to higher diversity whereas PCDM increases as diversity increases.

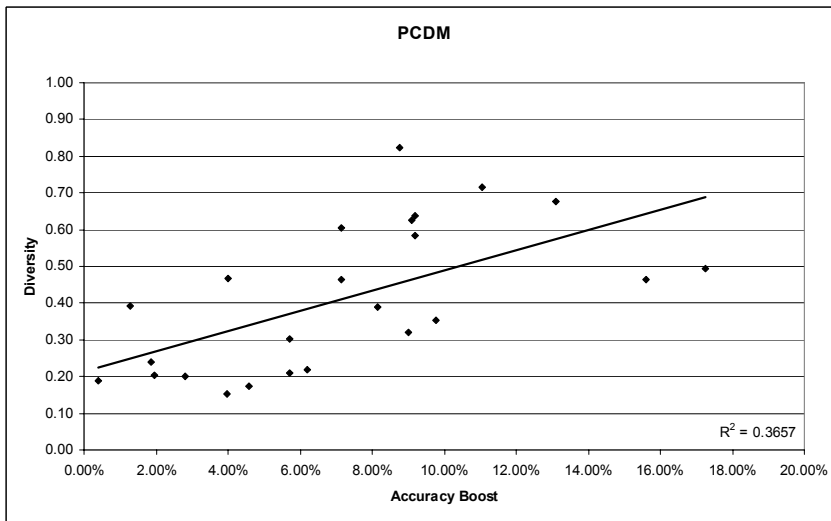
There are several instances where the accuracy of the ensemble of pruned trees is greater than that of the unpruned trees. Dietterich, though he generated the trees in a different fashion, found this as well [4]. However, in no case are the diversity scores for the pruned trees greater than the unpruned trees. Likewise, in no case is the increase in accuracy of the forest of pruned trees greater than that of the unpruned trees. Figure 6, which compares the PCDM against other diversity measures, shows that it picks up the trend of increasing accuracy corresponding to increasing diversity values. In fact, it outperforms the Q metric, which barely exhibits such a trend. The Q metric is also capable of generating divide by zero errors in the event that any one of the classifiers in the ensemble is either 100% or 0% accurate on the test set. In order to compensate for Q generating a divide by zero error, we invalidate the fold since this can occur no matter how diverse the two classifiers are. In terms of running time, both PCDM and Kappa are significantly faster than Q, while PCDM is only marginally faster than Kappa. For example on a 2.53 GHz Intel Pentium 4 it takes 0.008 seconds with PCDM, 0.018 seconds with Kappa, and 111.133 seconds with Q to generate diversity values from the Letter dataset.

Table 1. Diversity vs. accuracy results from datasets appearing in the UCI Data Repository. Q was unable to be calculated for any fold in the Glass dataset due to divide by zero errors in every fold

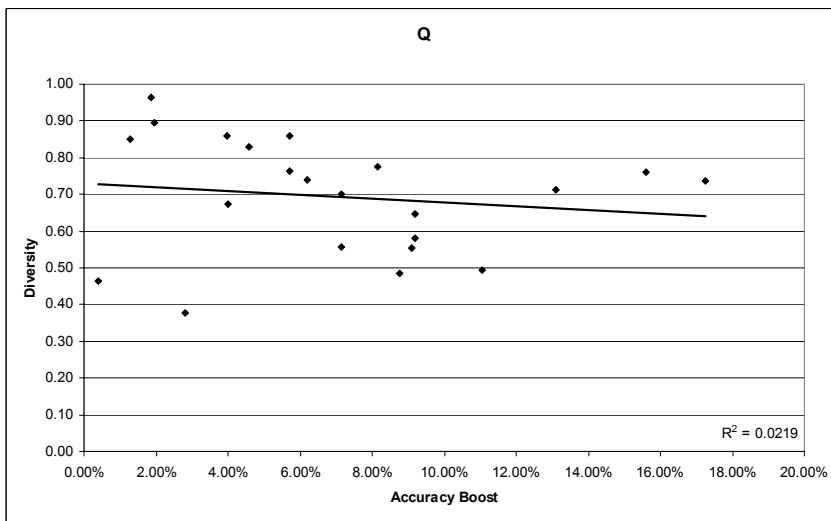
Dataset	Pruning?	Single Accuracy	Forest Accuracy	Accuracy Increase	κ	PCDM	Q
Letter	Unpruned	78.36%	95.61%	17.25%	0.37	0.49	0.74
Letter	Pruned	79.37%	94.97%	15.60%	0.39	0.46	0.76
Led-24	Unpruned	61.85%	74.96%	13.11%	0.41	0.68	0.71
Waveform	Unpruned	73.83%	84.88%	11.05%	0.23	0.72	0.49
Glass	Unpruned	85.95%	95.71%	9.76%	0.34	0.35	--
Waveform	Pruned	75.36%	84.56%	9.20%	0.28	0.64	0.58
Australian	Unpruned	77.47%	86.67%	9.19%	0.32	0.58	0.65
Cleveland	Unpruned	74.90%	84.00%	9.10%	0.29	0.63	0.55
Glass	Pruned	87.20%	96.19%	8.99%	0.37	0.32	--
German	Unpruned	66.82%	75.60%	8.78%	0.25	0.82	0.49
Satimage	Unpruned	83.65%	91.79%	8.14%	0.37	0.39	0.78
Heart	Unpruned	74.70%	81.85%	7.15%	0.31	0.60	0.56
Cleveland	Pruned	77.86%	85.00%	7.14%	0.39	0.46	0.70
Pendigits	Unpruned	92.87%	99.08%	6.21%	0.23	0.22	0.74
Pendigits	Pruned	93.20%	98.92%	5.72%	0.25	0.21	0.76
Satimage	Pruned	85.41%	91.12%	5.71%	0.46	0.30	0.86
Segmentation	Unpruned	93.38%	97.97%	4.58%	0.33	0.17	0.83
Heart	Pruned	77.10%	81.11%	4.01%	0.40	0.47	0.67
Segmentation	Pruned	93.88%	97.84%	3.96%	0.36	0.15	0.86
Iris	Unpruned	92.52%	95.33%	2.82%	0.42	0.20	0.38
Australian	Pruned	83.26%	85.22%	1.96%	0.63	0.20	0.89
Led-24	Pruned	73.09%	74.96%	1.87%	0.74	0.24	0.96
German	Pruned	71.02%	72.30%	1.28%	0.58	0.39	0.85
Iris	Pruned	92.26%	92.67%	0.41%	0.47	0.19	0.46



A



B



C

Fig. 6. Graphs showing the boost in accuracy vs. the diversity score for each of the three methods (A-C)

4 Thinning

By observing that classifiers obtain a diverse set of votes for an example, it is feasible to try to improve the ensemble by removing classifiers that cause misclassifications. In the context of an ensemble of decision trees, this process can be likened to “thinning a forest.” Hence, for the remainder of this paper, the process of removing classifiers from an ensemble will be dubbed “thinning.” In [9] an ensemble is thinned by attempting to include the most diverse and accurate classifiers. They create subsets of similar classifiers (those that make similar errors) and then choose the most accurate classifier from each subset. In [10], the McNemar test was used to determine whether to include a decision tree in an ensemble. This pre-thinning allowed an ensemble to be kept to a smaller size and is different from our “over-produce and choose” approach. We introduce Accuracy in Diversity (AID) thinning, where classifiers that are most often incorrect on examples that are misclassified by many classifiers are removed from the ensemble. That is, if a classifier incorrectly classifies an example which 99% of the others get right, removal would have no effect, whereas if 50% of the other classifiers get the example correct, then it may be a candidate for removal. We call the dataset that is used in analyzing these percentages the thinning set, and is separate from the training set.

A key step in designing the algorithm is to set proper boundaries for the accuracy percentages on thinning examples to use in deciding which classifiers to remove. The greater the diversity on a thinning set, the more variation can be expected on a test set, and setting an upper bound that is too low can result in misclassifying examples previously considered to be “easy.” In setting a lower bound, we would like to exclude the examples that most classifiers get wrong because almost no selection of classifiers will allow us to get these correct. The lower bound for the consideration of examples should be no smaller than the reciprocal of the number of classes which represents, at best, random guessing. One can imagine that mean individual classification accuracy also plays a part in determining the bounds, since it and diversity are so fundamentally related.

The equations in Figure 7 represent the fundamental characteristics chosen to effectively set the correct classifier percentage boundaries for AID thinning. The maximum value of d is 1, however in no case would we want to consider examples as high as 100% correct, so we set the value of α to 0.9. The AID thinning algorithm is shown in Figure 8. Note that after each tree is removed, the accuracy on the thinning set is recalculated.

$$\text{LowerBound} = \mu \cdot d + \frac{1-d}{N}$$
$$\text{UpperBound} = \alpha \cdot d + \mu \cdot (1-d)$$

μ = Mean individual classification accuracy
 α = Approximate maximum upper bound allowed
 d = Percentage correct diversity measure
 N = Number of classes

Fig. 7. Boundary equations for AID thinning

While number removed \leq Maximum number to remove
Recompute boundary points.
Remove the classifier that has the lowest individual accuracy rate for the set of examples between the boundary points.

Fig. 8. The AID thinning algorithm

Since greater diversity typically leads to larger boosts in the accuracy of the forest, we also have created a thinning algorithm that works off of the aforementioned Inter-rater Agreement function called Kappa thinning. In Figure 9, we compare all possible ensembles of $n-1$ classifiers, and eliminate the classifier whose removal causes the diversity to increase the most.

While number removed \leq Maximum number to remove
 For each classifier C_i of ensemble $C_1 \dots C_N$
 Calculate κ of ensemble $C_1 \dots C_{i-1}, C_{i+1} \dots C_N$
 Remove classifier C_i causing the lowest κ value.

Fig. 9. The Kappa thinning algorithm

Finally, we implement a sequential backwards selection (SBS) approach to removing classifiers. We calculate the voted accuracy after generating all possible ensembles of $n-1$ classifiers and remove the classifier which causes the accuracy to increase the most. The SBS algorithm shown in Figure 10 is similar to Kappa thinning except it looks at accuracy rather than diversity.

While number removed \leq Maximum number to remove
 For each classifier C_i of ensemble $C_1 \dots C_N$
 Calculate voted accuracy of ensemble $C_1 \dots C_{i-1}, C_{i+1} \dots C_N$
 Remove classifier C_i causing the highest voted accuracy

Fig. 10. Thinning by sequential backwards selection

To investigate the properties of these thinning algorithms, we performed a ten-fold cross validation, where 10% of the overall data was removed from the training data to create a thinning set. One thousand trees were built on the training data in each fold. Classifiers were chosen for removal based on the thinning set until only 100 classifiers remained. We compared various thinning methods against a randomly constructed ensemble of 100 classifiers, the number Brieman used in his forests [5]. Table 2 shows both AID and Kappa thinning were better than random construction in 21 of the 24 experiments while SBS was better in 20. A thinned ensemble of trees built and pruned on the Australian and Cleveland datasets shows a decrease in accuracy no matter the type of thinning used. This tends to suggest either that the diversity was too great for only 100 classifiers to overcome, or that thinning set selection was poor. Indeed all of these thinning algorithms will learn to overfit the thinning set, negatively affecting the generalization potential of the ensemble.

SBS ties twice with the other methods in accuracy but is better only once. While Kappa thinning produces a larger accuracy boost over AID thinning 14 times, the majority of these cases occur at the bottom half of the table where the maximum increase is smaller. Kappa thinning also has a greater running time than AID thinning. The method to use thusly depends on the needed gain in accuracy and the CPU time available.

Statistical significance tests will not show the small increases in accuracy to be significant; however, we note there is not a significant difference in accuracy even if 90% of the classifiers are removed because of the large variances between folds. With up to 90% of the classifiers removed from an ensemble, ensemble accuracy is generally clearly lower than the best accuracy. However, there is still significant variation between folds and a significance test will not show the change in accuracy to be significant.

Table 2. Thinning methodologies are compared against randomly constructing ensembles. Double asterisks indicate the highest performing AID, Kappa, or SBS thinning algorithm. The table is sorted by the highest maximum gain

Dataset	Pruning?	AID Acc	κ Acc	SBS Acc	Rand Acc	AID Over Rand	κ Over Rand	SBS Over Rand
Iris	Pruned	92.00%	94.67%	92.67%	88.46%	3.54%	6.20%**	4.20%
Iris	Unpruned	94.67%	94.67%	94.67%	89.21%	5.45%**	5.45%**	5.45%**
Heart	Pruned	81.11%	83.70%	81.11%	78.38%	2.73%	5.32%**	2.73%
German	Pruned	72.80%	74.30%	72.70%	69.92%	2.88%	4.38%**	2.78%
German	Unpruned	76.10%	76.00%	74.60%	74.09%	2.01%**	1.91%	0.51%
Heart	Unpruned	82.22%	82.78%	81.48%	81.32%	0.90%	1.46%**	0.16%
Cleveland	Unpruned	83.67%	83.33%	82.67%	82.47%	1.19%**	0.86%	0.19%
Australian	Unpruned	86.52%	85.07%	86.09%	85.62%	0.90%**	-0.55%	0.47%
Segmentation	Pruned	97.71%	98.01%	97.71%	97.37%	0.34%	0.64%**	0.34%
Led-24	Unpruned	75.40%	75.10%	75.10%	74.78%	0.62%**	0.32%	0.32%
Waveform	Unpruned	84.94%	84.72%	84.44%	84.38%	0.56%**	0.34%	0.06%
Led-24	Pruned	75.08%	75.14%	74.86%	74.63%	0.45%	0.51%**	0.23%
Glass	Pruned	95.24%	96.67%	95.71%	96.17%	-0.93%	0.50%**	-0.45%
Letter	Pruned	94.75%	95.04%	94.85%	94.55%	0.20%	0.49%**	0.30%
Satimage	Pruned	91.34%	91.42%	91.35%	90.95%	0.39%	0.47%**	0.41%
Letter	Unpruned	95.44%	95.41%	95.36%	95.22%	0.22%**	0.19%	0.13%
Segmentation	Unpruned	97.88%	97.92%	97.97%	97.77%	0.11%	0.15%	0.20%**
Pendigits	Pruned	98.86%	98.98%	98.90%	98.79%	0.08%	0.19%**	0.11%
Satimage	Unpruned	91.73%	91.79%	91.76%	91.62%	0.11%	0.17%**	0.14%
Pendigits	Unpruned	99.03%	99.00%	98.98%	98.90%	0.13%**	0.10%	0.08%
Waveform	Pruned	84.46%	84.50%	84.30%	84.41%	0.05%	0.09%**	-0.11%
Glass	Unpruned	96.19%	96.19%	96.19%	96.17%	0.02%**	0.02%**	0.02%**
Australian	Pruned	85.07%	84.49%	84.93%	85.11%	-0.04%**	-0.62%**	-0.18%
Cleveland	Pruned	83.00%	83.67%	83.00%	85.88%	-2.88%	-2.21%**	-2.88%

5 Discussion

The concept of diversity is of interest because its effects can easily be seen. However, its quantification and manipulation are not quite well defined. The percentage correct diversity measure allows for some degree of predictability in foreseeing how much of an increase in accuracy can be expected by increasing the diversity of the ensemble. Furthermore, the PCDM is more understandable and efficiently calculated than the Q statistic, and those are the grounds on which Kuncheva and Whitaker originally recommended Q. Finally, the basis for the AID thinning algorithm, removing classifiers that incorrectly classify examples that generate a diverse vote, shows how the diversity concept can be used to shrink ensembles while maintaining or improving accuracy. Kappa thinning shows this as well.

Comparing the original 1000 randomly assembled classifiers to the 100 thinned classifiers, there are generally small losses in accuracy across the board, though these are obviously less than comparing them with the 100 randomly assembled classifiers. A means of dynamically setting the stopping point of thinning and working backwards towards the smallest ensemble with the greatest accuracy has been created, and will be described and compared as part of future work.

Finally, the algorithms presented here could be used to combine multiple different types of classifiers. That is, decision trees, neural networks, and so on could all contribute classification boundary suggestions, the least diverse of which would be thinned away.

Acknowledgments

This work was supported in part by the United States Department of Energy through the Sandia National Laboratories ASCI VIEWS Data Discovery Program, contract number DE-AC04-76DO00789, and the National Science Foundation NSF EIA-013-768.

References

1. Dietterich T., Ensemble methods in machine learning, *Proceedings of the First International Workshop on Multiple Classifier Systems*, pp. 1-15, 2000.
2. Breiman L., Bagging predictors, *Machine Learning*, Volume 24, No. 2, pp. 123-140, 1996.
3. Breiman L., Pasting small votes for classification in large databases and on-line, *Machine Learning*, Volume 36, No. 1-2, pp. 85-103, 1999.
4. Dietterich T., An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization, *Machine Learning*, Volume 40, No. 2, pp. 139-158, 2000.
5. Breiman L., Random Forests, *Machine Learning*, Volume 45, No. 1, pp. 5-32, 2001.
6. Freund Y. and Schapire R.E., Experiments with a new boosting algorithm, *Proc. 13th International Conference on Machine Learning*, pp 148-156, 1996.
7. Kuncheva L.I., Whitaker C.J., Measures of diversity in classifier ensembles, *Machine Learning*, Volume 51, pp. 181-207, 2003.
8. Hansen L. and Salamon P., Neural network ensembles, *IEEE Transactions on PAMI*, 1990, V. 12, No. 10, pp. 993-1001.
9. Giacinto G. and Roli F., An approach to automatic design of multiple classifier systems, *Pattern Recognition Letters*, v. 22, pp. 25-33, 2001.
10. Latinne P., Debeir O., and Decaestecker C., Limiting the number of trees in random forests, *Second International Workshop on Multiple Classifier Systems*, pp. 178-187, 2001.
11. Kuncheva L., Whitaker C., Shipp C., and Duin R., Is independence good for combining classifiers?, *15th International Conference on Pattern Recognition*, pp. 168-171, 2000.
12. Merz C.J. and Murphy P.M., UCI Repository of Machine Learning Databases, Univ. of CA., Dept. of CIS, Irvine, CA., <http://www.ics.uci.edu/~mllearn/MLRepository.html>