Ensemble Diversity Measures and their Application to Thinning

Robert E. Banfield[1], Lawrence O. Hall[1], Kevin W. Bowyer[2], W. Philip Kegelmeyer[3]

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

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

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

Abstract. The diversity of an ensemble can be calculated in a variety of ways. Here a diversity metric and a means for altering the diversity of an ensemble, called "thinning", are introduced. We evaluate thinning algorithms on ensembles created by several techniques on 22 publicly available datasets. When compared to other methods, our percentage correct diversity measure algorithm shows a greater correlation between the increase in voted ensemble accuracy and the diversity value. Also, the analysis of different ensemble creation methods indicates each has varying levels of diversity. Finally, the methods proposed for thinning again show that ensembles can be made smaller without loss in accuracy.

Keywords: thinning, diversity, multiple classifier systems, decision trees, ensembles

1 Introduction

Multiple classifier systems have created a lot of excitement in the Machine Learning community thanks to their potential to greatly increase classification accuracy [1-6]. Many algorithms for generating multiple classifier systems can achieve this accuracy increase with a substantially reduced training time. The cost of testing however is typically not reduced, and in most cases will grow relative to the number of classifiers included in the ensemble. In applications where testing must be done very rapidly, or there is limited space to store the classifiers, this can present a problem. In another light, it stands to reason that given many classifiers, some select subset may make for an even more accurate ensemble.

The boost in accuracy by using multiple classifiers is at least partially a result of diversity [4, 7] – examples that are misclassified by some classifiers are correctly classified by others in such a way that the voted accuracy is greater than that of any single classifier. The work presented here extends our previous work [8] by testing additional thinning methods, using more datasets, and evaluating diversity and thinning algorithms on ensembles created by several different algorithms.

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. In a bias-variance decomposition, we are looking at the variance component when studying diversity [9].

Let S be the number of classes and N be the number of examples. For two classifiers C_n and C_m, let C_{ij} equal the number of examples that are predicted as $class_i$ by C_n and $class_j$ by C_m. C_{ii} is therefore the number of examples for which both classifiers predicted $class_i$. The Kappa statistic from Dietterich [4], which measures the degree of similarity, Θ_1, between two classifiers while subtracting for the probability that the similarity occurs by chance, Θ_2, serves as an illustrative starting point for examining diversity. Given Θ_1 and Θ_2, as defined in (1)-(2), one can calculate κ (3). 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.

$$\Theta_1 = \frac{\sum_{i=1}^{S} C_{ii}}{N} \tag{1}$$

$$\Theta_2 = \sum_{i=1}^{S} \left(\sum_{j=1}^{S} \frac{C_{ij}}{N} \sum_{j=1}^{S} \frac{C_{ji}}{N} \right) \tag{2}$$

$$\kappa = \frac{\Theta_1 - \Theta_2}{1 - \Theta_2} \tag{3}$$

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 each generated Kappa value could be used as a measure of ensemble diversity. A drawback of using such a method would be the computational complexity associated with calculating diversity for each pair-wise combination. The time complexity of this algorithm is $\Theta(L^2(N+S^2))$ where L is the number of classifiers and N is the number of examples.

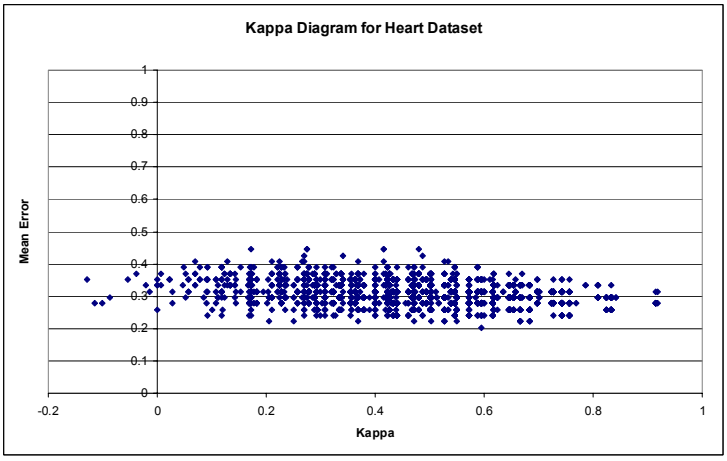


Figure 1. *A Kappa Diagram showing a large spread of Kappa and mean error scores*

Kuncheva and Whitaker [7] compare ten statistics that can be applied to the measurement of diversity. They look at four statistics that are averaged pair-wise results, and six that are non-pair-wise results. Since they found the importance of diversity unclear, they recommend the pair-wise Q statistic [10-12] based on the criteria that it is understandable and relatively simple to implement. 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 strictly on the class. Let $a,b \in \{0,1\}$ represent an incorrect (0) or correct (1) prediction. For two classifiers, C_i and C_k, let N^{ab} equal the number of times C_i chose a and C_k chose b. As every set of paired classifiers produces a Q value, the average, Q_{av}, is used for the diversity value of the ensemble. The Q_{av} equation is shown in (4). Taking into account there exists only two binary classes, the time complexity for this algorithm is $\Theta(L^2N)$.

$$Q_{av} = \frac{2}{L(L-1)} \sum_{i=1}^{L-1} \sum_{k=i+1}^{L} \frac{N_{ik}^{11} N_{ik}^{00} - N_{ik}^{01} N_{ik}^{10}}{N_{ik}^{11} N_{ik}^{00} + N_{ik}^{01} N_{ik}^{10}} \tag{4}$$

Dietterich's Kappa statistic is a variant of the Inter-rater Agreement function (*also* known as Kappa). Let $l(z_j)$ be the number of correct classifications for classifier j and p be the average classification accuracy across all classifiers. In (5), 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, κ does not take into account the actual classification but rather whether the classification was correct or incorrect. The time complexity of the Inter-rater Agreement function is $\Theta(LN)$. Since the algorithm computes diversity for an ensemble and not for each pairwise combination of classifiers, it is faster than either aforementioned method.

$$\kappa = 1 - \frac{\frac{1}{L}\sum_{j=1}^{N} l(z_j)(L - l(z_j))}{N(L-1)p(1-p)} \tag{5}$$

The approach closest to our diversity metric is the "measure of difficulty" [13]. 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 new approach, the percentage correct diversity measure (PCDM), measures the proportion of classifiers getting each example correct. However, rather than build a histogram of proportions, we examine the percent correct per example.

The PCDM algorithm [8] is shown in Figure 2. It works by finding the test set examples for which T_{low} to T_{high} 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. Likewise, difficult examples where few classifiers obtain correct predictions are also disregarded. Hence, if an example's

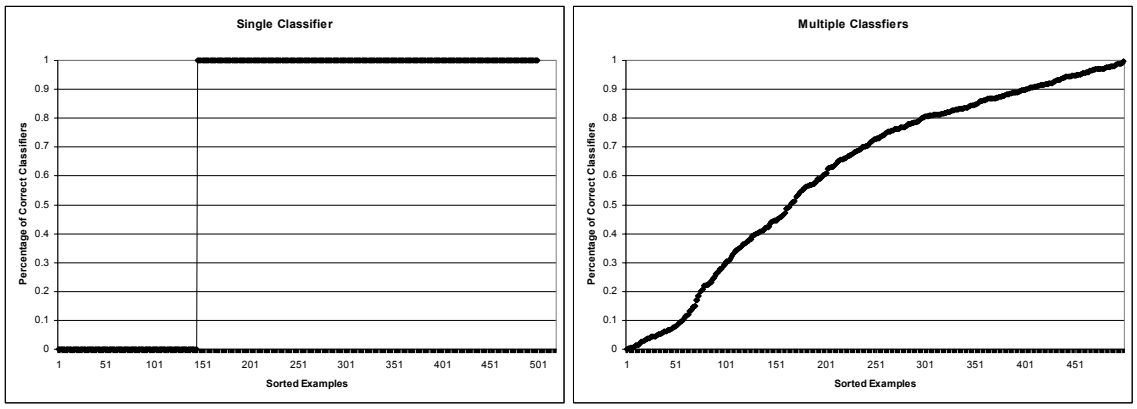
classification is ambiguous, as indicated by having only some percentage of classifiers in the range above vote correctly, then the classifiers, for at least that example, are said to be diverse. We use values of 10 and 90 for T_{low} and T_{high}. These were chosen empirically because they cause the algorithm to yield a somewhat uniform distribution of PCDM values over an array of ensemble creation techniques. Tighter bounds would place greater strictness on the examples deemed easy or difficult. The use of tighter bounds might be appropriate if comparing two extremely diverse ensembles.

```
Tally = 0
For each example
    For each classifier
        Classify example
    Endfor
    If T_low ≤ % Classifiers Correct ≤ T_high
        Tally = Tally + 1
Endfor
Diversity = Tally / Number of Examples
```

Figure 2. *The Percentage Correct Diversity Measure algorithm*

For visualization purposes, let $f(x_i)$ be the percent of classifiers voting correctly on example $x_i \in \{x_1, \ldots, x_N\}$, 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 3a 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 3b. Greater numbers of examples appearing between 0.1 and 0.9 equate to greater PCDM values.



A B

Figure 3. *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

We investigate the diversity metrics of the previous section on a variety of ensemble creation techniques. The goal is to determine whether any metric can predict accuracy increases as a function of diversity increases. In performing our experiments, we have modified C4.5 release 8 [14] to evaluate several ensemble creation techniques: bagging, random forests, random subspaces, and random trees.

Breiman's "bootstrap aggregating", known as "bagging" [2], creates classifiers by manipulating the original training set by successively resampling it with replacement to create many different training sets. For every training set created (called a "bag"), a classifier is trained. Prediction of an example by an ensemble uses an unweighted majority vote of all created classifiers.

Breiman introduced random forests, the concept of creating an ensemble of decision trees from bags of data in a non-deterministic way [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 test possible for that attribute would then be chosen. An ensemble of these trees is called a random forest. He found that this approach was comparable in accuracy to AdaBoost [6].

Ho's random subspaces algorithm [15, 16] creates multiple training sets by picking fifty percent of the attributes at random and using only those attributes in the new training set. Each training set consists of different attributes for the same examples. A classifier is then created for each training set.

Dietterich's random trees method [1] works by analyzing the best twenty tests across all attributes, and choosing one at random as the test on which to split. One continuous attribute can produce many tests, each of which is a candidate for the list of the twenty best splits. A discrete attribute on the other hand can produce only one test.

In performing these experiments we chose to build 1000 trees so that the resultant ensemble is almost certainly larger than necessary and we can better evaluate using diversity to remove trees. Breiman shows, using the Strong Law of Large Numbers, that ensemble techniques do not suffer from overfitting as more classifiers are added [5]. Our experiments use a ten-fold cross validation. We build 10,000 trees (1000 per fold) for each of 22 experimental datasets: 19 from the UC Irvine Repository [17], "Phoneme" and "Satimage" from the ELENA project [18], and "Credit-g" from the NIADD [19]. The accuracy of unpruned and pruned ensembles (using the C4.5 default certainty factor of 25 for error-based pruning) is calculated for each dataset. Appendix A Tables A.1-A.4

show 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. Figures 4-7 plot the diversity value against the accuracy increase from voting each classifier in the ensemble and provide a linear regression best-fit line. Decreasing values of Q and Kappa correspond to higher diversity whereas PCDM increases as diversity increases.

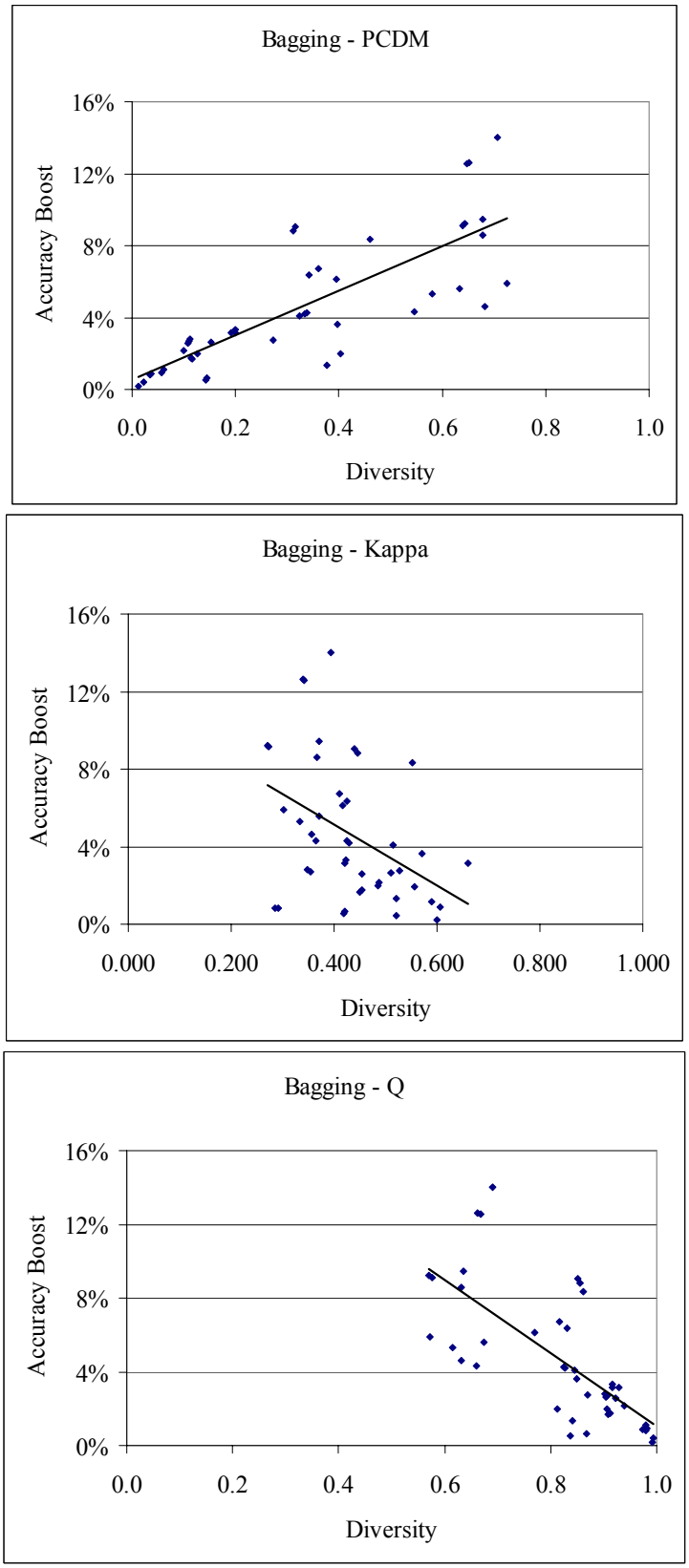


Figure 4. *Accuracy boost versus diversity values for bagging*

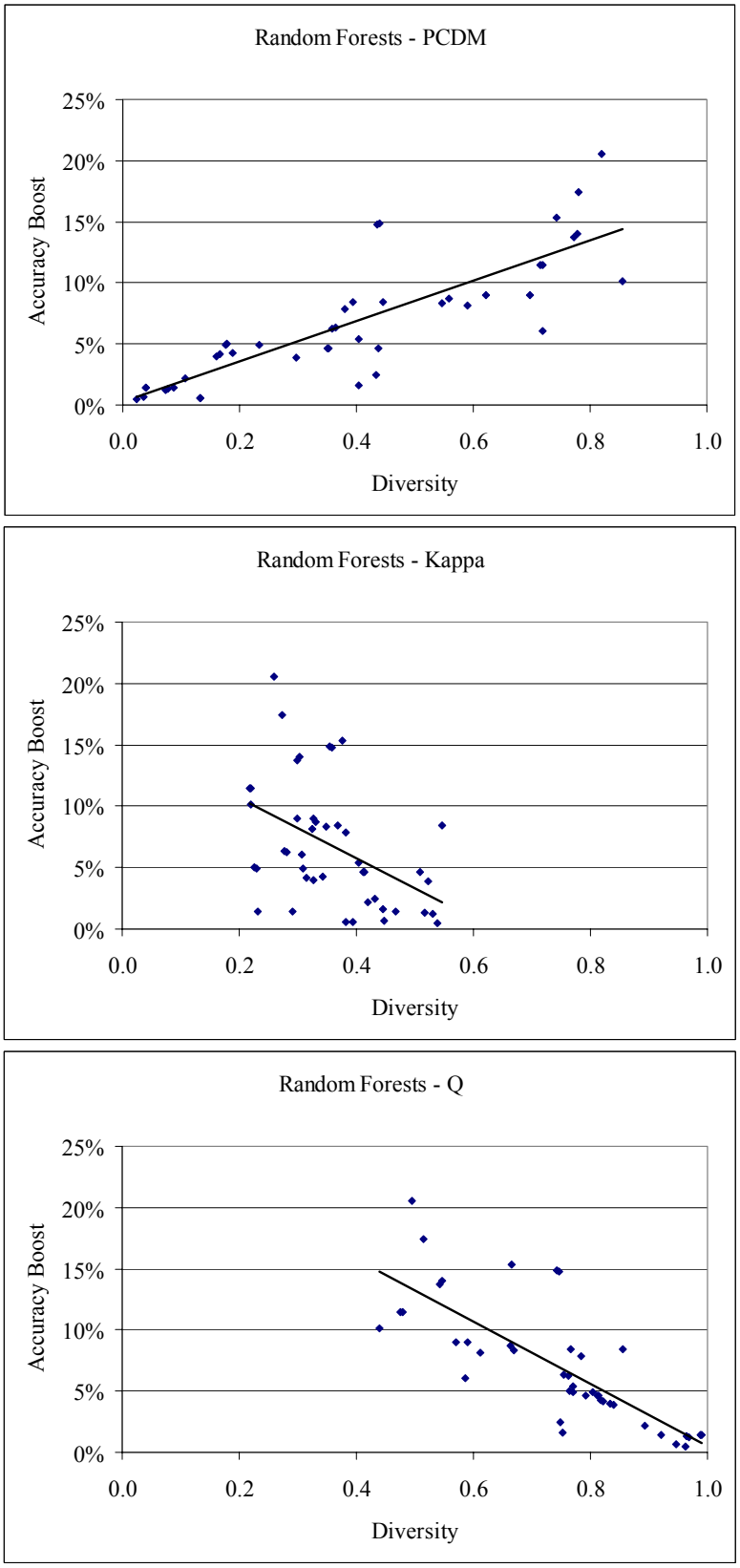


Figure 5. *Accuracy boost versus diversity values for random forests*

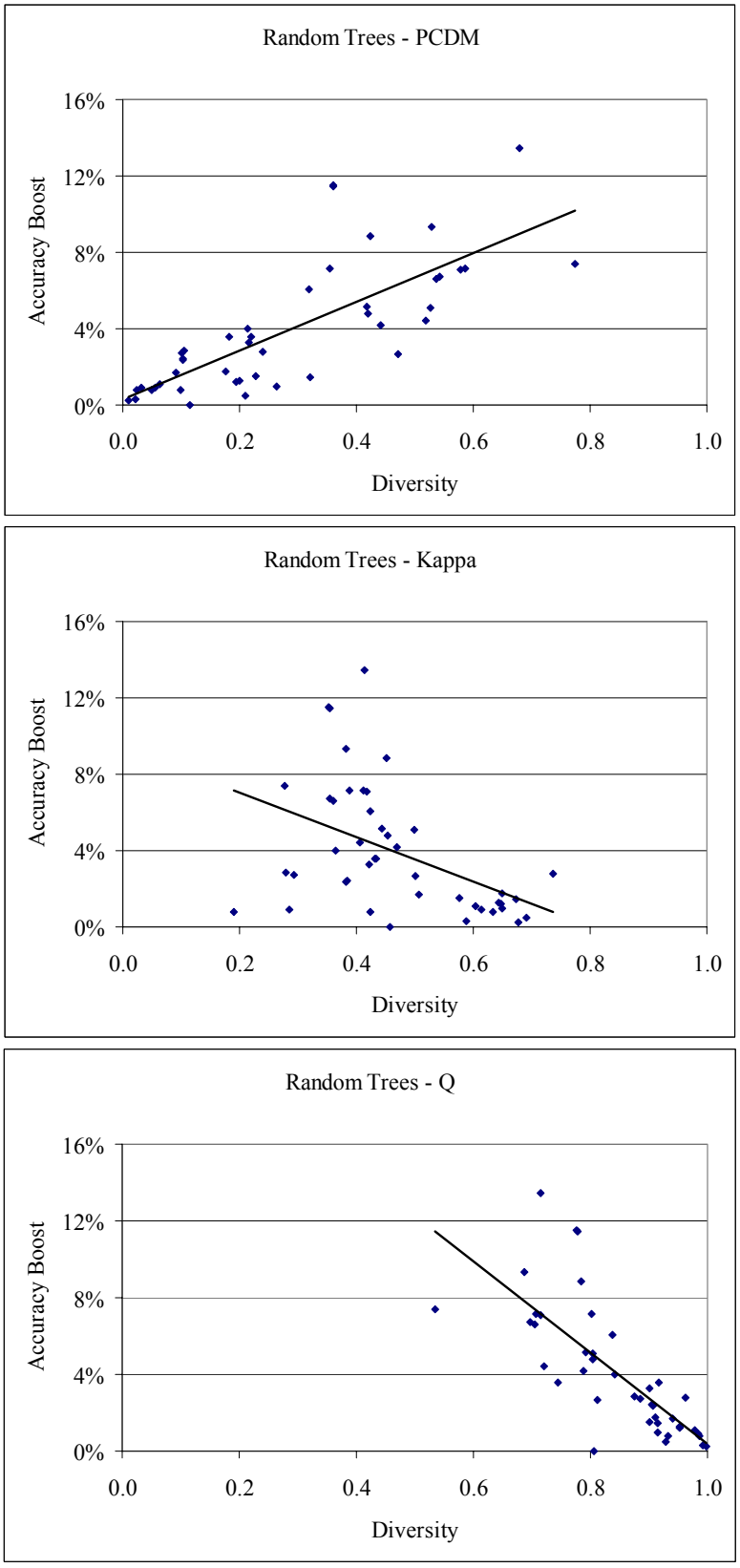


Figure 6. *Accuracy boost versus diversity values for random trees*

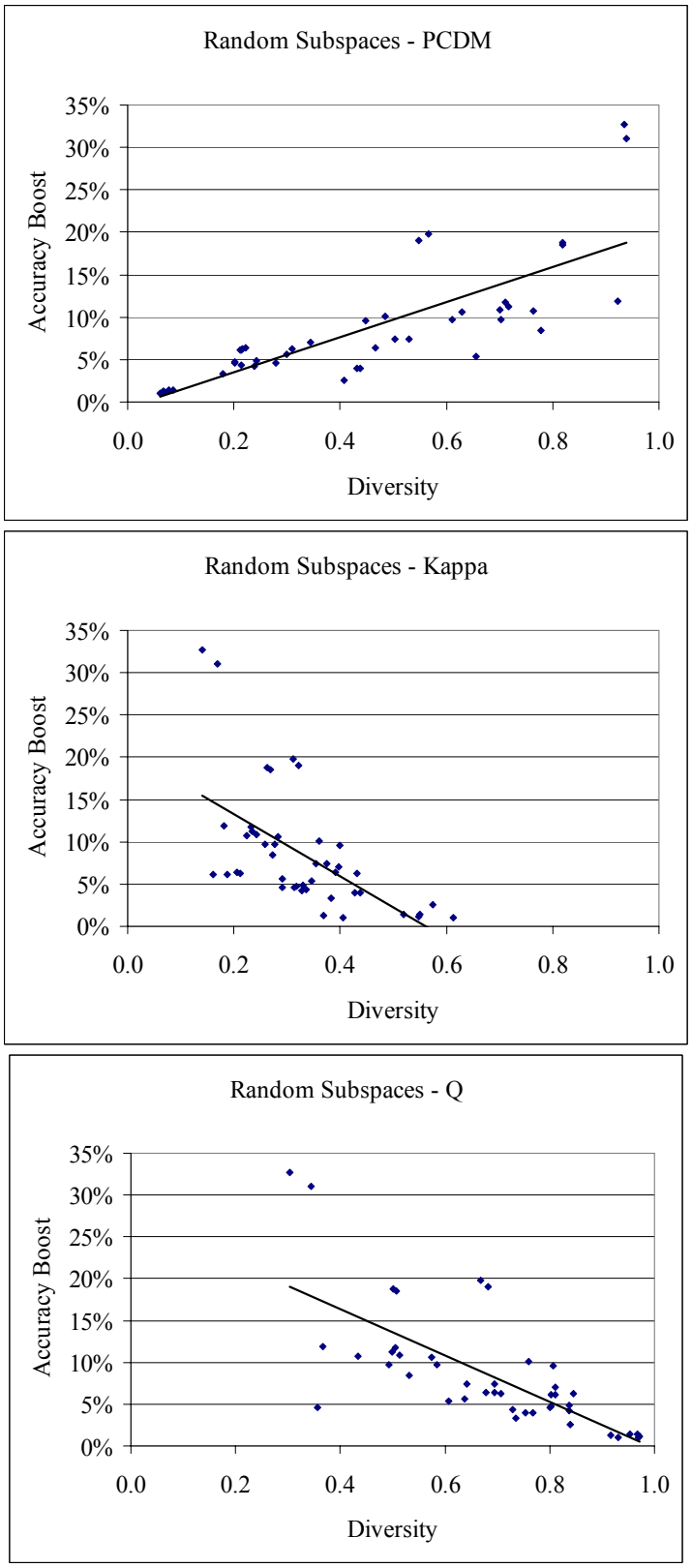


Figure 7. *Accuracy boost versus diversity values for random subspaces*

Table 1
R^2 values for each of the ensemble creation methods and diversity measures

Ensemble Creation Technques	PCDM	Q	K
Bagging	.6353	.5046	.1775
Random Forests	.6905	.6105	.2602
Random Trees	.5765	.5614	.2106
Random Subspaces	.6096	.5071	.3512

Table 1 shows the R^2 value, a measure for determining how well the linear regression fits the measured data, for each diversity algorithm. For all four ensemble creation methods, the PCDM metric has a higher R^2 value than does κ or Q. For these datasets, PCDM is slightly better correlated with the accuracy increase in an ensemble. Q has the second highest R^2 in all four cases. The Q metric however is 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. The Kappa algorithm does not show as strong a relationship between accuracy increase and diversity. 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 P4 using the Letter dataset takes .008 seconds to calculate the PCDM, .018 seconds to calculate Kappa, and 111.133 seconds to calculate the Q value.

Generally speaking, an ensemble of unpruned trees obtains a larger boost in accuracy by voting than a pruned ensemble. In 15 out of 88 experiments, the accuracy boost for the ensemble of pruned trees is greater. In 12 of these 15 cases the PCDM values for the pruned trees are greater than the unpruned trees, as would be expected despite the

inclination to say pruned trees are less diverse. The Q value shows this 8 times and Kappa 6 times.

Finally, Table 2 shows that of all the diversity methods used, the random subspaces method generates the most diverse trees. This is a surprising result because a generic decision tree is built, though the number of attributes is reduced randomly in each training set. Intuitively, random forests using only one randomly picked attribute would seem to be the most diverse since one can imagine the individual trees would be very different from each other. It also suggests that more than 100 trees, the amount recommended by Ho [16], should be used, since more classifiers are often needed to cope with an ensemble of highly diverse, and often inaccurate, classifiers.

Table 2

The average diversity of each of the ensemble creation techniques is compared. Low values of Q and κ correspond to high diversity. High values of PCDM correspond to high diversity

Ensemble Creation Techniques	Average Accuracy	Voted Accuracy	Accuracy Boost	Average κ	Average PCDM	Average Q
Random Trees	84.43%	88.36%	3.93%	0.468	0.283	0.845
Bagging	83.56%	88.16%	4.60%	0.434	0.329	0.821
Random Forests	82.23%	88.97%	6.73%	0.359	0.392	0.744
Random Subspaces	80.59%	89.07%	8.48%	0.330	0.439	0.686

4 Thinning

4.1 A Review of Other Methods

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. We reference the terminology described in [8] to describe the removal of decision trees from a forest. Since this process can be likened to “thinning a forest”, we call it “thinning.”

In [20], thinning was performed by testing random subsets of neural networks from an ensemble and choosing the best subset based on its performance on a validation set. While the results of this method showed some increase in accuracy, it is not scalable for ensembles with a large number of classifiers.

In [21] an ensemble is thinned by attempting to include the most diverse and accurate classifiers. They create subsets of similar neural networks (those that make similar errors) and then choose the most accurate from each subset. Since far fewer classifiers are used in neural network ensembles compared to typical decision tree ensembles, the time complexity was not prohibitive.

In [22], 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. In the rest of this section, we will propose and describe four additional thinning methods

4.2 Some New Thinning Approaches

In Accuracy in Diversity (AID) thinning, the 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 it is separate from the training and testing sets. The thinning set is used as a validation 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 8 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 9. 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

Figure 8. *Boundary equations for AID thinning*

```
While number removed ≤ Maximum number to remove
    Recompute the LowerBound and UpperBound boundary points.
    Remove the classifier that has the lowest individual accuracy rate for the set of
        examples between the boundary points.
Endwhile
```

Figure 9. *The AID thinning algorithm*

We have created another thinning algorithm based on the correctness of both the ensemble and the classifier with regard to a thinning set. A classifier is rewarded for obtaining a correct decision, and rewarded more for obtaining a correct decision when the ensemble is incorrect. A classifier is penalized in the event both the ensemble and classifier are incorrect. We call the algorithm Concurrency thinning and it is shown in Figure 10.

```
For each classifier Ci
   For each example
      If Ensemble Incorrect and Classifier Incorrect
         Metrici = Metrici -2
      If Ensemble Incorrect and Classifier Correct
         Metrici = Metrici + 2
      If Ensemble Correct and Classifier Correct
         Metrici = Metrici + 1
   Endfor
Endfor
Remove Ci with the lowest Metrici
```

Figure 10. *The Concurrency 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 11, 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 ≤ Maximum number to remove
    For each classifier C_i of ensemble C_1…C_N
        Calculate κ of ensemble C_1…C_i-1 , C_i+1…C_N
    Endfor
    Remove classifier C_i causing the lowest κ value.
Endwhile
```

Figure 11. *The Kappa thinning algorithm*

Finally, we implemented 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 12 is similar to Kappa thinning except it looks at accuracy rather than diversity.

```
While number removed ≤ Maximum number to remove
    For each classifier C_i of ensemble C_1…C_N
        Calculate voted accuracy of ensemble C_1…C_i-1, C_i+1…C_N
    Endfor
    Remove classifier C_i causing the highest voted accuracy
Endwhile
```

Figure 12. *Thinning by sequential backwards selection*

4.3 Experimental Methodology

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], for both bagging and random forests. Tables 3 and 4 show the results for random forests and bagging respectively. They are sorted by the maximum gain in accuracy of the three methods.

Bold face type indicates the algorithm with the highest accuracy. We observe an increase in accuracy by using one of the selected thinning methods over the randomly constructed ensemble. A summary of our findings, including a Borda count [23], is available in Table 5. The Borda count is calculated by assigning "place" values to each of the ensemble creation methods (first place, second place, etc.). The first place winner obtains *A* points, second place obtains *A-1* points, third place obtains *A-2* points, and so on, where *A* is the number of classification algorithms compared. The sum of those values across all datasets is the Borda count value. The highest value is an indicator of the best overall algorithm.

Table 3

Thinning compared against randomly constructed random forests

Dataset	Pruning?	Random Accuracy	AID Over Random	K Over Random	SBS Over Random	Concur. Over Random
Iris	Pruned	88.46%	3.54%	**6.20%**	4.20%	**6.20%**
Iris	Unpruned	89.21%	**5.45%**	**5.45%**	**5.45%**	**5.45%**
Credit-g	Pruned	69.92%	2.88%	4.38%	2.78%	**4.58%**
Glass	Unpruned	77.62%	1.43%	1.90%	0.95%	**2.86%**
Glass	Pruned	78.10%	0.95%	1.90%	-1.90%	**2.86%**
Credit-g	Unpruned	74.09%	**2.01%**	1.91%	0.51%	1.61%
Autos	Pruned	87.00%	1.50%	**2.00%**	0.50%	1.50%
Autos	Unpruned	87.00%	0.50%	**1.50%**	1.00%	**1.50%**
Heart-h	Pruned	77.93%	0.69%	**1.38%**	0.69%	**1.38%**
Heart-h	Unpruned	77.93%	0.34%	1.03%	0.34%	**1.38%**
Heart-c	Unpruned	82.47%	**1.19%**	0.86%	0.19%	0.19%
Ion	Pruned	92.86%	0.57%	**0.86%**	0.57%	**0.86%**
Ion	Unpruned	93.14%	0.29%	**0.86%**	0.29%	0.57%
Horse Colic	Unpruned	84.32%	0.54%	**0.81%**	**0.81%**	0.00%
Phoneme	Unpruned	90.30%	0.04%	**0.76%**	0.20%	0.35%
Phoneme	Pruned	90.22%	-0.11%	**0.67%**	0.07%	0.37%
Segmentation	Pruned	97.37%	0.34%	**0.64%**	0.34%	0.38%
Led-24	Unpruned	74.78%	**0.62%**	0.32%	0.32%	0.02%
Credit-a	Unpruned	85.80%	-0.58%	0.29%	-0.43%	**0.58%**
Waveform	Unpruned	84.38%	**0.56%**	0.34%	0.06%	0.38%
Horse Colic	Pruned	82.43%	**0.54%**	-0.27%	**0.54%**	-0.54%
Led-24	Pruned	74.63%	0.45%	**0.51%**	0.23%	0.39%
Letter	Pruned	94.55%	0.20%	**0.49%**	0.30%	0.40%
Satimage	Pruned	90.95%	0.39%	**0.47%**	0.41%	0.33%
Oil	Unpruned	96.17%	0.21%	0.32%	0.21%	**0.43%**
Oil	Pruned	96.17%	0.11%	0.21%	0.11%	**0.43%**
Breast-y	Unpruned	72.07%	**0.34%**	-1.03%	0.00%	**0.34%**
Waveform	Pruned	84.41%	0.05%	**0.09%**	-0.11%	0.29%
Letter	Unpruned	95.22%	**0.22%**	0.19%	0.13%	**0.22%**
Segmentation	Unpruned	97.77%	0.11%	0.15%	**0.20%**	0.15%
Pendigits	Pruned	98.79%	0.08%	**0.19%**	0.11%	0.02%
Satimage	Unpruned	91.62%	0.11%	**0.17%**	0.14%	0.14%
Pendigits	Unpruned	98.90%	**0.13%**	0.10%	0.08%	**0.13%**
Page	Pruned	97.93%	0.00%	0.04%	**0.09%**	0.05%
Page	Unpruned	97.97%	**0.07%**	0.02%	0.04%	0.02%
Hypo	Unpruned	99.56%	0.03%	**0.06%**	0.03%	**0.06%**
Hypo	Pruned	99.62%	0.00%	0.03%	0.00%	**0.06%**
Anneal	Pruned	99.78%	**0.00%**	-0.11%	-0.22%	-0.22%
Anneal	Unpruned	100.00%	**0.00%**	-0.11%	**0.00%**	-0.11%
Breast-w	Pruned	97.86%	0.00%	**-0.29%**	0.00%	-0.14%
Breast-w	Unpruned	98.00%	-0.43%	**-0.29%**	-0.43%	-0.43%
Credit-a	Pruned	86.38%	**-0.43%**	-1.74%	**-0.43%**	-1.59%
Breast-y	Pruned	74.14%	-1.03%	-2.41%	-1.03%	**-0.69%**
Heart-c	Pruned	85.88%	-2.88%	-2.21%	-2.88%	**-1.88%**

Table 4
Thinning compared against randomly constructed bagged ensembles

Dataset	Pruning?	Random Accuracy	AID Over Random	K Over Random	SBS Over Random	Concur. Over Random
Autos	Unpruned	76.50%	1.50%	5.00%	0.00%	**5.50%**
Autos	Pruned	78.00%	-0.50%	**3.50%**	0.00%	2.00%
Glass	Unpruned	70.48%	1.90%	1.90%	0.95%	**3.33%**
Glass	Pruned	70.95%	1.43%	2.86%	-0.48%	**3.33%**
Heart-c	Unpruned	80.00%	1.43%	0.00%	1.43%	**3.33%**
Heart-c	Pruned	79.52%	1.43%	0.48%	**2.86%**	2.81%
Breast-y	Unpruned	68.62%	0.00%	1.72%	0.69%	**2.07%**
Heart-h	Unpruned	75.52%	**1.38%**	0.00%	**1.38%**	**1.38%**
Horse Colic	Pruned	84.86%	0.81%	**1.35%**	0.54%	0.81%
Credit-g	Pruned	76.50%	-0.20%	-0.40%	-0.60%	**1.20%**
Breast-y	Pruned	72.07%	0.69%	**1.03%**	-0.34%	0.00%
Heart-h	Pruned	75.52%	-1.38%	**1.03%**	-1.38%	0.69%
Credit-g	Unpruned	76.10%	0.20%	0.20%	-0.60%	**0.90%**
Letter	Pruned	93.04%	0.11%	**0.78%**	0.11%	0.52%
Iris	Unpruned	94.00%	**0.67%**	0.00%	0.00%	**0.67%**
Iris	Pruned	94.00%	**0.67%**	0.00%	**0.67%**	**0.67%**
Letter	Unpruned	93.34%	0.05%	**0.65%**	0.03%	0.33%
Credit-a	Pruned	87.25%	0.43%	0.43%	**0.58%**	-0.14%
Segmentation	Pruned	97.14%	0.39%	**0.52%**	0.30%	0.48%
Segmentation	Unpruned	97.32%	0.35%	0.30%	0.09%	**0.52%**
Phoneme	Pruned	89.31%	0.07%	**0.46%**	0.30%	**0.46%**
Phoneme	Unpruned	89.41%	0.20%	0.43%	0.39%	**0.44%**
Credit-a	Unpruned	86.96%	-0.14%	0.14%	0.29%	**0.43%**
Oil	Pruned	95.85%	0.32%	0.21%	**0.43%**	0.00%
Oil	Unpruned	95.96%	**0.32%**	0.21%	0.11%	0.00%
Ion	Unpruned	94.57%	-0.29%	-0.29%	-0.29%	**0.29%**
Ion	Pruned	94.57%	0.00%	0.00%	0.00%	**0.29%**
Horse Colic	Unpruned	85.95%	-1.08%	0.00%	**0.27%**	-0.54%
Pendigits	Pruned	98.22%	0.06%	**0.24%**	0.03%	0.17%
Hypo	Unpruned	98.83%	0.06%	0.06%	0.09%	**0.19%**
Satimage	Pruned	91.22%	**0.18%**	0.02%	0.07%	-0.05%
Pendigits	Unpruned	98.36%	-0.03%	**0.17%**	-0.03%	0.03%
Satimage	Unpruned	91.22%	**0.16%**	0.02%	-0.05%	-0.32%
Breast-w	Pruned	96.14%	**0.14%**	-0.29%	**0.14%**	-0.57%
Waveform	Pruned	85.92%	0.10%	**0.14%**	**0.14%**	0.02%
Page	Unpruned	97.55%	0.07%	**0.13%**	-0.02%	0.09%
Hypo	Pruned	98.89%	0.06%	0.03%	0.06%	**0.13%**
Waveform	Unpruned	86.02%	-0.42%	0.10%	-0.12%	**0.12%**
Led-24	Unpruned	73.64%	**0.12%**	-0.28%	-0.34%	**0.12%**
Led-24	Pruned	74.70%	0.00%	**0.12%**	-0.08%	0.04%
Anneal	Pruned	98.78%	-0.11%	**0.11%**	-0.22%	**0.11%**
Page	Pruned	97.53%	0.02%	**0.11%**	-0.02%	**0.11%**
Anneal	Unpruned	99.22%	**0.00%**	**0.00%**	**0.00%**	**0.00%**
Breast-w	Unpruned	96.00%	-0.14%	-0.29%	**0.00%**	-0.14%

Table 5

The total accuracy increase for each ensemble creation method compared to random construction of a 100 member ensemble. A Borda count is also provided.

Ensemble Creation Method	Total Accuracy Increase (%)				Borda Count				
	AID	K	SBS	Concur.	AID	K	SBS	Concur.	Random
Random Forests	21.03	28.66	14.46	30.90	152	174	139	175	86
Bagging	11.02	22.94	7.38	31.82	146	166	129	177	100

4.4 Analysis of Results

For the random forest method, AID thinning was better than random construction 33 out of 44 times, Kappa thinning was better 35 times, SBS was better 32 times, and Concurrency thinning was better 35 times. In many cases, thinning causes the difference in accuracy between the unpruned and pruned ensemble to decrease. An ensemble of pruned trees on the "Credit-g" dataset, with an accuracy of only 69.92%, obtains an accuracy of 75.50% via Concurrency thinning; the unpruned trees increase from 74.09% to 75.70% after thinning. Ensembles built on "Breast-w" show no accuracy increase for any thinning method. This suggests either that the diversity was too great for only 100 classifiers to overcome, or that the 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. Figure 13 shows a small example of using a thinning set. Thinning set accuracy continues to increase while the accuracy on the test set begins to decrease. On larger datasets this can be more profound. Accuracy on both the thinning set and the test set will be less once there are no longer enough trees to support an ensemble. A potential solution to the overfitting problem is to use a validation set to determine the stopping point for the thinning algorithms.

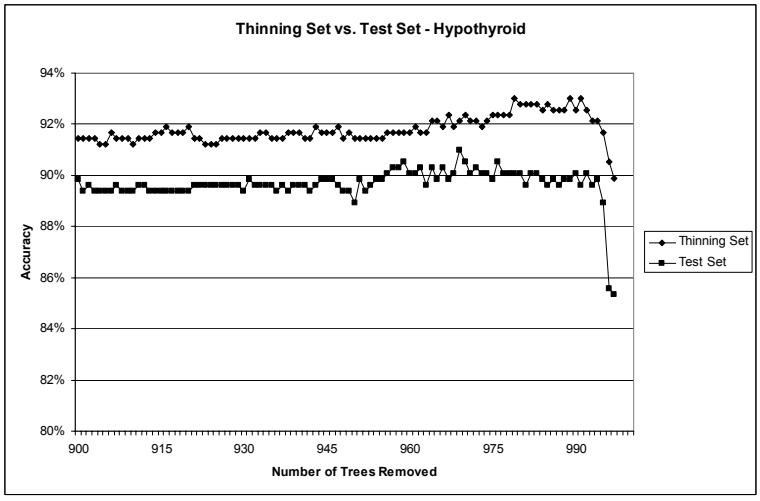


Figure 13. *Thinning set versus test set accuracy*

SBS ties six times with the other methods in accuracy but is better only twice. SBS is often outperformed despite the fact that it concentrates specifically on voted accuracy. Based on the knowledge of the importance of diversity, it is clear why this happens. SBS does not consider diversity to be a factor, causing the generalization potential of the ensemble on new data to be poor. This is born out in the Borda count where SBS is consistently the least performing thinning algorithm.

For the bagging ensemble creation method, AID thinning was better than a randomly constructed ensemble 30 of 44 times, Kappa thinning was better 32 times, SBS was better 24 times, and Concurrency thinning was better 34 times. SBS continues to remain the least effective method, not able to keep up in wins/losses, average accuracy increase, or Borda count. Concurrency thinning and Kappa thinning both perform well, having the highest accuracy increases and Borda counts.

Overall, Kappa and Concurrency thinning are more accurate than AID thinning. However both also have a greater running time than AID thinning. Kappa thinning runs for approximately 33% more time than AID thinning. Concurrency thinning runs for

approximately 50% more time than Kappa thinning. These distinctions in running time might play an important role as datasets become larger. The method to use thus depends on the needed gain in accuracy and the CPU time available. It should be noted that SBS has the greatest running time of all these methods since a majority vote, which takes longer to evaluate than average accuracy or the diversity calculations, must be computed multiple times before any classifier is removed. A summary of all methods is provided in Table 5.

Statistical significance tests did not show the small increases in accuracy to be significant; however, there is not a significant difference in accuracy even if 90% of the classifiers are removed because of the large variances between folds. This is validated by comparing the original 1000 random classifiers with the 100 random classifiers. With up to 90% of the classifiers removed from an ensemble, ensemble accuracy is 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. Despite the lack of statistical significance in the accuracy increases, the Borda count shows that the thinning algorithms are consistently more accurate than random assembly of the ensemble.

5 Summary and 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 simple and more efficiently calculated than the Q statistic, and those are the grounds on which Kuncheva and Whitaker originally recommended Q over the other nine measures they considered.

Removing classifiers that incorrectly classify examples for which there is a diverse vote shows how the diversity concept can be used to shrink ensembles while maintaining or improving accuracy. Each thinning algorithm scores a higher Borda count and has a higher accuracy increase over random assembly. Concurrency thinning shows this particularly well, besting the other algorithms in every category.

Comparing the original 1000 randomly generated classifiers to the 100 thinned classifiers, the latter is slightly less accurate for most datasets, but obviously more accurate than the ensemble consisting of 100 randomly assembled classifiers. In general, the thinning methods produce smaller, accurate ensembles.

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

References

1. T. Dietterich, "Ensemble methods in machine learning," presented at 1st International Workshop on Multiple Classifier Systems, Cagliari, Italy, pp. 1-15, 2000.
2. L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123-140, 1996.
3. L. Breiman, "Pasting small votes for classification in large databases and on-line," *Machine Learning*, vol. 36, pp. 85-103, 1999.
4. T. Dietterich, "An experimental comparison of three methods for constructing ensembles of decision trees: Bagging, boosting, and randomization," *Machine Learning*, vol. 40, pp. 139-158, 2000.
5. L. Breiman, "Random forests," *Machine Learning*, vol. 45, pp. 5-32, 2001.

6. Y. Freund and R. E. Schapire, "Experiments with a new boosting algorithm," presented at 13th International Conference on Machine Learning, pp. 148-156, 1996.
7. L. I. Kuncheva and C. J. Whitaker, "Measures of diversity in classifier ensembles," *Machine Learning*, vol. 51, pp. 181-207, 2003.
8. R. E. Banfield, L. O. Hall, K. W. Bowyer, and W. P. Kegelmeyer, "A new ensemble diversity measure applied to thinning ensembles," presented at 4th International Workshop on Multiple Classifier Systems, pp. 306-316, 2003.
9. P. Domingos, "A unified bias-variance decomposition and its applications," presented at 17th International Conference on Machine Learning, pp. 231-238, 2000.
10. G. U. Yule, "On the association of attributes in statistics," presented at Philosophical Transactions of the Royal Society of London, pp. 257-319, 1900.
11. L. I. Kuncheva, C. J. Whitaker, C. Shipp, and R. Duin, "Is independence good for combining classifiers?," presented at 15th International Conference on Pattern Recognition, pp. 168-171, 2000.
12. L. I. Kuncheva, Skurichina M., Duin W.I., "An experimental study on diversity for bagging and boosting with linear classifiers.," *Information Fusion*, vol. 3, pp. 245-258, 2002.
13. L. Hansen and P. Salamon, "Neural network ensembles," *IEEE Transactions on PAMI*, vol. 12, pp. 993-1001, 1990.
14. J. R. Quinlan, *C4.5: Programs for machine learning*. San Francisco, CA, USA: Morgan Kaufmann, 1993.
15. T. K. Ho, "Random decision forests," presented at 3rd International Conference on Document Analysis and Recognition, pp. 278-282, 1995.
16. T. K. Ho, "The random subspace method for constructing decision forests," *IEEE Transactions on PAMI*, vol. 20, pp. 832-844, 1998.
17. C. J. Merz and P. M. Murphey, "UCI repository of machine learning databases," http://www.ics.uci.edu/~mlearn/MLRepository.html, 2003.
18. "ELENA Project," http://www.dice.ucl.ac.be/neural-nets/Research/Projects/ELENA/elena.htm, 2003.
19. "NIAAD," http://www.liacc.up.pt/ML, 2003.
20. A. J. C. Sharkey, N. E. Sharkey, U. Gerecke, and G. O. Chandroth, "The 'test and select' approach to ensemble combination," presented at 1st International Workshop on Multiple Classifier Systems, pp. 30-44, 2000.
21. G. Giacinto and F. Roli, "An approach to automatic design of multiple classifier systems," *Pattern Recognition Letters*, vol. 22, pp. 25-33, 2001.
22. P. Latinne, O. Debeir, and C. Decaestecker, "Limiting the number of trees in random forests," presented at 2nd International Workshop on Multiple Classifier Systems, pp. 178-187, 2001.
23. J. C. de Borda, *Memoire sur les elections au scrutin*. Paris: Historie de l'Academie Royale des Sciences, 1781.

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. A shorter version of this work was published June 2003 in the proceedings of the 4th International Workshop on Multiple Classifier Systems, Guildford, UK.

Appendix A

Tables A.1 to A.4 are provided as a reference showing the experimental results for each of the diversity measures when used on ensembles created by bagging, random forests, random trees, and random subspaces. The data generated in these tables was used to create the graphs in Figures 1—4 shown previously. Decreasing values of Q and Kappa correspond to higher diversity whereas PCDM increases as diversity increases.

Table A.1
Accuracy boost and diversity results for bagging ranked by the accuracy boost

Dataset	Pruning?	Accuracy Boost	PCDM	K	Q
Led-24	Unpruned	14.02%	0.707	0.394	0.690
Autos	Pruned	12.63%	0.652	0.339	0.661
Autos	Unpruned	12.57%	0.648	0.342	0.667
Glass	Pruned	9.44%	0.677	0.371	0.636
Waveform	Unpruned	9.24%	0.644	0.270	0.571
Waveform	Pruned	9.13%	0.639	0.273	0.576
Letter	Unpruned	9.08%	0.316	0.440	0.850
Letter	Pruned	8.82%	0.311	0.446	0.856
Glass	Unpruned	8.58%	0.677	0.367	0.632
Led-24	Pruned	8.34%	0.461	0.552	0.861
Satimage	Unpruned	6.74%	0.360	0.411	0.816
Satimage	Pruned	6.36%	0.342	0.426	0.831
Credit-a	Unpruned	6.15%	0.394	0.416	0.769
Credit-g	Unpruned	5.90%	0.726	0.302	0.571
Credit-g	Pruned	5.58%	0.634	0.371	0.674
Heart-c	Unpruned	5.31%	0.581	0.333	0.615
Breast-y	Unpruned	4.63%	0.683	0.357	0.632
Heart-c	Pruned	4.30%	0.545	0.365	0.659
Phoneme	Unpruned	4.28%	0.338	0.425	0.824
Phoneme	Pruned	4.22%	0.334	0.429	0.827
Horse-Colic	Unpruned	4.09%	0.324	0.516	0.845
Breast-y	Pruned	3.64%	0.397	0.570	0.849
Ion	Unpruned	3.30%	0.200	0.422	0.918
Horse-Colic	Pruned	3.15%	0.192	0.661	0.929
Ion	Pruned	3.15%	0.200	0.421	0.917
Pendigits	Unpruned	2.81%	0.111	0.348	0.903
Credit-a	Pruned	2.77%	0.272	0.527	0.871
Pendigits	Pruned	2.69%	0.109	0.354	0.907
Breast-w	Unpruned	2.63%	0.153	0.510	0.905
Oil	Unpruned	2.59%	0.109	0.453	0.922
Oil	Pruned	2.13%	0.100	0.487	0.939
Heart-h	Unpruned	2.01%	0.403	0.486	0.813
Breast-w	Pruned	1.96%	0.127	0.556	0.907
Segmentation	Pruned	1.74%	0.114	0.454	0.912
Segmentation	Unpruned	1.67%	0.116	0.450	0.909
Heart-h	Pruned	1.35%	0.377	0.520	0.841
Page	Unpruned	1.13%	0.061	0.589	0.979
Page	Pruned	0.91%	0.056	0.607	0.982
Anneal	Pruned	0.85%	0.037	0.285	0.974
Anneal	Unpruned	0.82%	0.035	0.291	0.979
Iris	Unpruned	0.64%	0.144	0.420	0.867
Iris	Pruned	0.53%	0.143	0.418	0.836
Hypo	Unpruned	0.43%	0.023	0.522	0.994
Hypo	Pruned	0.20%	0.013	0.600	0.992

Table A.2
Accuracy boost and diversity results for random forests ranked by the accuracy boost

Dataset	Pruning?	Accuracy Boost	PCDM	K	Q
Autos	Pruned	20.55%	0.819	0.259	0.495
Autos	Unpruned	17.44%	0.781	0.273	0.515
Led-24	Unpruned	15.35%	0.742	0.376	0.666
Letter	Unpruned	14.89%	0.440	0.355	0.742
Letter	Pruned	14.74%	0.436	0.358	0.746
Glass	Pruned	14.05%	0.777	0.302	0.547
Glass	Unpruned	13.75%	0.773	0.299	0.542
Waveform	Unpruned	11.50%	0.719	0.217	0.476
Waveform	Pruned	11.48%	0.716	0.219	0.480
Credit-g	Unpruned	10.16%	0.856	0.220	0.440
Heart-c	Unpruned	9.01%	0.623	0.299	0.570
Breast-y	Unpruned	8.99%	0.697	0.327	0.590
Credit-a	Unpruned	8.73%	0.558	0.331	0.663
Led-24	Pruned	8.46%	0.446	0.546	0.856
Satimage	Unpruned	8.43%	0.395	0.367	0.767
Horse-Colic	Unpruned	8.33%	0.546	0.348	0.669
Heart-c	Pruned	8.15%	0.590	0.324	0.612
Satimage	Pruned	7.90%	0.380	0.382	0.784
Ion	Unpruned	6.31%	0.364	0.277	0.754
Ion	Pruned	6.21%	0.358	0.281	0.762
Credit-g	Pruned	6.05%	0.719	0.306	0.586
Credit-a	Pruned	5.41%	0.404	0.404	0.771
Pendigits	Unpruned	5.05%	0.179	0.227	0.765
Pendigits	Pruned	4.97%	0.176	0.230	0.771
Breast-w	Unpruned	4.91%	0.234	0.310	0.803
Breast-y	Pruned	4.68%	0.438	0.509	0.792
Phoneme	Unpruned	4.63%	0.353	0.411	0.811
Phoneme	Pruned	4.62%	0.350	0.414	0.814
Breast-w	Pruned	4.24%	0.189	0.342	0.818
Segmentation	Unpruned	4.18%	0.166	0.315	0.822
Segmentation	Pruned	3.98%	0.161	0.327	0.834
Horse-Colic	Pruned	3.86%	0.297	0.523	0.840
Heart-h	Unpruned	2.49%	0.433	0.431	0.748
Oil	Unpruned	2.15%	0.106	0.420	0.893
Heart-h	Pruned	1.58%	0.403	0.445	0.753
Anneal	Unpruned	1.45%	0.040	0.231	0.991
Oil	Pruned	1.43%	0.087	0.467	0.920
Anneal	Pruned	1.39%	0.040	0.291	0.989
Page	Unpruned	1.37%	0.078	0.517	0.965
Page	Pruned	1.19%	0.073	0.530	0.968
Hypo	Unpruned	0.70%	0.037	0.448	0.946
Iris	Unpruned	0.56%	0.133	0.394	-------
Iris	Pruned	0.54%	0.133	0.383	-------
Hypo	Pruned	0.45%	0.023	0.539	0.962

Table A.3
Accuracy boost and diversity results for random trees ranked by the accuracy boost

Dataset	Pruning?	Accuracy Boost	PCDM	K	Q
Led-24	Unpruned	13.48%	0.679	0.413	0.715
Letter	Unpruned	11.51%	0.360	0.352	0.776
Letter	Pruned	11.47%	0.361	0.355	0.778
Autos	Pruned	9.33%	0.529	0.383	0.687
Autos	Unpruned	8.83%	0.424	0.452	0.784
Credit-g	Unpruned	7.39%	0.775	0.277	0.535
Glass	Unpruned	7.16%	0.586	0.411	0.708
Satimage	Unpruned	7.13%	0.355	0.388	0.802
Glass	Pruned	7.10%	0.577	0.418	0.716
Waveform	Unpruned	6.73%	0.542	0.354	0.698
Waveform	Pruned	6.60%	0.536	0.359	0.705
Satimage	Pruned	6.07%	0.318	0.424	0.839
Credit-a	Unpruned	5.17%	0.417	0.444	0.792
Breast-y	Unpruned	5.11%	0.528	0.499	0.804
Horse-Colic	Unpruned	4.82%	0.419	0.454	0.804
Heart-c	Unpruned	4.44%	0.519	0.406	0.721
Heart-c	Pruned	4.16%	0.442	0.469	0.788
Breast-w	Unpruned	4.03%	0.214	0.364	0.842
Breast-w	Pruned	3.55%	0.181	0.432	0.744
Ion	Pruned	3.55%	0.219	0.433	0.916
Ion	Unpruned	3.24%	0.217	0.422	0.901
Pendigits	Unpruned	2.82%	0.104	0.280	0.876
Led-24	Pruned	2.77%	0.239	0.736	0.963
Pendigits	Pruned	2.72%	0.101	0.293	0.885
Credit-g	Pruned	2.65%	0.471	0.502	0.812
Segmentation	Pruned	2.45%	0.103	0.385	0.906
Segmentation	Unpruned	2.38%	0.102	0.382	0.907
Horse-Colic	Pruned	1.73%	0.176	0.649	0.912
Oil	Unpruned	1.69%	0.090	0.507	0.941
Credit-a	Pruned	1.54%	0.228	0.576	0.900
Breast-y	Pruned	1.44%	0.321	0.673	0.915
Phoneme	Unpruned	1.30%	0.200	0.643	0.952
Phoneme	Pruned	1.24%	0.195	0.648	0.953
Oil	Pruned	1.09%	0.064	0.604	0.979
Heart-h	Unpruned	0.97%	0.263	0.649	0.915
Anneal	Pruned	0.92%	0.032	0.285	-------
Page	Unpruned	0.90%	0.055	0.614	0.984
Page	Pruned	0.81%	0.049	0.634	0.987
Iris	Pruned	0.79%	0.100	0.423	0.933
Anneal	Unpruned	0.78%	0.025	0.190	-------
Heart-h	Pruned	0.48%	0.210	0.690	0.929
Hypo	Unpruned	0.30%	0.021	0.587	0.992
Hypo	Pruned	0.26%	0.009	0.678	0.998
Iris	Unpruned	-0.03%	0.114	0.458	0.806

Table A.4
Accuracy boost and diversity results for random subspaces ranked by the accuracy boost

Dataset	Pruning?	Accuracy Boost	PCDM	K	Q
Led-24	Unpruned	32.67%	0.935	0.140	0.304
Led-24	Pruned	31.08%	0.938	0.169	0.344
Letter	Unpruned	19.77%	0.565	0.313	0.668
Letter	Pruned	19.04%	0.547	0.321	0.682
Glass	Unpruned	18.71%	0.818	0.264	0.500
Glass	Pruned	18.52%	0.818	0.269	0.508
Credit-g	Unpruned	11.82%	0.922	0.182	0.367
Credit-a	Unpruned	11.71%	0.710	0.231	0.506
Waveform	Unpruned	11.18%	0.716	0.235	0.498
Waveform	Pruned	10.88%	0.700	0.243	0.514
Heart-c	Unpruned	10.69%	0.765	0.224	0.433
Horse-Colic	Unpruned	10.58%	0.630	0.283	0.574
Autos	Pruned	10.07%	0.486	0.361	0.760
Credit-a	Pruned	9.69%	0.612	0.278	0.584
Heart-c	Pruned	9.67%	0.703	0.258	0.493
Autos	Unpruned	9.57%	0.448	0.398	0.807
Credit-g	Pruned	8.45%	0.777	0.273	0.532
Horse-Colic	Pruned	7.44%	0.503	0.376	0.695
Heart-h	Unpruned	7.38%	0.530	0.355	0.642
Satimage	Unpruned	6.99%	0.343	0.397	0.811
Pendigits	Unpruned	6.42%	0.222	0.205	0.694
Heart-h	Pruned	6.35%	0.467	0.391	0.679
Pendigits	Pruned	6.28%	0.215	0.212	0.707
Satimage	Pruned	6.24%	0.309	0.432	0.845
Anneal	Pruned	6.12%	0.213	0.188	0.803
Anneal	Unpruned	6.10%	0.212	0.160	0.810
Iris	Unpruned	5.63%	0.300	0.291	0.637
Breast-y	Unpruned	5.31%	0.655	0.346	0.606
Ion	Pruned	4.88%	0.242	0.331	0.838
Segmentation	Pruned	4.78%	0.203	0.317	0.801
Segmentation	Unpruned	4.65%	0.202	0.314	0.800
Iris	Pruned	4.57%	0.280	0.291	0.356
Breast-w	Unpruned	4.36%	0.213	0.336	0.730
Ion	Unpruned	4.23%	0.239	0.329	0.837
Phoneme	Unpruned	4.01%	0.438	0.427	0.753
Phoneme	Pruned	3.98%	0.432	0.438	0.767
Breast-w	Pruned	3.36%	0.180	0.383	0.735
Breast-y	Pruned	2.52%	0.407	0.573	0.839
Page	Unpruned	1.39%	0.078	0.520	0.966
Oil	Unpruned	1.36%	0.085	0.551	0.954
Hypo	Unpruned	1.27%	0.067	0.368	0.917
Page	Pruned	1.19%	0.070	0.547	0.972
Oil	Pruned	1.08%	0.063	0.614	0.970
Hypo	Pruned	1.04%	0.061	0.405	0.931