Decision Tree Ensemble: Small Heterogeneous Is Better Than Large Homogeneous

Mike Gashler, Christophe Giraud-Carrier and Tony Martinez
Brigham Young University
Department of Computer Science
Provo, UT, U.S.A.
mikegashler@gmail.com,cgc,martinez@cs.byu.edu

Abstract

Using decision trees that split on randomly selected attributes is one way to increase the diversity within an ensemble of decision trees. Another approach increases diversity by combining multiple tree algorithms. The random forest approach has become popular because it is simple and yields good results with common datasets. We present a technique that combines heterogeneous tree algorithms and contrast it with homogeneous forest algorithms. Our results indicate that random forests do poorly when faced with irrelevant attributes, while our heterogeneous technique handles them robustly. Further, we show that large ensembles of random trees are more susceptible to diminishing returns than our technique. We are able to obtain better results across a large number of common datasets with a significantly smaller ensemble.

1 Introduction

Ensembles offer a simple yet effective technique for obtaining increased levels of predictive accuracy by combining the predictions of many different learning algorithm instances [12, 17, 9, 19]. However, such improvements are predicated upon there existing some form of diversity among the elements of the ensemble [21, 15]. Indeed, if every instance in the ensemble behaves nearly the same way, little is achieved by combining their predictions.

Decision trees are particularly well-suited for ensembles because they are fast and unstable. Hence, it is often possible to create synergy within decision tree ensembles or forests. A popular technique for promoting variance in decision tree forests is to use trees that randomly choose on which attributes to divide the data [13]. In sufficiently large ensembles, this technique can yield better accuracy than standard entropy-reducing decision trees on many datasets because it creates more variance within the models. Breiman showed that bagging is particularly effective with forests of random decision trees [4]. On the downside, however, our results indicate that random trees yield poor results on data with many irrelevant attributes.

In this paper, we contrast the random forest algorithm with another ensemble technique that combines multiple tree algorithms using cross-validation selection. We show that this technique both has a higher ceiling of diminishing returns and is more robust to irrelevant features than homogeneous tree ensembles. Heterogeneity in our ensembles is achieved through a combination of entropy-reducing decision trees, which build axis aligned decision boundaries, and a new class of decision trees, known as mean margins decision trees (MMDT), which build oblique decision boundaries.

The paper is organized as follows. Section 2 briefly reviews significant related work. In section 3, we describe the mean margins decision tree learning algorithm and the hybrid cross-validation decision tree learning algorithm which may be obtained by combining the mean margins decision tree learning algorithm with the standard entropy-reducing decision tree learning algorithm. Section 4 presents a thorough analysis of the resulting heterogeneous ensemble learning algorithm. Finally, section 5 concludes the paper.

2 Related Work

The MMDT algorithm we introduce here uses linear combinations of inputs to define the decision boundaries of its induced model trees. Such trees were first discussed in [5] and later implemented in a number of algorithms, such as Multivariate Decision Trees [6], Oblique Decision Trees [16], and Perceptron Decision Trees [22].

Linear combination trees offer significant flexibility over trees that only divide data with axis-aligned boundaries. Unfortunately, this flexibility tends to be a hindrance more

than a benefit. Training a perceptron tree, for example, involves optimization in a very non-convex heuristic space. Further, there is a strong tendency for perceptron trees to use their extreme flexibility to overfit the training data. MMDT, on the other hand, is asymptotically as efficient as the well-known entropy-reducing decision tree learning algorithm. It is also parameterless and tends to produce good results with many datasets.

Although a complete review is outside the scope of this paper, much of the research involving ensemble methods is clearly relevant to our work (e.g., see [9, 7, 11, 10]). Of particular interest is work on diversity in ensembles. The need for diversity in ensemble is well known and has been the object of many studies. Many techniques have been proposed from bagging [3] to stacking [23] to mixture of experts [14] to random forests [4], to COD-based approaches [18], to name only a few. A recent survey of techniques for creating diversity in ensemble is in [7]. We use the term *homogeneous* to refer to techniques that use a single algorithm and achieve diversity through some form of variability in the data (e.g., randomization), and we use the term *heterogeneous* to refer to techniques that achieve diversity through the use of multiple algorithms. A thorough comparison of randomization-based decision tree ensemble methods is in [2]. Our approach is heterogeneous and is compared with one well-known homogeneous approach, namely random forests.

3 Mean Margins Decision Tree Learning

We first describe a new decision tree learning algorithm called Mean Margins Decision Tree (MMDT) learning, which builds oblique decision boundaries. The MMDT algorithm is designed to be simple, efficient, and free of parameters. It is, therefore, well-suited for use in ensembles. Suppose we have a set of patterns, P, for a binary classification problem such that P_T is the subset of patterns of class $true$, and P_F is the subset of patterns of class $false$. Further, suppose all patterns are vectors of real values.

At a high level, an MMDT is constructed in a manner very similar to that of any other decision tree, as shown in Figure 1. The difference is that the MMDT algorithm chooses decision boundaries in the form of linear combinations of inputs that maximize the margins between the means of P_T and P_F, as illustrated in Figure 2.

Of course, not all classification tasks have exactly two classes and only real-valued inputs. However, the MMDT algorithm is easily extended to handle nominal attributes, and any number of classes. To handle nominal attributes, we represent each value as an orthogonal dimension. For example, suppose some nominal attribute ranges over the values {red, green, blue}, and some pattern contains the value $v =$ red. We would represent this value with three real values $< 1, 0, 0 >$. These values may be thought of as a categorical distribution of confidence over the nominal values. (Note that this technique also naturally provides a mechanism for handling missing nominal values: just assign equal confidences to each value, e.g., $< 0.33, 0.33, 0.33 >$.) To convert back to a nominal value, just find the value with the maximum confidence.

```
function build_tree(P)
    μ⃗, ν⃗ ← choose_decision_boundary(P)
    P_left, P_right ← divide_data(μ⃗, ν⃗, P)
    if |P_left| == 0
        return new LeafNode(P_right)
    if |P_right| == 0
        return new LeafNode(P_left)
    node_left ← build_tree(P_left)
    node_right ← build_tree(P_right)
    return new InteriorNode(node_left, node_right)

function choose_decision_boundary(P)
    μ⃗_F ← 1/|P_F| Σ_{p⃗∈P_F} p⃗
    μ⃗_T ← 1/|P_T| Σ_{p⃗∈P_T} p⃗
    μ⃗ ← (μ⃗_F + μ⃗_T)/2
    ν⃗ ← μ⃗_T − μ⃗_F
    return μ⃗, ν⃗

function divide_data(μ⃗, ν⃗, P)
    P_left ← {}
    P_right ← {}
    for each p⃗ ∈ P
        if (p⃗ − μ⃗) · ν⃗ ≥ 0
            P_right ← P_right + p⃗
        else
            P_left ← P_left + p⃗
    return P_left, P_right
```

Figure 1. MMDT Learning Algorithm

To handle more than two classes, we use the following technique. If class labels are nominal, they are also converted to real vectors as per the above procedure, such that L is the set of real vector labels (one for each pattern in P). Each time before *choose_decision_boundary* is called, we first divide P into P_T and P_F in the following manner:

1. Compute the mean $\vec{\mu}_L$ and first principal component $\vec{\nu}_L$ of L.
2. For each $\vec{l} \in L$, if $(\vec{l} - \vec{\mu}_L) \cdot \vec{\nu}_L \geq 0$ then P_T contains patterns with label $\vec{l}$, otherwise P_F contains patterns with label $\vec{l}$.

Thus, the MMDT algorithm can easily work with multiple

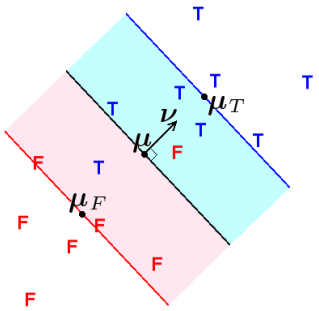

Figure 2. Margin Maximization

function compute_1st_principal_component(L)
  for i = 1 to d
    $\vec{\nu}_{Li} \leftarrow random\_standard\_normal()$
  $\vec{\nu}_L \leftarrow \frac{\vec{\nu}_L}{|\vec{\nu}_L|}$
  do 10 times
    $\vec{\alpha} \leftarrow 0_d$
    for each $\vec{l} \in L$
      $\vec{\alpha} \leftarrow \vec{\alpha} + ((\vec{l} - \vec{\mu}_L) \cdot \vec{\nu}_L)(\vec{l} - \vec{\mu}_L)$
    $\vec{\nu}_L \leftarrow \frac{\vec{\alpha}}{|\vec{\alpha}|}$

Figure 3. Computing 1^{st} Principal Component

classes, including continuous labels. For completeness, the pseudocode, derived from [20], to quickly compute the first principal component about the mean of L in d dimensions is shown in Figure 3. In some rare cases, more iterations may be required to obtain a precise estimate of the first principal component, but for this algorithm an imprecise estimate will work just fine, so ten iterations are sufficient.

As an illustration of MMDT's performance in complex environments, we design the following simple interpolation task. Of course, better techniques for image interpolation exist. The purpose here is only to assist an intuition of the workings of MMDT. We create a training set with one pattern per pixel from a small 20x20 pixel image. We then evaluate at sub-pixel positions to interpolate a 160x160 image. We compare bagged MMDT with a random forest of standard entropy-reducing decision trees (ERDTs). The results are shown in Figure 4. Note how bagged MMDT is better able to follow non-axis-aligned contours.

As it turns out, the MMDT algorithm is not as effective overall as ERDT learning for common classification tasks, but MMDT tends to perform well in many cases where ERDT does poorly. On a set of 43 common datasets from the UCI repository [1], ERDT achieves higher predictive accuracy (measured by 5x2 cross-validation) than MMDT on almost two thirds (26 out of 43) of the datasets. However,

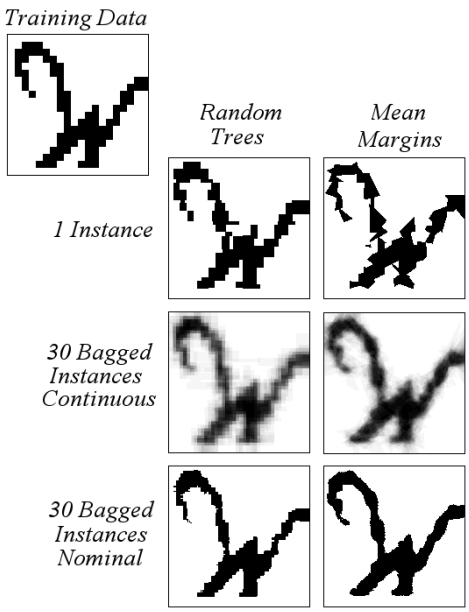

Figure 4. MMDT vs RDT Interpolation

MMDT appears to cover an important deficiency in the remaining one third. MMDT computes mean values in order to choose its decision boundaries. Mean values can be estimated with more accuracy when there is plenty of data. It seems intuitive, therefore, that the MMDT algorithm might do well with datasets that densely sample their input space.

To test this notion, we compute the sample density of each dataset as the product of the arity of each attribute divided by the number of patterns. Since there is no concept of arity with continuous values, we use a value of 5 for continuous attributes, which is close to the average arity of nominal values in the datasets we consider. In a pairwise comparison between bagged MMDT (size 100) and bagged ERDT (size 100) across only the densest half of the datasets, MMDT performs better than ERDT on 45.5% (10 out of 22) of the datasets. On the densest quarter, MMDT does best on 72.7% (8 out of 11) of the datasets, and on the densest eighth, it also does best on 83.3% (5 out of 6) of the datasets. Hence, sample density does seem to characterize a significant portion of the strength of MMDT. There also remain 7 out of 21 datasets (33.3%) among the sparsest half of the datasets on which bagged MMDT outperforms bagged ERDT.

4 Decision Tree Ensemble Learning

We will refer to a bagged ensemble of 100 random decision tree (RDT) instances as "100×RDT", to a bagged ensemble of 100 ERDT instances as "100×ERDT", and like-

wise for other algorithms.

We first compare the predictive accuracy of three bagged ensembles: 100×RDT, 100×ERDT, and 100×MMDT. We measure predictive accuracy on 43 common datasets from the UCI repository [1] using 5x2 cross-validation (5x2CV). Our choice of 5x2CV, rather than the somewhat more popular 10-fold cross-validation, is motivated by recent results which suggest that 5x2CV yields lower type II error than 10-fold cross-validation [8]. In this experiment, each ensemble is homogeneous in that it contains multiple instances of just one algorithm. The results are shown in Table 1. The column "Baseline" corresponds to a majority learner that chooses the most common class. It is shown in order to contrast the effectiveness of the various ensemble techniques. 100×RDT is the most accurate with the most (17) datasets. RDTs are particularly effective at creating diversity within the ensemble, so this result emphasizes the importance of having model diversity. 100×ERDT does best on 13 datases, while 100×MMDT does best on 12 datasets. One dataset has no clear winner.

Given the complementary nature of the strengths of ERDT and MMDT as discussed above, it would seem that performance could be further improved by building heterogeneous ensembles. Rather than combining several ERDTs and MMDTs, we first design the following simple, cross-validation-based decision tree learning algorithm, which we refer to as CVDT (Cross-Validation Decision Tree).

1. Perform 1x2CV on the training set with ERDT
2. Perform 1x2CV on the training set with MMDT
3. Select the algorithm that performed best and train on full training set

We then create heterogeneous ensembles of CVDTs through bagging. When one of the two algorithms (ERDT or MMDT) is clearly better than the other for a particular problem, this is equivalent to a bagged ensemble of that model. When both algorithms achieve similar accuracy, the bagged ensemble will contain a mixture of both algorithms in proportion to the number of times that each achieved better accuracy during cross-validation.

In and of itself, building a heterogeneous ensemble is not that novel, and although training a CVDT requires more computation than training an ERDT or an MMDT, this cost is only required at training time. Evaluation with a CVDT is as efficient as the model that it selects. Furthermore, we will show shortly that, in the context of ensemble learning, significantly smaller ensembles of CVDTs may be used, that achieve higher accuracy and better tolerance to noise than much larger ensembles of ERDTs or random forests, with an overall smaller computational footprint.

We compare ensembles of ERDTs and RDTs with ensembles of bagged CVDTs. A simple analogy motivates this design. A bagged ensemble of ERDTs may be analogous to a panel of expert medical doctors that all graduated from the same university. An ensemble of RDTs may be analogous to a panel of novices that dropped out of medical schools from all over the world. Even though each novice may have less talent than any one of the experts, the diversity in this group may enable them to produce a better combined diagnosis. This may explain why ensembles of RDTs can outperform ensembles of ERDTs. It would seem, therefore, that an ideal panel of medical doctors would contain both a significant amount of diversity *and* expert talent. We seek this balance by using a bagged ensemble of CVDTs. Each CVDT contains only algorithms that build their model with deliberate divisions, but diversity is also enhanced (in addition to the diversity injected as part of the bagging ensemble technique) by the utilization of more than one algorithm.

Dataset	***Baseline***	***100x RDT***	***100x ERDT***	***100x MMDT***
colic	0.663	**0.684**	0.637	0.655
balance-scale	0.453	0.846	0.816	**0.877**
segment	0.130	0.966	**0.973**	0.845
breast-cancer	0.703	**0.714**	0.667	0.711
anneal	0.762	0.762	0.762	**0.848**
diabetes	0.651	0.749	**0.758**	0.707
primary-tumor	0.248	**0.419**	0.412	0.413
breast-w	0.655	**0.970**	0.961	0.959
soybean	0.117	0.868	0.810	**0.914**
hepatitis	0.794	**0.831**	0.822	0.775
kropt	0.162	0.630	0.535	**0.652**
badges2	0.714	0.993	**1.000**	0.824
nursery	0.330	0.961	0.971	**0.981**
waveform-5000	0.333	0.825	**0.836**	0.829
heart-c	0.525	**0.829**	0.777	0.583
vowel	0.167	**0.931**	0.874	0.921
zoo	0.406	**0.931**	0.883	0.915
iris	0.289	0.948	0.941	**0.956**
kr-vs-kp	0.522	0.962	**0.989**	0.973
labor	0.583	**0.864**	0.850	0.625
titanic	0.677	0.782	**0.785**	0.784
audiology	0.243	0.615	0.501	**0.717**
credit-a	0.555	**0.865**	0.824	0.648
splice	0.519	0.765	0.934	**0.936**
vehicle	0.240	0.724	**0.746**	0.579
glass	0.534	**0.825**	0.817	0.621
ionosphere	0.641	0.920	0.913	**0.931**
mushroom	0.518	1.000	1.000	1.000
hypothyroid	0.923	0.957	**0.967**	0.923
cars	0.700	0.851	0.906	**0.941**
wine	0.370	0.982	**0.949**	0.674
lymphoma	0.479	0.633	**0.725**	0.604
lenses	0.625	0.608	0.675	**0.758**
heart-statlog	0.527	**0.821**	0.813	0.601
autos	0.327	**0.716**	0.640	0.364
spambase	0.606	**0.949**	0.944	0.727
credit-g	0.700	0.720	**0.723**	0.612
vote	0.614	0.945	**0.956**	0.947
heart-h	0.639	**0.811**	0.793	0.639
lymph	0.547	**0.826**	0.770	0.764
sick	0.939	0.964	**0.971**	0.939
colon	0.645	0.703	0.752	**0.816**
sonar	0.480	**0.781**	0.780	0.773

Table 1. Homogeneous Ensembles

Here, we consider bagged ensembles of 1,000 ERDTs and 1,000 RDTs with bagged ensembles of only 100 CVDTs. The choice of 1,000 for ensembles of ERDTs and RDTs is rather standard (e.g., see [2]). The results are shown in Table 2.

Dataset	Baseline	1000x ERDT	1000x RDT	100x CVDT	Improvement Over ERDT	Improvement Over RDT
titanic	0.677	0.788	0.787	0.785	-0.461%	-0.323%
nursery	0.330	0.973	0.965	0.973	-0.019%	0.746%
lenses	0.625	0.725	0.717	0.725	0.000%	1.163%
cars	0.700	0.896	0.856	0.910	1.641%	6.398%
balance-scale	0.453	0.819	0.852	0.870	6.214%	2.064%
iris	0.289	0.948	0.948	0.959	1.125%	1.125%
kropt	0.162	0.537	0.653	0.556	3.586%	-14.757%
vote	0.614	0.958	0.949	0.953	-0.528%	0.438%
diabetes	0.651	0.756	0.754	0.752	-0.551%	-0.345%
primary-tumor	0.248	0.418	0.418	0.417	-0.289%	-0.289%
zoo	0.406	0.891	0.901	0.911	2.183%	1.084%
breast-cancer	0.703	0.657	0.710	0.709	7.987%	-0.197%
badges2	0.714	1.000	0.989	1.000	0.000%	1.100%
breast-w	0.655	0.961	0.970	0.960	-0.091%	-1.003%
glass	0.534	0.834	0.836	0.831	-0.431%	-0.593%
heart-c	0.525	0.759	0.827	0.767	1.038%	-7.262%
heart-h	0.639	0.802	0.816	0.798	-0.509%	-2.250%
vowel	0.167	0.898	0.933	0.919	2.408%	-1.430%
hepatitis	0.794	0.818	0.848	0.812	-0.800%	-4.251%
credit-a	0.555	0.834	0.869	0.833	-0.104%	-4.136%
lymph	0.547	0.764	0.826	0.792	3.717%	-4.092%
heart-statlog	0.527	0.816	0.830	0.810	-0.726%	-2.321%
wine	0.370	0.936	0.984	0.951	1.561%	-3.425%
labor	0.583	0.846	0.873	0.818	-3.292%	-6.373%
kr-vs-kp	0.522	0.990	0.968	0.991	0.101%	2.425%
hypothyroid	0.923	0.970	0.955	0.971	0.022%	1.621%
sick	0.939	0.971	0.964	0.971	0.027%	0.770%
anneal	0.762	0.762	0.762	0.831	9.094%	9.094%
credit-g	0.700	0.726	0.724	0.712	-1.928%	-1.575%
vehicle	0.240	0.746	0.727	0.736	-1.331%	1.202%
segment	0.130	0.974	0.964	0.974	-0.071%	0.988%
sonar	0.480	0.793	0.804	0.795	0.242%	-1.077%
soybean	0.117	0.822	0.875	0.911	10.796%	4.047%
colic	0.663	0.643	0.685	0.638	-0.845%	-6.905%
autos	0.327	0.670	0.728	0.642	-4.211%	-11.786%
audiology	0.243	0.552	0.619	0.737	33.494%	19.170%
ionosphere	0.641	0.922	0.929	0.932	0.994%	0.248%
spambase	0.606	0.943	0.950	0.942	-0.065%	-0.837%
splice	0.519	0.930	0.743	0.939	0.937%	26.322%
mushroom	0.518	1.000	1.000	1.000	-0.022%	-0.030%
waveform-5000	0.333	0.839	0.838	0.830	-1.025%	-0.917%
colon	0.645	0.781	0.661	0.787	0.826%	19.024%
lymphoma	0.479	0.727	0.648	0.713	-2.006%	9.968%
Average					**1.597%**	**0.763%**

Table 2. Heterogeneous Ensemble vs. Homogeneous Ensembles

These results indicate that a bagged ensemble of CVDTs has a higher ceiling of diminishing returns than much larger ensembles of RDTs or ERDTs. Furthermore, despite having one tenth the size, the much smaller ensemble of $100\times$CVDT still yields somewhat higher accuracy on average, and wins outright over both $1,000\times$ERDT and $1,000\times$RDT on 15 of the 43 datastets. On the 16 datasets for which $100\times$CVDT looses out to both competitors, the loss is usually rather insignificant for at least one of them. Again, these results are obtained at a much lower computational cost since $100\times$CVDT builds only 500 models (200 ERDTs, 200 MMDTs and 100 CVDTs) rather than the 1,000 required by the other approaches.

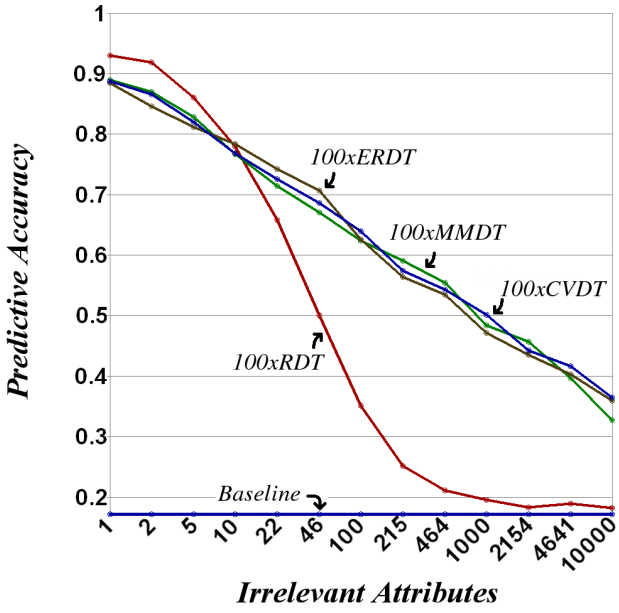

Figure 5. Irrelevant Attributes on Vowel

In addition to accuracy, we look at how well our proposed ensemble technique handles irrelevant attributes. This property is often ignored in the analysis of many algorithms because popular collections of data tend to contain only attributes that have a fairly significant degree of relevance to the output class. Irrelevant attributes, however, are becoming more and more prevalent as the ease with which data can be collected gives rise to a "let us collect everything we can and worry about its value later" kind of attitude in many machine learning and data mining applications.

For purposes of experimentation, irrelevant attributes are not difficult to generate. We inject varying numbers of attributes containing Gaussian noise into several common datasets. For each dataset, we measure the predictive accuracy of $100\times$RDT, $100\times$ERDT, $100\times$MMDT, and $100\times$CVDT. Figure 5 shows results with the vowel dataset. Other datasets yield very similar trends, so only vowel is shown here as a representative. Note that the horizontal axis is shown on a logarithmic scale, so the right side of the chart represents a broader domain than the left side.

Both $100\times$ERDT and $100\times$MMDT handle irrelevant attributes very well. Both algorithms exhibit a nearly linear decrease in accuracy with an exponential increase in the number of irrelevant attributes. It follows, as expected, that the CVDT algorithm, which selects between these two algorithms, also handles irrelevant attributes well. The accuracy of $100\times$RDT, on the other hand, begins to degrade very quickly after the majority of attributes are irrelevant with respect to class labels.

A common justification for randomly selecting decision boundaries is that this creates models with more variance, and if it randomly decides to split on an irrelevant feature, it may still split on a relevant feature deeper in the tree. As the number of irrelevant features becomes large, however, it constructs models with less and less total relevant information. Consequently, algorithms that identify relevant decision boundaries tend to handle irrelevant attributes better. We, therefore, suggest that utilizing a diversity of algorithms, such as the one proposed here, is a better technique for inducing variance within an ensemble.

5 Conclusion

Although RDT may be somewhat effective at producing desirable model variance within an ensemble, heterogeneous ensembles that select from among multiple models can outperform such homogeneous ensembles. Furthermore, ensembles of RDT are not robust to irrelevant attributes.

The MMDT algorithm introduced here is intuitive, efficient, simple to implement and parameterless. It is, therefore, well-suited for use in ensembles. Although, by itself, MMDT is often less accurate than ERDT, MMDT tends to do well with a different set of problems than ERDT. Using cross-validation selection between ERDT and MMDT creates a particularly powerful model. Our results demonstrate that very small ensembles of such cross-validation decision trees (100 vs. 1,000) can outperform very large homogeneous ensembles of RDT both in terms of accuracy and tolerance to irrelevant attributes.

References

[1] A. Asuncion and D. Newman. UCI machine learning repository, 2007.

[2] R. Banfield, L. Hall, K. Bowyer, and W. Kegelmeyer. A comparison of decision tree ensemble creation techniques. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 29(1):173–180, 2007.

[3] L. Breiman. Bagging predictors. *Machine Learning*, 24(2):123–140, 1996.

[4] L. Breiman. Random forests. *Machine Learning*, 45(1):5–32, 2001.

[5] L. Breiman, J. Friedman, R. Olshen, and C. Stone. *Classification and Regression Trees*. Wadsworth and Brooks, Monterey, CA, 1984.

[6] C. Brodley and P. Utgoff. Multivariate decision trees. *Machine Learning*, 19:45–77, 1995.

[7] G. Brown, J. Wyatt, R. Harris, and X. Yao. Diversity creation methods: a survey and categorisation. *Information Fusion*, 6(1):5–20, 2005.

[8] J. Demsar. Statistical comparisons of classifiers over multiple data sets. *Journal of Machine Learning Research*, 7:1–30, 2006.

[9] T. Dietterich. Ensemble methods in machine learning. *Lecture Notes in Computer Science*, 1857:1–15, 2000.

[10] T. Dietterich. Ensemble learning. In M. A. Arbib, editor, *The Handbook of Brain Theory and Neural Networks, Second edition*, pages 405–408. MIT Press, 2002.

[11] Y. Freund and R. Schapire. A decision-theoretic generalization of on-line learning and an application to boosting. In *European Conference on Computational Learning Theory*, pages 23–37, 1995.

[12] L. Hansen and P. Salamon. Neural network ensembles. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 12(10):993–1001, 1990.

[13] T. Ho. Random decision forests. In *Proceedings of the Third International Conference on Document Analysis and Recognition*, pages 278–282, Los Alamitos, CA, USA, 1995. IEEE Computer Society.

[14] R. Jacobs, M. Jordan, S. Nowlan, and G. E. Hinton. Adaptive mixture of local experts. *Neural Computation*, 3(1):79–87, 1991.

[15] L. Kuncheva and C. Whitaker. Measures of diversity in classifier ensembles. *Machine Learning*, 51:181–207, 2003.

[16] S. Murthy, S. Kasif, and S. Salzberg. A system for induction of oblique decision trees. *Journal of Artificial Intelligence Research*, 2:1–32, 1994.

[17] D. Opitz and R. Maclin. Popular ensemble methods: An empirical study. *Journal of Artificial Intelligence Research*, 11:169–198, 1999.

[18] A. Peterson and T. Martinez. Estimating the potential for combining learning models. In *Proceedings of the ICML Workshop on Meta-Learning*, pages 68–75, 2005.

[19] R. Polikar. Ensemble based systems in decision making. *IEEE Circuits and Systems Magazine*, 6:21–45, 2006.

[20] S. Roweis. Em algorithms for PCA and SPCA. In M. I. Jordan, M. J. Kearns, and S. A. Solla, editors, *Advances in Neural Information Processing Systems*, volume 10, 1998.

[21] P. Sollich and A. Krogh. Learning with ensembles: How overfitting can be useful. In D. Touretzky, M. Mozer, and M. Hasselmo, editors, *Advances in Neural Information Processing Systems*, volume 8, pages 190–196. The MIT Press, 1996.

[22] P. Utgoff. Perceptron trees: A case study in hybrid concept representations. In *Connection Science*, volume 1, pages 377–391, 1989.

[23] D. Wolpert. Stacked generalization. *Neural Networks*, 5(2):241–259, 1992.