
On Pruning and Averaging Decision Trees

Jonathan J. Oliver
Computer Science Dept.
Monash University
Clayton, Vic. 3168, Australia
jono@cs.monash.edu.au

David J. Hand
Statistics Dept.
Open University
Milton Keynes, MK7 6AA, U.K.
D.J.Hand@open.ac.uk

Abstract

Pruning a decision tree is considered by some researchers to be the most important part of tree building in noisy domains. While, there are many approaches to pruning, an alternative approach of averaging over decision trees has not received as much attention. We perform an empirical comparison of pruning with the approach of averaging over decision trees. For this comparison we use a computationally efficient method of averaging, namely averaging over the *extended fanned set* of a tree. Since there are a wide range of approaches to pruning, we compare tree averaging with a traditional pruning approach, along with an optimal pruning approach.

1 INTRODUCTION

A wide variety of splitting rules have been described in the decision tree literature such as the GINI index of diversity (Breiman et al. 1984), Information Gain (Quinlan 1986), and Minimum Encoding splitting criteria (Quinlan 1989, Wallace and Patrick 1993). There have been a number of comparisons of splitting rules (e.g., Mingers 1989 and Buntine and Niblett 1992).

In noisy domains, authors such as Breiman et al. (1984) and Gelfand et al. (1991) consider pruning to be the most important part of tree building. An alternative to pruning is to average over a set of trees (Buntine 1990, 1992, Clark and Pregibon 1992, Hastie and Pregibon 1990, Kwok and Carter 1990, Oliver and Hand 1994a, 1996). Averaging has also been found to improve the accuracy of classifiers of other forms (Bahl et al. 1989, Kononenko 1992). In this paper, we perform an empirical comparison of pruning and averaging over trees. In this comparison, we compare pessimistic pruning as used by C4.5 (Quinlan 1993), an *optimal pruning strategy*, and a computationally efficient method of averaging, namely averaging over the *extended fanned set* of a tree.

2 OPTIMAL PRUNING

It is common practice to grow a *complete* tree, and then prune it back. A complete tree is grown by recursively splitting each leaf until each leaf is either class pure, or considered too small to split. Given a complete tree T we consider the *set of pruned trees* of T ,

$$\{ Pt_1, Pt_2, \dots, Pt_n \}.$$

The set of pruned trees contains each tree that has 0 or more decision nodes transformed into leaves. For example, Figure 1 gives the set of pruned trees for the complete tree Pt_1 . Pruning involves finding the single “best” tree from the set of pruned trees according to some pruning criterion.

We define *optimal pruning* to be the process of evaluating the error rate on a test set according to every tree in the pruned set, and selecting the tree with minimal error rate. This definition of optimal pruning, is optimal pruning with respect to a given splitting rule. An optimal pruning approach cannot be achieved in practice; it provides a lower bound on the error rate for pruning approaches.

3 TREE AVERAGING

The basic idea of *tree averaging* is to use a set of trees to classify new objects, rather than use the single “best” tree (as the pruning approach would do). Methods for using sets of trees to classify new objects have been suggested by Bahl et al. (1989), Buntine (1990, 1992), Kwok and Carter (1990), Hastie and Pregibon (Hastie 1990), Clark and Pregibon (1992), Quinlan (1992) (in the case of Model Trees) and Oliver and Hand (1994a, 1996).

Procedure Average classifies a new object, O , by averaging over a set of trees $S = \{ t_1, t_2, \dots, t_n \}$ using a set of weights $W = \{ w_1, w_2, \dots, w_n \}$. Each weight, w_i , reflects how well t_i explains the training set. New objects are classified by estimating $P(c | O)$,

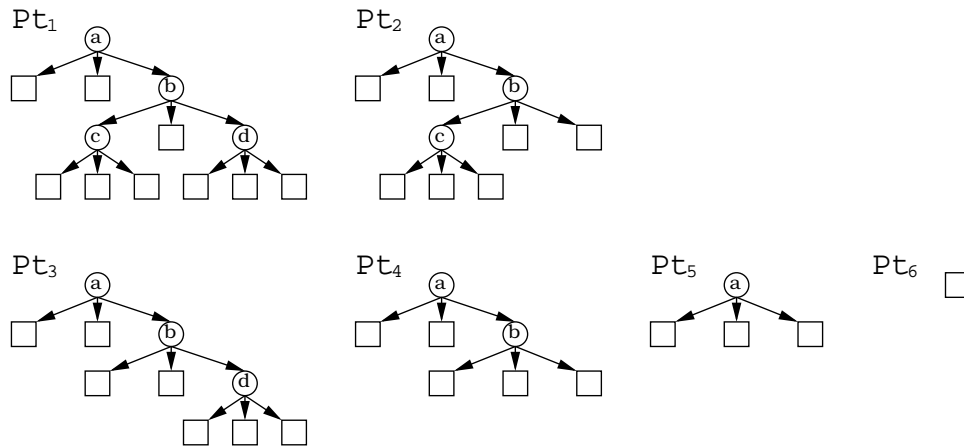


Figure 1: The Set of Pruned Trees

the probability that an object, O , will have class c . To estimate this probability, we take the weighted sum of the probability distributions given by each tree in S . A number of approaches have been used to estimate weights. For example, Buntine (1990, 1992) used a Bayesian approach to calculate these weights; Bahl et al. (1989) set aside data independent of the data used to grow the tree to estimate the weights.

Procedure Average (O, S, W)

1. Determine $P(c | O, t_i)$, the probability which tree t_i gives object O of having class c .
2. Define O to have a probability of belonging to class c of:

$$P(c | O) = \sum_{i=1}^n w_i \times P(c | O, t_i)$$
3. Assign O to the class with maximum probability.

3.1 SETS OF TREES TO AVERAGE OVER

It is impractical to average over every possible tree, so we restrict, S , the set we average over. In the remainder of this section, we describe three sets of tree we consider averaging over: the path set (used by the IND Package, Buntine 1990, 1992), the fanned set (Oliver and Hand 1994a, 1996), and the extended fanned set.

3.1.1 The Path Set

The *path set*, $PSet$, of a complete tree T is dependent on the new object, O , we wish to classify. $PSet = Path_Set(T, O)$ is constructed by:

1. Marking the path down T we associate with O .

2. Pruning each decision node along this path in turn, and adding the newly created tree to $PSet$.

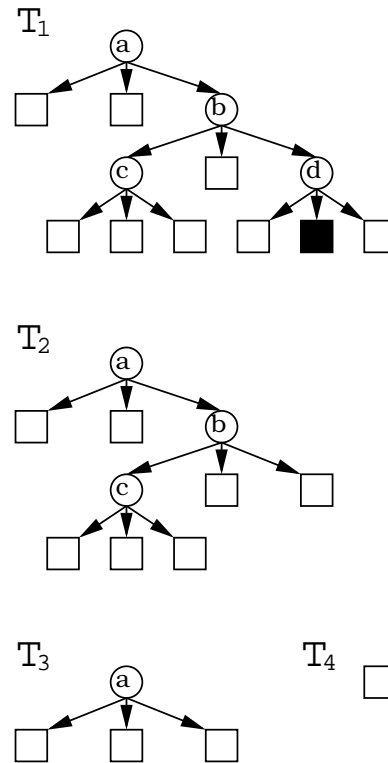


Figure 2: The Path Set for an Example Decision Tree

Figure 2 depicts the path set for tree, T_1 , for an object in the leaf highlighted in the figure:

$$PSet = \{ T_1, T_2, T_3, T_4 \}$$

The path set is a subset of the set of pruned trees (depicted in Figure 1).

3.1.2 The Fanned Set

Another method for generating a set of trees is to use every possible attribute in turn to extend a tree by one level. We define this as the *fanned set* of a tree (Oliver and Hand 1994a, 1996). The fanned set, $FSet$, of a tree T is dependent on the new object, O , we wish to classify. $FSet = Fanned_Set(T, O)$ is constructed by:

1. Finding the leaf, L , we associate with O .
2. Set $FSet$ to the null set.
3. For each attribute A , construct T' by splitting L on A and add T' to $FSet$.

Figure 3 depicts the fanned set for tree T_1 . In this figure, we assume that there are only four attributes available: $\{a, b, c, d\}$. Unlike the path set, the fanned set is not a subset of the set of pruned trees. Given a tree, T , we can classify a new object, O , by using Procedure Average with $S = Fanned_Set(T, O)$.

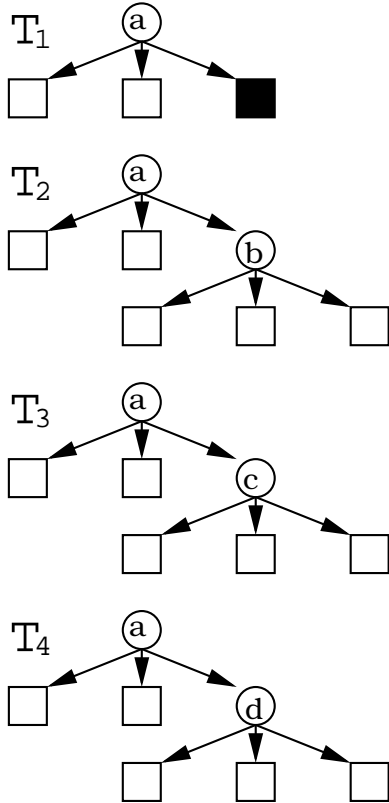


Figure 3: The Fanned Set for an Example Decision Tree

When we consider constructing the fanned set for data sets with continuous attributes, then there are multiple cut-points to consider for each continuous attribute. Typically, a single cut-point is considered by decision tree growing schemes. The cut-point is normally selected using the same criterion that is used

to determine which attribute to split a node on. We restrict ourselves in this paper, to one cut-point for each continuous attribute, by selecting the cut-point which maximises the posterior probability of the tree (see Section 4).

3.1.3 The Extended Fanned Set

We may extend the fanned set to include every tree that is considered during the growing phase (if we use recursive partitioning to grow the tree). We define the *extended fanned set* for an object O using a complete tree, T , as the set formed by the following process:

1. Set $PSet$ to be the path set of T .
2. For each $T_i \in PSet$, let $F_i = Fanned_Set(T_i, O)$.
3. The extended fanned set is the union of the fanned sets constructed in Step 2.

$$ESet = F_1 \cup F_2 \cup \dots \cup F_n$$

Like the fanned set, the extended fanned set is not a subset of the set of pruned trees.

We form a set of weights, W , for each tree in $ESet$. The extended fanned set can then be used to classify a new item O , by using Procedure Average($O, ESet, W$).

4 WEIGHTS FOR DECISION TREES

In this paper, we use a Bayesian approach and set the weights to be proportional to the posterior probability of each tree, $t_i \in S = \{t_1, t_2, \dots, t_n\}$:

$$Prob(t_i | Data) = \frac{Prob(t_i) \times Prob(Data | t_i)}{Prob(Data)}$$

Since $Prob(Data)$ is constant, we set weight w_i to be the normalised value of the product of the prior and the likelihood:

$$w(t_i) = \frac{Prob(t_i) \times Prob(Data | t_i)}{\sum_{t_j \in S} Prob(t_j) \times Prob(Data | t_j)} \quad (1)$$

4.1 THE PRIOR PROBABILITY OF A DECISION TREE

There are two areas where prior distributions over decision trees are discussed; Buntine discusses a range of prior distributions from a Bayesian perspective (Buntine 1990, 1992), and Quinlan and Rivest (1989) and Wallace and Patrick (1993) have proposed codes for decision trees (within a Minimum Encoding Inference framework). We can interpret a code over decision trees as implying a prior distribution over trees by taking:

$$Prob(t_i) = 2^{-Message_Length(t_i)}$$

The prior distributions implied by codes is discussed in Section 5 of Oliver and Hand (1994b).

Consider assigning a prior probability to a tree, t_i , with N nodes, of which L are leaves and I are internal nodes (so $N = L + I$). Buntine's Type II (page 66, Buntine 1992) prior assigns tree t_i the prior probability¹:

$$Prob(t_i) \propto \omega^N$$

If we take $\omega = \frac{1}{2}$, then we have a prior equivalent to the prior implied by Quinlan and Rivest's (1989) code for binary trees. However, the Type II prior is not a universal prior (Rissanen 1983, 1987, Baxter and Oliver 1994) for trees which have attributes with more than one distinct arity, e.g., the Heart Disease domain (Murphy and Aha 1992).

We therefore used the universal prior implied by Wallace and Patrick's (1993) code for decision trees. Let $ap(i)$ be the arity of the parent of node i . Since the root of a tree doesn't have a parent, we set $ap(\text{root}) = \frac{\text{numberofattributes} + 1}{\text{numberofattributes}}$.

$$Prob(t_i) \propto \prod_{i=1}^I \frac{1}{ap(i)} \prod_{l=1}^L \left(1 - \frac{1}{ap(l)}\right) \quad (2)$$

4.2 THE LIKELIHOOD OF A DECISION TREE

Consider a tree, t_i , with L leaves, K classes and leaf l contains M_c training items belonging to class c ($c = 1 \dots K$). We associate a probability with the vector of classes at each leaf² ($P_{VC}(l)$, $l = 1 \dots L$):

$$P_{VC}(l) = \frac{\prod_{c=1}^K M_c!}{(\sum_{c=1}^K M_c)!}$$

We define the likelihood of tree, t_i , to be:

$$Prob(\text{Data} | t_i) = \prod_{l=1}^L P_{VC}(l) \quad (3)$$

4.3 THE POSTERIOR PROBABILITY OF A DECISION TREE

We take the posterior probability of a tree (and hence evaluate the weights for a tree) as being proportional to the product of the prior (Equation (2)) and the likelihood (Equation (3)).

5 COMPLEXITY OF FANNING

The decision tree approach to classification can be divided into three parts: (i) the Growing Phase, (ii) the

¹Since we are normalising the weights in Equation (1), we do not require normalised prior probabilities.

²Other likelihood functions are discussed by Buntine (page 66, 1992).

Pruning (or Averaging) Phase, and (iii) the Classification Phase. In this section, we estimate the time and storage requirements of the traditional decision trees, fanned decision trees, and extended fanned decision trees.

For all three decision tree schemes, the same growing phase is used, and hence the time and storage requirements are of the same order.

5.1 COMPLEXITY OF THE PRUNING PHASE

To estimate the complexity, we define the following symbols: Consider the situation where the growing phase constructed tree T from a data set with D training items, A attributes and C classes. Let H be the height of T , L be the number of leaves in T , I be the number of internal nodes in T , and $N = L + I$ be the total number of nodes in T .

If we prune tree T , then for each split node we must determine whether we should keep it as a split node, or "prune" it back to a leaf:

$$Time(\text{Pruning}) = \sum_{i=1}^I T_{Prune}(Node_i)$$

where $T_{Prune}(Node_i)$ is the time required to calculate the criterion used to determine whether decision node, $Node_i$, should be pruned.

5.2 COMPLEXITY OF THE AVERAGING PHASE

We consider constructing the fanned set for a tree, T , with N nodes, with L leaves, C classes, and $NItems(i)$ training items at node i .

$$Time(\text{Construct Fanned Set})$$

$$\propto \sum_{i=1}^L A \times NItems(Leaf_i)$$

$$Time(\text{Construct Extended Fanned Set})$$

$$\propto \sum_{i=1}^N A \times NItems(Node_i)$$

To use a fanned tree for classification, we require the class distribution (a vector of C integers) for each possible split of each leaf:

$$Additional_Storage(\text{Fanned Tree})$$

$$\propto L \times A \times C$$

$$Additional_Storage(\text{Extended Fanned Tree})$$

$$\propto N \times A \times C$$

Table 1: A Summary of the Data Sets Considered

Data Set	Cont Attr	Binary Attr	Many Valued Attr	Total Attr	Classes	Size of Data Set	Base Acc (%)
Geographic	8	-	-	8	4	106	37.7
Glass	9	-	-	9	6	214	35.5
Heart Disease	5	3	5	13	2	303	54.1
LED	-	7	-	7	10	3000	10.0
Mushroom	-	4	18	22	2	8124	51.8
Pole	4	-	-	4	2	1847	51.0
Votes	-	-	16	16	2	435	61.4

5.3 COMPLEXITY OF THE CLASSIFICATION PHASE

The classification process requires little additional storage.

A new object can be classified by a traditional tree by traversing the path to the appropriate leaf, and examining the class distribution at that leaf:

$$\begin{aligned} & \textit{Time}(\textit{Classify Using Tree}) \\ &= O(H) + O(C) \end{aligned}$$

The time required to classify a new object with a fanned tree is:

$$\begin{aligned} & \textit{Time}(\textit{Classify Using Fanned Tree}) \\ &= O(H) + O(C \times A) \end{aligned}$$

The time required to classify a new object with an extended fanned tree is:

$$\begin{aligned} & \textit{Time}(\textit{Classify Using Extended Fanned Tree}) \\ &= O(H \times C \times A) \end{aligned}$$

6 COMPARISON OF AVERAGING AND PRUNING

We used six data sets from the UCI repository of machine learning databases (Murphy and Aha 1992): Glass, Heart Disease, LED, Mushroom, Pole, and Voting (as modified in Buntine and Niblett 1992) and the Geographic data set (described in Wallace and Patrick 1993). A summary of some properties of these data sets is given in Table 1. The base accuracy (Base Acc) is the proportion of cases that have the most common class.

We implemented the extended fanned tree using the set described in Section 3.1.3 and the weighting scheme described in Section 4. In addition, we implemented the optimal pruning strategy described in Section 2.

The complete tree used for both the extended fanned tree and the optimally pruned tree was generated by

using the splitting rule which maximised the posterior probability at each stage of growing the tree. CART’s stopping rule (not splitting a leaf which has less than 5 items) was used.

Table 2 gives the time taken (in seconds) on a DECstation 5000/240 for C4.5 and the extended fanned tree (EFT) for the Mushroom data set. These results were averaged over 41 runs. The standard deviation of the time taken is indicated after the “±” symbol.

Data Set	Items	C4.5		EFT	
		Time (secs)		Time (secs)	
Mushroom	10	2.4	± 0.5	7.4	± 0.7
	20	2.5	± 0.6	8.2	± 0.8
	30	2.6	± 0.8	8.5	± 0.8
	40	2.6	± 0.9	8.8	± 1.0
	50	3.1	± 1.2	9.2	± 1.1
	100	2.3	± 0.4	10.0	± 0.8
	150	2.1	± 0.2	19.2	± 2.6
	250	2.1	± 0.1	12.9	± 2.7
	500	2.2	± 0.2	10.5	± 0.6
	1000	2.3	± 0.3	10.7	± 0.4
2000	2.7	± 0.2	16.1	± 4.7	

Table 2: Time Taken for C4.5 and EFT

Tables 3 and 4 gives the percentage accuracy for C4.5 (Quinlan 1993), the optimally pruned tree (OPT), and the extended fanned tree (EFT). These results were averaged over 41 runs. For each run a training set was randomly selected, and the success rate was calculated using the remainder of the data set as the test set. The standard deviation of the accuracy is indicated after the “±” symbol. For the C4.5 results, we indicate if the extended fanned tree scheme was significantly better using a one sided matched pair t-test with 40 degrees of freedom. A single star (*) indicates significance at the 0.05 level, a double star (**) at the 0.01 level, and a triple star (***) at the 0.005 level. We indicated with a dagger (†), or a triple dagger (†††) those occasions (in the LED and Pole data sets) when C4.5 significantly outperformed the extended fanned tree scheme at the 0.05 and 0.005 levels respectively.

Table 3: Accuracy Rates for C4.5, OPT, and EFT

Data Set	Items	C4.5		OPT		EFT	
		Accuracy %		Accuracy %		Accuracy %	
Geographic	10	44.5	± 8.4	47.8	± 7.2	45.3	± 8.0
	20	55.8	± 10.2	58.9	± 10.0	57.4	± 11.1
	30	64.8	± 8.0	66.5	± 8.2	64.4	± 8.2
	40	68.0	± 8.4	71.7	± 8.7	69.2	± 8.4
	50	70.7	± 19.4	78.5	± 6.0	75.4	± 8.0
Glass	10	37.8	± 7.7	40.0	± 6.4	39.4	± 6.7
	20	48.3	± 6.9	50.1	± 6.6	49.7	± 7.9
	30	51.0	± 6.4	55.5	± 6.4	52.7	± 7.3
	40	53.8	± 5.6	56.3	± 6.2	55.1	± 7.2
	50	57.3	± 5.6	60.0	± 5.0	57.7	± 5.3
	100	63.4	± 5.2	67.1	± 4.5	65.2	± 4.8
Heart Disease	10	62.5	± 7.8	65.3	± 7.2	65.6	± 8.1
	20	65.0	± 7.2	69.9	± 5.0	68.4	± 5.8
	30	66.7	± 5.6	71.2	± 4.3	70.6	± 4.6
	40	67.6	± 4.9	72.4	± 4.2	71.0	± 4.7
	50	69.1	± 4.5	73.6	± 4.3	71.9	± 4.6
	100	71.7	± 3.4	75.6	± 2.5	73.5	± 2.9
	150	71.5	± 3.6	77.5	± 3.2	74.4	± 3.6
LED	10	26.6	± 4.7	28.3	± 4.3	27.6	± 7.0
	20	40.1	± 6.2	42.1	± 8.3	39.1	± 8.8
	30	48.6	± 7.0	51.8	± 7.2	46.3	± 7.2
	40	55.0	± 7.4	56.5	± 6.4	51.7	± 7.8
	50	58.7	± 6.1	57.4	± 7.4	53.3	± 7.8
	100	65.8	± 2.9	65.5	± 4.4	64.3	± 4.6
	150	67.0	± 2.3	67.8	± 3.4	66.4	± 4.2
	250	68.6	± 1.3	69.9	± 1.7	69.2	± 1.8
	500	70.5	± 1.0	71.9	± 0.8	70.9	± 1.0
	1000	72.2	± 1.0	72.9	± 0.8	72.1	± 0.9

7 DISCUSSION

The extended fanned tree is averaging over a relatively small set of trees (to minimise the computational complexity). We expect the predictive accuracy to be higher if a larger set of trees were used for averaging (such as option trees (Buntine 1992)). Averaging over the extended fanned set of a decision tree has some appealing properties.

- Firstly, we are averaging over the set of trees which a traditional tree growing algorithm inspects during the growing phase. The growing phase requires little additional computational cost.
- Secondly, an extended fanned tree retains much of the comprehensibility of normal trees. Averaging over a larger set of trees (e.g., option trees) may not be comprehensible (page 72, Buntine 1992).
- Thirdly, the additional computation cost (given in Table 2) is reasonable in the light of the expected improvement in predictive performance.

In many cases, the EFT was competitive with an optimally pruned tree (OPT)³. In some cases, (for example, in the Mushroom data set), the EFT actually outperformed the OPT. Two possible reasons for EFTs outperforming OPTs may be (a) the averaging approach may be expressing relationships in the data which cannot be expressed by a single tree; or (b) EFTs are averaging over a larger set of trees of trees than the pruned set of trees.

Within the LED domain, pruning appeared to be a superior strategy to averaging. This could be due to the property that performing prediction using the complete tree is a competitive strategy for this domain (Schaffer 1992). For some sizes of the training set, C4.5 outperformed the optimal pruning strategy. This may be due to C4.5 not using a stopping rule, or to a splitting rule which is more suited to this domain.

³We note that the OPT is not achievable in practice.

Table 4: Accuracy Rates for C4.5, OPT, and EFT

Data Set	Items	C4.5			OPT		EFT	
		Accuracy %			Accuracy %		Accuracy %	
Mushroom	10	76.0	± 10.3	***	80.4	± 11.5	83.9	± 10.6
	20	84.7	± 9.4	***	90.9	± 5.3	93.6	± 5.0
	30	91.3	± 6.0	***	93.6	± 4.8	95.3	± 3.8
	40	93.9	± 4.8	***	95.5	± 3.1	96.1	± 3.5
	50	94.1	± 5.8	***	96.2	± 2.5	96.7	± 2.3
	100	97.5	± 2.0	***	98.3	± 1.2	98.4	± 0.9
	150	98.5	± 0.5	***	98.9	± 0.6	98.8	± 0.6
	250	98.7	± 0.4	***	99.2	± 0.4	99.1	± 0.5
	500	99.0	± 0.6	***	99.6	± 0.3	99.5	± 0.4
	1000	99.6	± 0.4	***	99.8	± 0.2	99.8	± 0.2
2000	99.91	± 0.11	***	99.92	± 0.10	99.95	± 0.08	
Pole	10	74.0	± 13.7		77.2	± 12.4	75.1	± 12.6
	20	79.6	± 4.9		82.2	± 4.2	79.5	± 7.1
	30	79.6	± 4.6	***	83.1	± 2.8	81.4	± 3.7
	40	79.7	± 3.9	***	83.2	± 2.5	81.2	± 3.9
	50	80.5	± 3.8	***	84.2	± 1.7	82.1	± 3.7
	100	82.3	± 2.4		84.2	± 1.6	82.8	± 2.2
	150	82.9	± 2.0		84.8	± 1.1	83.3	± 1.6
	250	83.9	± 1.5		85.2	± 0.9	83.7	± 1.6
	500	85.8	± 1.4	†	86.4	± 1.4	85.2	± 1.5
	1000	87.5	± 1.3	†††	88.6	± 1.5	86.8	± 1.8
Votes1	10	56.2	± 9.8	***	83.1	± 6.9	82.6	± 6.9
	20	83.5	± 5.4	***	84.7	± 4.8	84.6	± 5.2
	30	84.6	± 3.3		85.0	± 3.2	83.1	± 13.7
	40	85.3	± 3.0		86.2	± 2.5	85.4	± 2.4
	50	85.6	± 2.7	*	86.6	± 2.2	86.1	± 2.5
	100	87.0	± 2.3	*	88.0	± 1.8	87.7	± 1.9
	150	87.8	± 1.9	*	88.8	± 1.5	88.4	± 1.6

8 FANNING OVER GRAPHS AND PRODUCTION RULES

It is possible to average over the fanned set for more complicated structures than trees such as production rules (Quinlan 1993) and decision graphs (Oliver 1992). While, it is not obvious how to construct a path set and set weights for these structures (Buntine 1993), it is relatively straightforward to construct a fanned set and assign weights for these structures.

Decision trees, production rules and decision graphs partition the object space. If we partition the object space into components $\{C_1, C_2, \dots, C_n\}$, then we associate a subset of the training set, T_i , with component C_i . Traditionally, we classify new objects which fall in component C_i with the class which has the most elements in T_i . We can do an analogous operation to fanning. For each component, C_i , we construct a set of models, M_i , by splitting T_i on each attribute, and assign a weight to each model in M_i . A new object which falls into component C_i can then be classified by averaging over M_i .

9 CONCLUSION

Over the data sets considered, averaging appeared to be a superior strategy to pruning. The extended fanned trees rarely had lower predictive accuracy than C4.5, and in many cases significantly outperformed C4.5.

Researchers such as Buntine (1990, 1992) and Kwok and Carter (1990) have developed approaches to averaging where fundamental diversity among the trees was sought. This work demonstrated that averaging over a relatively small set of trees can be helpful to predictive performance.

Acknowledgments

This work was supported by Australian Research Council (ARC) Postdoctoral Research Fellowship F39340111. We would like to thank the anonymous referees for their helpful comments. We would also like to thank Chris Wallace, Wray Buntine and Lloyd Allison for valuable discussions.

References

- [1] L.R. Bahl, P.F. Brown, P.V. deSouza, and R.L. Mercer (1989). A tree-based statistical language model for natural language speech recognition. *IEEE Transactions on Acoustics, Speech, and Signal Processing*, 37:1001–1008.
- [2] R.A. Baxter and J.J. Oliver (1994). MDL and MML: Similarities and differences. Technical report TR 207, Department of Computer Science, Monash University, Clayton, Victoria 3168, Australia.
- [3] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone (1984). *Classification and Regression Trees*. Wadsworth, Belmont.
- [4] W.L. Buntine (1990). *A Theory of Learning Classification Rules*. PhD thesis, School of Computing Science in the University of Technology, Sydney.
- [5] W.L. Buntine (1992). Learning classification trees. *Statistics and Computing*, 2:63–73.
- [6] W.L. Buntine (1993). Personal communication.
- [7] W.L. Buntine and T. Niblett (1992). A further comparison of splitting rules for decision-tree induction. *Machine Learning*, 8:75–85.
- [8] L.A. Clark and D. Pregibon (1992). Tree-based models. In J.M. Chambers and T.J. Hastie, editors, *Statistical Models in S*, pages 377–420. Wadsworth and Brooks, California.
- [9] S.B. Gelfand, C.S. Ravishankar, and E.J. Delp (1991). An iterative growing and pruning algorithm for classification tree design. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 13(2):163–174.
- [10] T. Hastie and D. Pregibon (1990). Shrinking trees. Technical report, AT&T Bell Laboratories, Murray Hill, New Jersey 07974, USA.
- [11] I. Kononenko (1992). Combining decisions of multiple rules. In B. du Boulay and V. Sgurev, editors, *Artificial Intelligence V: Methodology, Systems, Applications*, pages 87–96. Elsevier Science, Amsterdam.
- [12] S.W. Kwok and C. Carter (1990). Multiple decision trees. In R.D. Schachter, T.S. Levitt, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*, pages 327–335. Elsevier Science, Amsterdam.
- [13] J. Mingers (1989). An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 3:319–342.
- [14] P.M. Murphy and D.W. Aha (1992). UCI repository of machine learning databases.
- [15] J.J. Oliver, D.L. Dowe, and C.S. Wallace (1992). Inferring decision graphs using the minimum message length principle. In A. Adams and L. Sterling, editors, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 361–367. World Scientific, Singapore.
- [16] J.J. Oliver and D.J. Hand (1994a). Averaging over decision stumps. In *Lecture Notes in Artificial Intelligence 784, Machine Learning: ECML-94*, pages 231–241. Springer-Verlag, Berlin.
- [17] J.J. Oliver and D.J. Hand (1994b). Introduction to minimum encoding inference. Technical report TR 4-94, Dept. of Statistics, Open Uni., Walton Hall, Milton Keynes, MK7 6AA, UK. Also available as TR 205 Dept. Computer Science, Monash Uni., Clayton, Vic 3168, Australia.
- [18] J.J. Oliver and D.J. Hand (1996). Averaging over decision trees. *Journal of Classification*, To appear. An extended version is available as Technical Report TR 5-94, Department of Statistics, Open University, Walton Hall, Milton Keynes, MK7 6AA, UK.
- [19] J.R. Quinlan (1986). Induction of decision trees. *Machine Learning*, 1:81–106.
- [20] J.R. Quinlan (1992). Learning with continuous classes. In A. Adams and L. Sterling, editors, *Proceedings of the 5th Australian Joint Conference on Artificial Intelligence*, pages 343–348. World Scientific, Singapore.
- [21] J.R. Quinlan (1993). *C4.5: Programs for Machine Learning*. Morgan Kaufmann, San Mateo, CA.
- [22] J.R. Quinlan and R.L. Rivest (1989). Inferring decision trees using the minimum description length principle. *Information and Computation*, 80:227–248.
- [23] J. Rissanen (1983). A universal prior for integers and estimation by minimum description length. *Annals of Statistics*, 11:416–431.
- [24] J. Rissanen (1987). Stochastic complexity. *Journal of the Royal Statistical Society (Series B)*, 49:223–239.
- [25] C. Schaffer (1992). Deconstructing the digit recognition problem. In *Machine Learning: Proceedings of the Ninth International Workshop*, pages 394–399.
- [26] C.S. Wallace and J.D. Patrick (1993). Coding decision trees. *Machine Learning*, 11:7–22.