Learning Classification Trees

Wray Buntine
RIACS
AI Research Branch, Mail Stop 244-17
NASA Ames Research Center
Moffett Field, CA 94035

NASA Ames Research Center
Artificial Intelligence Research Branch

Technical Report FIA-90-12-19-01

February, 1991

Learning Classification Trees

Wray Buntine
wray@ptolemy.arc.nasa.gov
RIACS & NASA Ames Research Center*
Mail Stop 244-17, Moffet Field, CA 94035, USA

February 19, 1991

Abstract

Algorithms for learning classification trees have had successes in artificial intelligence and statistics over many years. This paper outlines how a tree learning algorithm can be derived from Bayesian decision theory. This introduces Bayesian techniques for splitting, smoothing, and tree averaging. The splitting rule turns out to be similar to Quinlan's information gain splitting rule, while smoothing and averaging replace pruning. Comparative experiments with reimplementations of a minimum encoding approach, Quinlan's C4 [20] and Breiman *et al.*'s CART [4] show the full Bayesian algorithm is consistently as good, or more accurate than these other approaches though at a computational price.

This paper was presented at the *Third International Workshop on Artificial Intelligence and Statistics*, Fort Lauderdale, Florida, Jan. 3-5, 1991.

*Research Institute for Advanced Computer Science.

Learning Classification Trees

Wray Buntine
wray@ptolemy.arc.nasa.gov
RIACS & NASA Ames Research Center
Mail Stop 244-17, Moffet Field, CA 94035, USA

February 12, 1991

Abstract

Algorithms for learning classification trees have had successes in artificial intelligence and statistics over many years. This paper outlines how a tree learning algorithm can be derived from Bayesian decision theory. This introduces Bayesian techniques for splitting, smoothing, and tree averaging. The splitting rule turns out to be similar to Quinlan's information gain splitting rule, while smoothing and averaging replace pruning. Comparative experiments with reimplementations of a minimum encoding approach, Quinlan's C4 [20] and Breiman *et al.*'s CART [4] show the full Bayesian algorithm is consistently as good, or more accurate than these other approaches though at a computational price.

1 Introduction

A common inference task is where we wish to make a discrete prediction about some object given other details about the object. For instance, in financial credit assessment [7] we wish to decide whether to accept or reject a customer's application for a loan given their personal particulars. This prediction task is a basic function in many expert systems, and is referred to in AI as the classification problem, where the prediction is referred to as the classification. The task is to learn a classifier given a *training sample* of classified examples. In the credit assessment case above, classes correspond to "accept" and "reject". A training sample in this case would be historical records of previous loan applications together with how the loan turned out. This generic learning task is referred to as supervised learning in pattern recognition, and induction or empirical learning in machine learning [18]. Various techniques have been developed in the statistical community such as discriminant analysis and nearest neighbour methods [22].

This prediction problem is often handled with *partitioning classifiers*. These classifiers split the example space into partitions, for instance, ID3 [18] and CART [4] use classification trees to recursively partition the example space and CN2 [11] and PVM [28] use disjunctive rules (disjunctive rules also partition a space, but not recursively in the manner of trees). Tree algorithms build trees such as the ones shown in Figure 1. The one shown on the left has the classes *hypo* (hypothyroid) and *not* (not hypothyroid) at the leaves. This tree is referred to as a *decision tree* because decisions about class membership are represented at the leaf nodes. Notice that the first test, $TSH > 200$ is a test on the real valued attribute TSH. In typical problems involving noise, class probabilities are usually given at the leaf nodes instead of class decisions, forming a *class probability tree* (where each

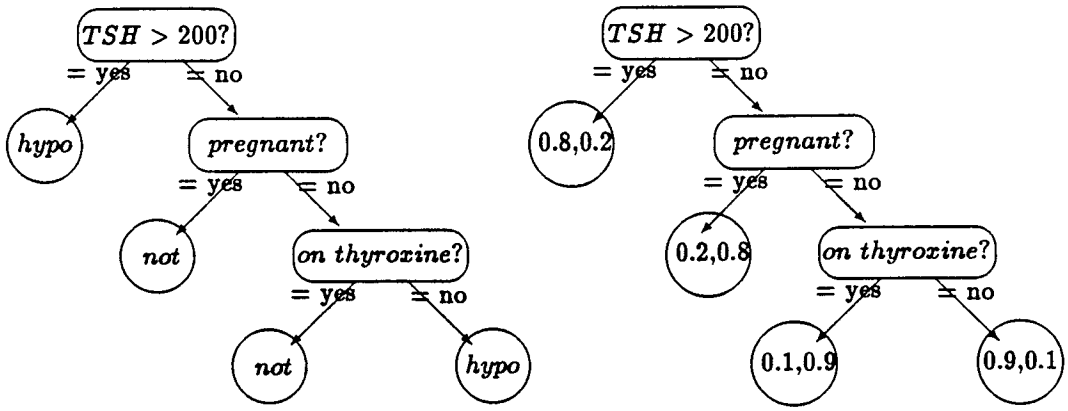

Figure 1: A decision tree and a class probability tree from the thyroid application

leaf node has a vector of class probabilities). A corresponding class probability tree is given in the right of Figure 1. The leaf nodes give predicted probabilities for the two classes. Notice that this tree is a representation for a conditional probability distribution of class given information higher in the tree. We will only be concerned with class probability trees in this paper since decision trees are a special case. The tree-based approaches have been pursued in many areas such as applied statistics, character recognition and information theory for well over two decades. Perhaps the major classical statistics text in this area is [4], and a wide variety of methods and comparative studies exist in other areas [17,16,6,16,2,21,12,10].

The standard approach to building a class probability tree is to use a sequence of stages. A tree is first *grown* to completion so that the tree partitions the training sample into terminal regions of all the one class. This is usually done from the top down using a *recursive partitioning* algorithm. Choose a test for the root node to create a tree of depth one and partition the training set among the new leaves just created. Now apply the same algorithm recursively to each of the leaves. The test is chosen at each stage using a greedy one-ply lookahead heuristic. Experience has shown that a tree so grown will suffer from over-fitting, in the sense that nodes near the bottom of the tree will have just been fitted to noise in the sample, and their removal can often increase predictive accuracy, see [18] for an introduction. To help overcome this over-fitting problem, a second process is subsequently employed to *prune* back the tree, for instance using resampling or hold-out methods [4,12], approximate significance tests [19,15], or minimum encoding [21]. The pruned tree may still have observed class probabilities at the leaves with zeroes for some classes, an unrealistic situation when noisy data is being used. So *smoothing* techniques, eqexplained later, are sometimes employed to make better class probability estimates [2,10]. A final technique is to build multiple trees and use the benefits of *averaging* to arrive at possibly more accurate class probability estimates [13].

This paper outlines a Bayesian approach to the problem of building trees that is related to the minimum encoding techniques of Wallace and Patrick [27] and Rissanen [24]. These two encoding approaches are based on the idea of the "most probable model", or its logarithmic counterpart, the minimum encoding. But the approach here averages over the best few models using two techniques, the simplest is a Bayesian variant of the smoothing technique of Bahl, Brown, De Souza and Mercer [2]. A fuller discussion of the Bayesian methods presented here, including the treatment of real values and extensive comparative experiments, can be found in [6]. A fuller presentation of the

experiments discussed here can be found in [5].

2 Theory

2.1 Basic theory

To reason about posterior probabilities of class probability trees, we need to separate out the discrete and the continuous component of a class probability tree. These need to be modelled by probability functions, and probability density functions respectively. The treatment of real-valued tests has not been given here, see [6] for further details. More advanced techniques such as subsetting have also not been addressed.

A class probability tree partitions the space of examples into disjoint subsets, each leaf corresponding to one such subset, and associates a conditional probability rule with each leaf. Denote the *tree structure* that defines the partition by T; this is determined by the branching structure of the tree together with the tests made at the internal nodes. Suppose there are C classes, $d_1, \ldots, d_C$. The probabilistic rule associated with each leaf can be modelled as a conditional probability distribution. Suppose example x falls to leaf l in the tree structure T, denoted x *falls in leaf* l. Then the tree gives a vector of class probabilities $\phi_{j,l}$ for $j = 1, \ldots, C$ which give the proportion of examples at the leaf that have class d_j. A class probability tree then corresponds to a (discrete) tree structure T, together with the (continuous) matrix of *class proportions* $\Phi_T = \{\phi_{j,l} : j = 1, \ldots, C, l \in leaves(T)\}$. If the choice of a test at a node requires selecting a "cut-point" (a real value), however, as does the test at the root of the tree in Figure 1, then this framework needs to be modified. The "cut-points" chosen are continuous, not discrete parameters. This problem is not adequately handled here, although some rough approximations are discussed in [6].

In this paper, the loss function assumed for prediction is minimum errors, so this corresponds to determining which class has maximum posterior probability of being true. In the Bayesian framework, the prediction task to be solved involves determining posterior probabilities for different tree structures and class proportions, and then returning the posterior class probability vector for a new example based on posterior probability over all possible trees. Since posterior class probabilities are being calculated, loss functions other than minimum errors could be readily handled in the same framework. The mathematics of this process is outlined in the next few sections.

For the learning problem, we are given a training sample consisting of N examples $\vec{x}$ with known classification given by N class values $\vec{c}$. Examples and their classes are assumed to be generated identically and independently. The distribution of a single classified example x, c can be specified by a probability distribution on the example together with a conditional probability distribution on the class c given the example x. Only the conditional probability distribution is important here and it is this we are modelling with class probability trees. Given a sample consisting of N examples $\vec{x}$ with known classification $\vec{c}$, we are interested in determining the *conditional posterior* of a class probability tree given the training sample:

$$\begin{aligned} Pr(T, \Phi_T \mid \vec{x}, \vec{c}) &\propto Pr(T, \Phi_T) \prod_{i=1}^{N} Pr(c_i \mid x_i, T, \Phi_T) , \\ &= Pr(T, \Phi_T) \prod_{l \in leaves(T)} \prod_{j=1}^{C} \phi_{j,l}^{n_{j,l}} , \end{aligned}$$

where $n_{j,l}$ is the number of examples of class d_j falling in the l-th leaf of the tree structure T. Consider a prior[1] consisting of a term for the tree structure, and a term for the class proportions at each leaf, given by

$$Pr(T, \Phi_T) = Pr(T)Pr(\Phi_T|T) = Pr(T) \prod_{l \in leaves(T)} \frac{1}{B_C(\alpha_l, \ldots, \alpha_l)} \prod_{j=1}^{C} \phi_{j,l}^{\alpha_l - 1} . \quad (1)$$

This assumes different class probability vectors at the leaves are *a prior* independent. B_C is the C-dimensional beta function defined by

$$B_C(x_1, \ldots, x_C) = \frac{\prod_{j=1}^{C} \Gamma(x_j)}{\Gamma(\sum_{j=1}^{C} x_j)} ,$$

and Γ is gamma function [1] (for instance, $\Gamma(n) = (n-1)!$). The term indexed by l in Equation (1) is a Dirichlet distribution (a C-dimensional variant of the two-dimensional beta distribution) with all C parameters identically equal to α_l. This form of prior is chosen because it seems reasonably general, and it is a conjugate prior (that is, in the same functional form as the likelihood function [3]) so it simplifies the mathematics to follow. Choices for $Pr(T)$ and the α_l are discussed below. Using standard properties of the Dirichlet distributions [6, Sec.2.5], posteriors conditioned on the training sample can now be computed as follows.

$$\begin{aligned} Pr(T, \Phi_T \mid \vec{x}, \vec{c}) &= Pr(T \mid \vec{x}, \vec{c}) \cdot Pr(\Phi_T \mid \vec{x}, \vec{c}, T) , \\ Pr(T \mid \vec{x}, \vec{c}) &\propto Pr(T) \prod_{l \in leaves(T)} \frac{B_C(n_{1,l} + \alpha_l, \ldots, n_{C,l} + \alpha_l)}{B_C(\alpha_l, \ldots, \alpha_l)} , \quad (2) \\ Pr(\Phi_T \mid \vec{x}, \vec{c}, T) &\propto \prod_{l \in leaves(T)} \frac{1}{B_C(\alpha_l, \ldots, \alpha_l)} \prod_{j=1}^{C} \phi_{j,l}^{n_{j,l} + \alpha_l - 1} . \end{aligned}$$

One important property of these posteriors is that they are multiplicative on the nodes in the tree whenever the prior is also multiplicative on the nodes. This means posteriors for a collection of similar trees can be efficiently added together using an extended distributive law as described below. A second important property is that the discrete space of tree structures is combinatoric, and only partially ordered. This means for smaller data sets we might expect to have very many different looking trees with a similar high posterior. In contrast, consider the space of polynomials of a single variable. This is a linearly ordered discrete space, so we can expect polynomials with a high posterior to be closely related in the space.

Posterior expected values and variances for the proportions Φ_T can be deduced from the posterior using standard properties of the Dirichlet distribution. For instance, the posterior expected[2] class proportions given a particular tree structure are:

$$\mathbf{E}_{\Phi_T | \vec{x}, \vec{c}, T}(\phi_{j,l}) = \frac{n_{j,l} + \alpha_l}{n_{\cdot,l} + C\alpha_l} , \quad (3)$$

where $n_{\cdot,l} = \sum_{j=1}^{C} n_{j,l}$.

[1] In what follows, the term "prior" is an abbreviation for "prior probability distribution".

[2] The notation for the expected value $\mathbf{E}_{x|y}(z(x,y))$ denotes the expected value of $z(x,y)$ according to the distribution for x conditioned on knowing y.

The posterior for the tree structure can be approximated when the sample size at each leaf ($n_{\cdot,l}$) is large using Sterling's approximation [1].

$$-\log Pr(T \mid \vec{x}, \vec{c}) \;\approx\; -\log Pr(T) + N \cdot I(C|T) + constant \;, \tag{4}$$

where the dominant term $I(C \mid T)$ represents the expected information gained about class due to making the tests implicit in the tree structure T. That is:

$$I(C \mid T) \;\equiv\; \sum_{l \in leaves(T)} \frac{n_{\cdot,l}}{N} \, I\left(\frac{n_{1,l}}{n_{\cdot,l}}, \ldots, \frac{n_{C,l}}{n_{\cdot,l}}\right) \;.$$

The function I used here is the standard information or entropy function over discrete probability distributions. The information measure $I(C \mid T)$ when applied to a tree of depth one is used as a splitting rule when growing trees [18,4]. If some leaves have small numbers of examples, then approximation (4) is poor, and the beta function really needs to be used. The beta function then has the effect of discounting leaves with small example numbers.

2.2 Choice of prior

There are two components for the prior: the choice of α_l for each leaf node l, and the choice of the prior on the tree structure T. The actual choice of prior depends on the problem at hand. The choice of a non-informative prior for Dirichlet distributions (i.e. for a multinomial sampling process, what is a non-informative prior on the unknown parameters?) is a controversial topic in Bayesian statistics. Laplace originally suggested $\alpha = 1$, and $\alpha = 0.5$ is also widely used. Both these values do not seem to make sense when there are a large number of classes, because with C large in Formula (3), it takes many examples before the effect of the prior becomes diminished. In general, a smaller α gives more prior preference to extreme distributions. A common compromise is to use $\alpha = \frac{1}{C}$, and this has worked well in experiments.

Wallace [27] has suggested setting

$$\alpha_l \;=\; \gamma \beta^{depth(l)} \;,$$

where γ and β are some other parameters less then 1.0, and $depth(l)$ is the depth of the leaf l. This has the effect of saying nodes should be more accurate as they get lower down the tree. (This is because decreasing α makes extreme probabilities in the multinomial more likely.) While this is certainly true during the growing process, there seems no inherent justification for it of the final tree in general. Why should larger trees be inherently more accurate than smaller trees.

Wallace also suggests attempting to maximise the posterior distribution with respect to these prior meta-parameters. For instance, we make the parameters uniform, $\alpha_l = \alpha$, and then use gradient descent or the Newton-Raphson method to find the value of α maximising the posterior. This is not difficult because derivatives of the beta function are readily determined from the digamma family of functions [1]. One would think that modification of the prior in such a way is sacrilege to a Bayesian. Surprisingly enough this is not the case; the devout Bayesian can take up this meta-prior without guilt. Because we do not know α, our prior should really average over possible values of α as well. But this is not usually computationally practical, for then we lose the benefits of using a conjugate prior. Instead we assume the majority weight of the posterior will be localised in one

particular region of α values, so we only need consider α set near that value. The maximisation process is used to determine where the majority weight of the posterior lies.

For the choice of prior on the tree structure T, there are several choices that seem reasonable. Each of these priors are multiplicative on the nodes in the tree, which means the posterior of Formula (2) is a multiplicative function over the nodes in the tree. Also, the priors presented are not normalised as their later use does not require it.

I: Each tree structure is equally likely, $Pr(T)$ is constant.

II: Give a slight preference to simpler trees, such as $Pr(T) \propto \omega^{|leaves(T)|}$, for $\omega < 1$.

III: The tree shapes (tree structure minus choice of tests at internal nodes) are equally likely.

$$Pr(T) \propto \prod_{n \in nodes(T) - leaves(T)} \frac{1}{\#possible\ tests\ at\ n}.$$

IV: Tree structures are coded using bits for "node", "leaf", and "choice of test". See [24, p.165] or [27] for details.

The priors have been given in increasing strength, in the sense that type IV priors represent an extreme preference for simpler trees, but type I priors represent no such preference. In medical data, for instance, where many of the attributes supplied for each example may well be noise, we would expect the stronger priors to be more reasonable. In problems like the classic faulty LED problem [4], we would expect all faulty LED indicators to be somewhat informative about the actual digit being represented, so larger trees seem reasonable. The minimum encoding approaches [21,24] seem to suggest there is some inherent universality of the tree encoding (or prior) they outline. In the Bayesian framework, the "best" choice of prior over a combinatorial discrete model space like trees is not at all clear, and the notion of a universal prior makes no sense. Experimental results discussed later support the view that when choosing the most probable model from limited data a prior can and should be selected with caution.

2.3 Tree growing

Formula (2) suggests a heuristic for growing a tree in the standard recursive partitioning algorithm described in the introduction. When expanding the (single) current node, for each possible test grow a tree of depth one at that node by extending it one more level. Then choose the new test yielding a tree structure with the maximum posterior probability. Because the posterior is multiplicative, we only need look at that component of the posterior contributed by the new test and its leaves. The one-ply lookahead heuristic for evaluating a test then becomes:

$$Quality(test) = Pr(test) \prod_{l \in outcomes(test)} \frac{B_C(n_{1,l} + \alpha_l, \ldots, n_{C,l} + \alpha_l)}{B_C(\alpha_l, \ldots, \alpha_l)}, \qquad (5)$$

where $Pr(test)$ is the contribution to the tree prior, $outcomes(test)$ is the set of test outcomes, and $n_{j,l}$ is the number of examples in the training sample at the current node with class j and having test outcome l. Computation should be done in log-space to avoid underflow. A test should be chosen to maximise Formula (5).

If the test being evaluated contains a cut-point, as does the test at the root of the tree in Figure 1, then this too should be averaged over. Formula (5) in this case represents an evaluation of the test conditioned on knowing the cut-point. In reality we have to chose this as well, so we should calculate the expected value of Formula (5) across the full range of cut-points [6].

The quality heuristic of Equation (5) can also be used as a stopping rule [8], which is a method for guessing when to stop expanding the current node. If the measure is much less than 1.0, then the expanded subtree is expected to be worse than turning the current node into a leaf. The quality heuristic can also be readily improved using an N-ply lookahead beam search instead of the 1-ply.

The heuristic of Equation (5) is very similar in performance [6, Sec.6.6.1] to the information gain measure of Quinlan [18] and to the GINI measure of CART [4]. It has the distinct advantage, however, of returning a measure of quality that is a probability. This can be used to advantage when either growing trees from a very large training sample, or when growing trees incrementally, and as already explained, can be used as a stopping rule.

When determining which test to make at a node using any one-ply lookahead heuristic, we need to do $O(AN)$ operations, or $O(AN \log N)$ if a sort is involved, where N is the size of the sample at that node, and A is the number of potential tests. To reduce computation for very large N, we could evaluate the tests on a subset of the sample (i.e. reduce N in the computation). Because the measure of quality is in units of probability, one can readily determine if one test is significantly better than another according to the measure simply by taking their ratio. This can be used to determine if evaluation on the current subsample is sufficient, or if we need to view a larger subsample.

A related problem occurs when growing trees incrementally [12]. In this regime, a tree needs to be updated given some additions to the training sample. Crawford shows that if we take the naive approach as done in [25] and attempt to update the current tree so the "best" split according to the updated sample is taken at each node, the algorithm suffers from repeated restructuring. This occurs because the best split at a node vacillates while the sample at the node is still small. To overcome this problem Crawford suggested allowing restructuring only if some other test was currently significantly better than the current test at the node, and suggests using a resampling approach to determine significance of splits. Crawford suggested this relatively expensive approach because there was no distribution theory available for developing an appropriate significance test for the quality of tests. The probabilistic measure of Formula (5) now provides a comparative significance test. As the sample becomes larger, this same comparative test could also be used to show that the higher nodes of a tree (which have the larger sample sizes) need not have their tests checked again, to further reduce computational cost.

2.4 Tree smoothing

The posterior on the tree structure also suggests a pruning method based on the "most probable model" approach [9]. Of all those tree structures obtained by pruning back the completed tree, pick the tree structure with maximum posterior probability. This pruning approach was tried with a range of different priors and the approach was sometimes better in performance than Quinlan's C4 or Breiman et al.'s CART, and sometimes worse. With careful choice of prior, the most probable model approach was usually better, however it was unduly sensitive to the choice of prior. This somewhat disappointing performance suggests a more thorough Bayesian approach is needed.

To implement a full Bayesian approach, we need to introduce averaging over possible models,

as for instance approximated by Kwok and Carter [13]. Intuitively, this involves averaging all those trees that seem a reasonable explanation of the classifications in the training sample, and then predicting the class of a new example based on the weighted predictions of the reasonable trees. This is what is required to estimate the posterior probability conditioned on the training sample that a new example x will have class c, given by the formula:

$$\mathbf{E}_{T,\Phi_T|\vec{x},\vec{c}}(Pr(c \mid x, T, \Phi_T)) = \frac{\sum_{T,\ x\ falls\ to\ leaf\ l\ in\ T} Pr(T \mid \vec{x}, \vec{c}) \frac{n_{j,l}+\alpha_l}{n_{.,l}+C\alpha_l}}{\sum_T Pr(T \mid \vec{x}, \vec{c})}, \tag{6}$$

where the summations are over all possible tree structures. This formula says to compute the posterior expected class probabilities for each tree structure, and take the average of all the resultant class probabilities, weighted by their posteriors on the tree structures. Given the combinatoric number of tree structures, such a calculation is not feasible.

There are two computationally reasonable approximations to this general formula that can be made by restricting the summations to a reduced subset of tree structures. Notice that the denominator of the right hand side of Equation (6) can be simplified to 1.0. However, when approximating the summations with a reduced set of tree structures, this full form is required to normalise the result.

The first approximation I call *Bayesian smoothing*. The standard approach for classifying an example using a class probability tree is to send the example down to a leaf and then return the class probability at the leaf. In the smoothing approach, we also average all the class probability vectors encounted at interior nodes along the way [2, p1005]. Given a particular tree structure T', presumably grown using the algorithm described in Section 2.3, consider the space of tree structures, $pruned(T')$, obtained by pruning the tree structure T' in all possible ways. If we restrict the summation in Equation (6) to this space and the prior on the tree structure is a multiplicative function over nodes in the tree, then the sum can be recursively calculated (using a grand version of the distributive law) and is computable in a number of steps linear in the size of the tree [6, Lemma 6.5.1].

$$\begin{aligned}
&\mathbf{E}_{T,\Phi_T|\vec{x},\vec{c}}(Pr(c \mid x, T, \Phi_T)) \\
&\approx \frac{\sum_{T \in pruned(T'),\ x\ falls\ to\ leaf\ l\ in\ T} Pr(T \mid \vec{x}, \vec{c}) \frac{n_{j,l}+\alpha_l}{n_{.,l}+C\alpha_l}}{\sum_{T \in pruned(T')} Pr(T \mid \vec{x}, \vec{c})}, \\
&= \sum_{n \in traversed(x,T')} Pr(n\ is\ leaf \mid \vec{x}, \vec{c}, pruning\ of\ T') \frac{n_{j,n}+\alpha_n}{n_{.,n}+C\alpha_n},
\end{aligned} \tag{7}$$

where $traversed(x, T')$ is the set of nodes on the path traversed by the example x as it falls to a leaf, and $Pr(n\ is\ leaf \mid \vec{x}, \vec{c}, pruning\ of\ T')$ is the posterior probability that the node n in the tree T' will be pruned back to a leaf. This smoothing process can also be used to prune a tree because in some cases, $Pr(n\ is\ leaf \mid \vec{x}, \vec{c}, pruning\ of\ T)$ for n and all its descendents will be so small that they will have no effect on the sum, so can be pruned.

This smoothing approach sometimes significantly improved class probability estimates for a class probability tree (for instance, as measured using the half-brier score [4]), and sometimes made no significant difference. This happened regardless of the pruning and growing approach used to construct the tree originally. In some cases smoothing an adequate replacement for tree pruning, compared with the standard pruning approaches such as pessimistic pruning or cost complexity

pruning. However, for really noisy data using a weak prior, this form of pruning was not strong enough, whereas, for strongly structured data with a strong prior, the pruning was too severe due to the choice of prior. The smoothing method gives predictions quite sensitive to the prior, although it was generally better or at least as good as finding the most probable model.

2.5 Option trees and averaging

The second approximation described here helps to overcome the problem of the sensitivity of smoothing to the tree structure prior, but does so at a computational cost. This second approach involves searching for and compactly storing the most dominant (i.e. high posterior) tree structures in Equation 6. This approach involves building *option trees* and then doing *Bayesian averaging* on these trees. Option trees are a generalisation of the standard tree where options are included at each point. At each interior node, instead of there being a single test and subtrees for its outcomes, there are several optional tests with their respective subtrees. This is a compact way of representing many different trees that share common prefixes. Classification on the resultant structure is done using Equation 6.

A class probability option tree built from Fisher's iris data is represented in Figure 2. These

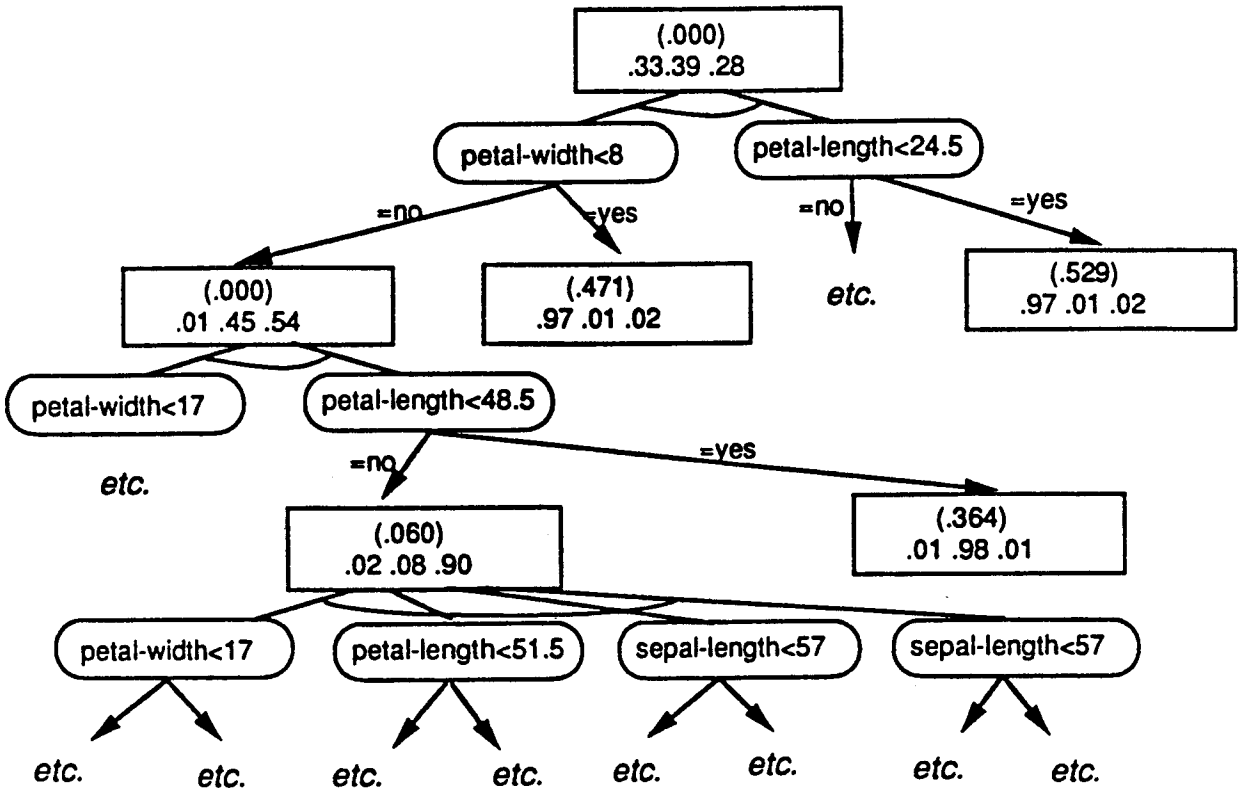

Figure 2: A class probability option tree from the iris application

trees have leaf nodes and test nodes as before, but optional test nodes are sometimes exclusively-or'ed together in a cluster. For instance the very top node of the option tree exclusively-or's together two test nodes, *petal-width* < 8 and *petal-length* < 24.5. These two tests are referred

to as *options* because when selecting a single tree, we are to choose exactly one. Each node is labelled with the probability with which we force that node to be a leaf, which corresponds to the first probability in Equation (7). These determine the probability of selecting any single tree. Each node is also labelled with the estimated class probabilities at the node, used when averaging. These probabilities are given inside a square box for each node. The cluster of two tests at the top is referred to as a node of degree two. So for the top node of degree two, we should treat it as a leaf with probability 0.0 and otherwise choose either the test $petal\text{-}width < 8$ or the test $petal\text{-}length < 24.5$. Both the two options have subtrees with high probability leaves, so these two optional trees are about as good as each other. When parts of the option tree are not included in the figure, the word *etc.* appears. The bottom cluster of options is a node of degree four, which has its subtrees excluded from the figure. The fact that this bottom cluster contains optional tests on all four attributes, which then lead to additional branching tests, indicates that this cluster gives little indication as to which test is more appropriate or if any test is appropriate at all. A good pruning algorithm would perhaps prune the options tree by turning the bottom node of degree four into a leaf, reducing the options tree to depth three.

The smoothing process described in Section 2.4 can be applied to these option trees, but because this no longer involves dealing with a single tree, we refer to the process of growing and smoothing as *tree averaging.* Option trees are grown by expanding the node using those tests that according to the quality measure of Equation (5) have posterior within a factor of the best test. For nodes with large samples, usually only one test is expanded because all others are insignificant according to the quality measure. With smaller samples, or in domains where trees are a poor representation (such as noisy DNF concepts) many tests may be almost as good according to the quality measure so many options will be expanded. This means option trees tend to have more options at the lower nodes.

3 Comparison

3.1 Experiments

The various algorithms discussed, CART, C4, a generic minimum encoding method, and the Bayesian approaches, were applied to 12 different data sets with a range of characteristics. The versions of CART, C4 and minimum encoding were reimplementations by the author that gave comparable performance to the original algorithms on the same data sets. Data sets included Quinlan's hypothyroid and XD6 data [19], the CART digital LED problem [4], three medical domains from Bratko's induction group [8], and a variety of other data sets from the Irvine Machine Learning Database such as "glass", "voting records", "hepatitus" and "mushrooms". More details of the data sets, and acknowledgements to the various sources can be found in [5]. Data sets where divided into training/test pairs, a classifier was built on the training sample and the accuracy, predicted accuracy, and mean square error estimated on the test sample. This was done for 20 random trials of the training/test pair, and significance of difference between two algorithms was checked using the paired t-test.

Accuracy estimates for the different approaches are displayed in Figures 3, 4 and 5. To standardise the experiments, all methods used the tree growing method of Section 2.3. The non-Bayesian methods use different pruning approaches to the grown tree, reimplementations of CART, C4, a generic MML approach (using the type IV prior), and using no pruning at all ("none"). More details

of these methods are given in sections below. Bayesian averaging (growing option trees, followed by smoothing) using both 1 or 2-ply lookahead ("Aver-1P", "Aver-2P") and Bayesian smoothing ("Smooth") on the single grown tree were also tried. Finally, a simple multiple models approach was tried where 5 trees were randomly grown and smoothed, and their class probability vectors averaged using equal weighting ("Mult"). This corresponds to doing a Monte Carlo approximation to the sum in Equation 6 using importance sampling. Trees were grown randomly with each choice of test proportional to the quality heuristic of Formula 5, to try and generate trees in approximate proportion with their posterior.

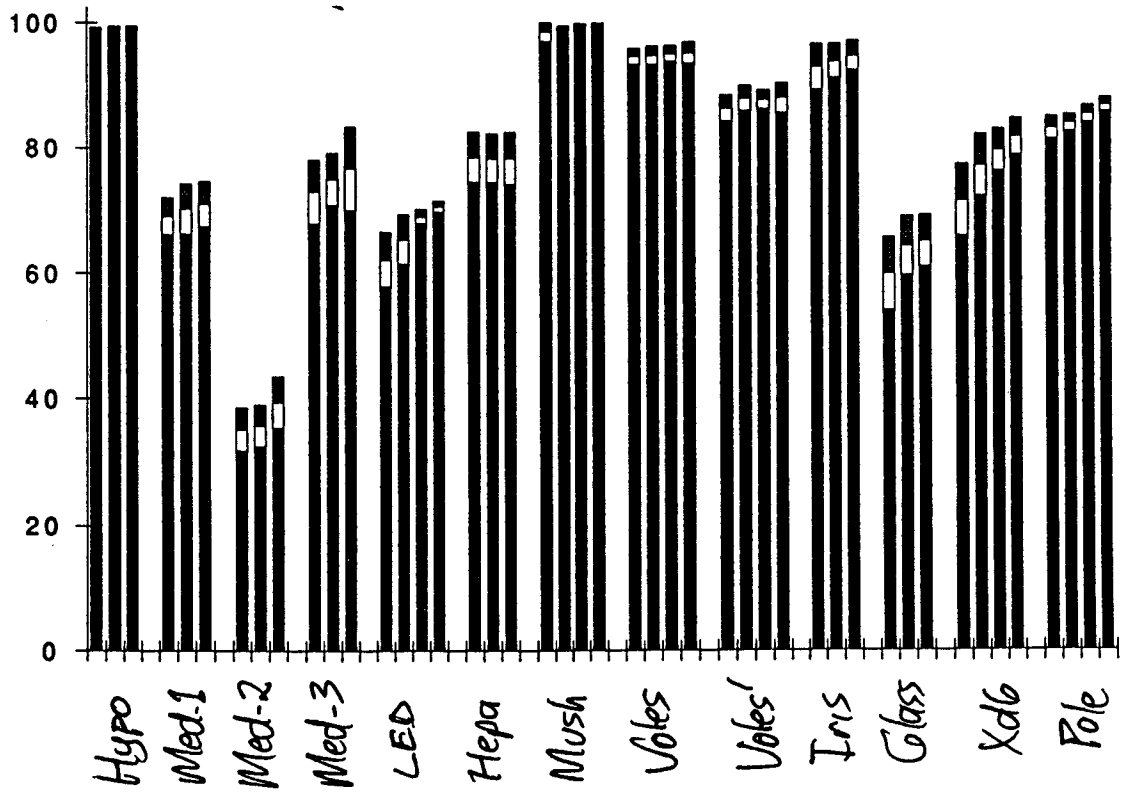

Figure 3: Average accuracy estimates and standard deviations

Each column in the bar chart or line graphs corresponds to one data set at a particular size. Columns are grouped according to the data sets, and then the three or four columns in a group correspond to different sample sizes, ordered in increasing size. The bar chart, Figure 3, gives the estimated accuracy averaged across all methods plus or minus one standard deviation (the average standard deviation for any given method). This average and the standard deviation has been used to normalise the accuracy estimates for the individual methods, as they are displayed in Figures 4 and 5.

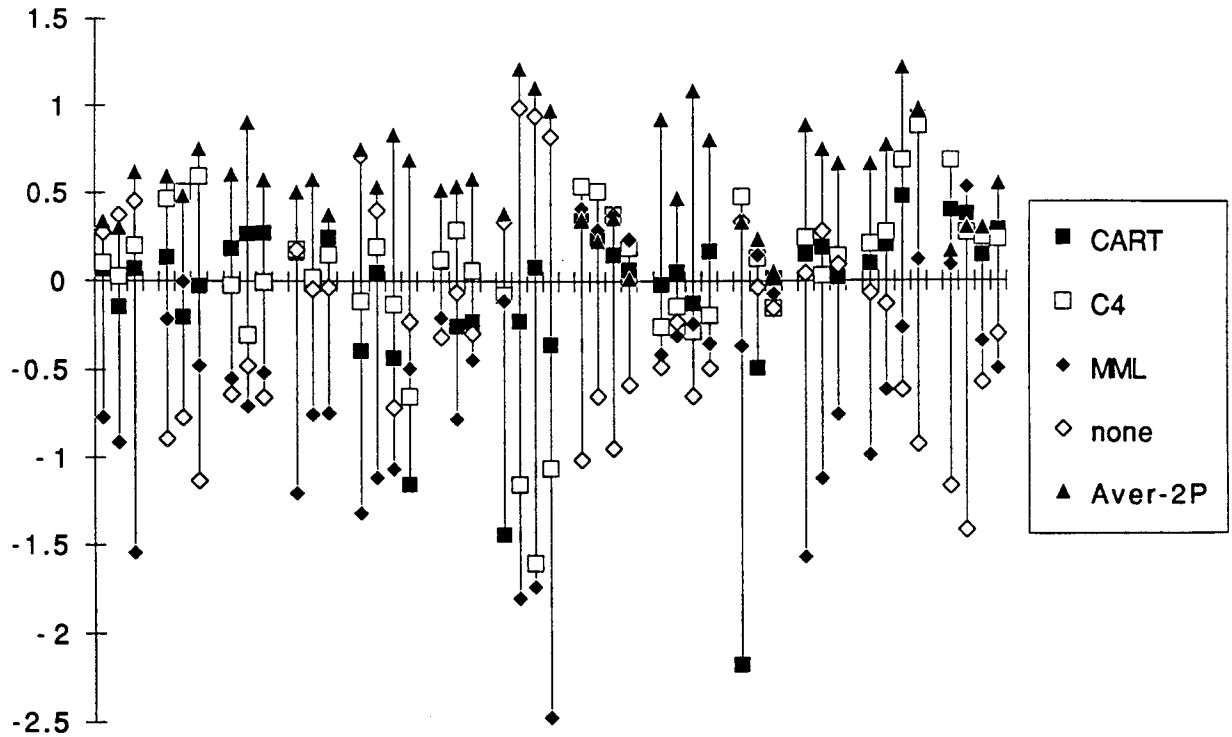

Figure 4: Normalised Accuracies for Methods

Of the algorithms tried, a generic minimum encoding approach, CART, C4, Bayesian smoothing and Bayesian averaging, the averaging approach using a weak prior (type I prior) on tree structures was the only approach that consistently produced the best predictions. In several cases it was pairwise significantly better than all other non-Bayesian approaches at the 5% level. Bayesian averaging with a two-ply lookahead during growing yields improvement in predictive accuracy averaged over 20 trials as often as high as 2%, sometimes more. With a one-ply lookahead, the improvement is not as dramatic but still significant. One has to be cautious in interpreting this result, however, because option trees are more than just a single decision tree, they effectively involve an extension of the model space. The growing of option trees sometimes involved an extra order of magnitude in time and space. Certainly the Bayesian approaches are competitive, and they appear to be superior for smaller samples.

For many of the data sets, it was appropriate to select a prior that had stronger preference towards smaller trees. When this was done, Bayesian smoothing of a single tree gave good predictions, often as good as the Bayesian averaging with one-ply lookahead. This is useful because we would not always wish to go to the computational expense of Bayesian averaging, or we may require just a single tree for explanatory purposes.

A second point of comparison of the algorithms is the parameters available when driving the algorithms. CART and C4 have default settings for their parameters. With CART, heavy pruning

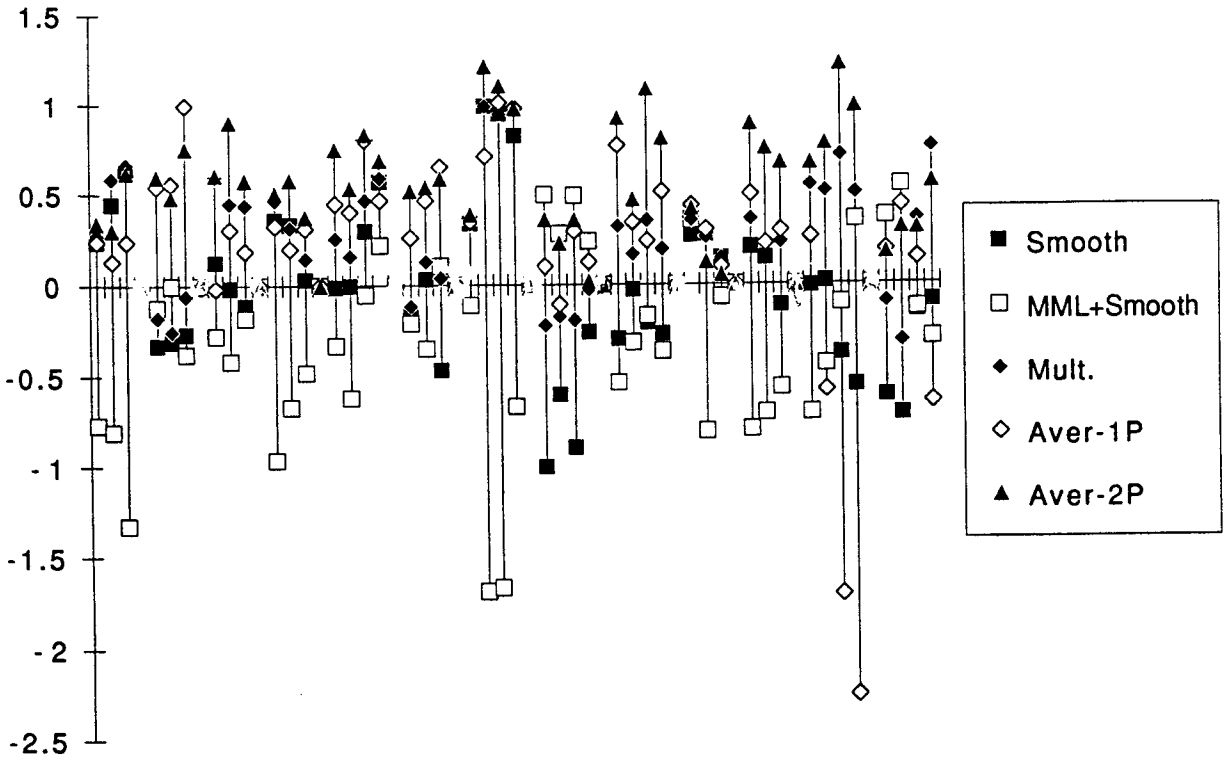

Figure 5: Normalised Accuracies for Bayesian Variations

can be achieved using the 1-SE rule rather than the 0-SE rule. The number of partitions to use in cross-validation cost complexity pruning can also be changed, but the effect of this is unclear, especially since leaving-one-out cross-validation cost complexity pruning gives poor predictive accuracy. The minimum encoding approaches are (according to the purist) free of parameters. However, these approaches often strongly overprune, so Quinlan and Rivest introduce a parameter that allows lighter pruning. So all approaches Bayesian and non-Bayesian have parameters that allow more or less pruning, that can be chosen depending on the amount of structure believed to exist in the data. In the fuller Bayesian approach with option trees and Bayesian averaging, choices available also allow greater search during growing and fuller elaboration of the available optional trees. These parameters have the useful property that predictive accuracy (or some other utility measure) and computational expense are on average monotonic in the value of the parameter. The parameter setting allows improved predictive accuracy at computational expense.

The experiments are described in more detail in [5], and earlier related experiments are reported in [6].

3.2 Minimum encoding approaches

The negative logarithm of the posterior for the tree structure given in Equation (2), given a suitable prior, can be interpreted as the code length for the tree and the classifications combined, and so is

related to the encoding measures of Wallace and Freeman [26], and Rissanen [23]. Wallace et al. derive their Minimum Message Length measure as an approximation to the area under the posterior in the neighbourhood of the proposed model. However, Wallace and Patrick use the exact formula in [27] which is equivalent to the one presented here apart from the choice of prior. Rissanen uses a cruder approximation of this posterior with his MDL measure, although in [24] he extends Quinlan and Rivest's encoding [21] to a more accurate encoding virtually equivalent to the one presented here (using a type IV tree prior and $\alpha_l = 0.5$).

These encoding approaches correspond to the "most probable model" approach. Each represent strong preference for simpler trees because they implicitly use the type IV tree structure prior, which may or may not be appropriate. Experiments indicate the approach often underprunes, sometimes severely (see also [21]). This illustrates how the minimum encoding approaches are unduly sensitive to their implicit choice of prior. Quinlan and Rivest, for instance, suggested weakening the preference for simpler trees (in a manner compromising the correctness of their coding scheme) to obtain somewhat larger trees but more accurate predictions in some cases.

The experiments show that by using a weaker prior but also doing Bayesian averaging on trees as suggested here, the sensitivity to the prior was considerably decreased. Of course, this is to be expected from the Bayesian decision theory, and supports Wallace et al.'s view [26] of minimum encoding as a first approximation to Bayesian theory.

3.3 CART and C4

Breiman, Friedland, Olshen and Stone's approach to building class probability trees [4] is best distinguished from the others in that it uses cost complexity pruning using either cross-validation or hold-out methods to estimate the right cost complexity tradeoff. Whereas Quinlan's earlier version of C4 (reimplemented for these experiments) uses pessimistic pruning [19]. The versions of the algorithms used were reimplementations of the published descriptions. Comparisons with the original algorithms had been made earlier on the full suite of data sets, and the reimplementations are, on the whole, in fair agreement with the original.

In experiments, cost complexity pruning was done with 10-fold cross validation using the 0-SE rule. Breiman et al. suggest this gives the most accurate predictions, and this was confirmed experimentally. There was no consistent decline in performance by changing to the 1-SE rule, or by using 5-fold cross-validation (although leaving-one-out cross-validation performed poorly). Rather, changing the parameters of the pruning method meant changes, sometimes for the worse, sometimes for the better.

In experiments on problems with less structure and more noise, CART was the most consistently accurate of the compared methods other than Bayesian averaging, and C4 was the most consistently accurate in problems with more structure and less noise. In general CART tended to overprune (even though the 0-SE rule was used) and C4 tended to underprune. I tried fiddling the various parameters in these algorithms to adjust for this behaviour but was unsuccessful.

As training samples became large, the CART or C4 approaches became comparable with Bayesian averaging. Bayesian methods are of course particularly suited to learning from smaller samples, where one has to take care with the choice of prior, and to average over the many possible tree structures that seem reasonable given the training sample. With larger samples, all this extra work may be unnecessary. Also, because CART seems to have an implicit preference for smaller trees (it usually overprunes on smaller samples), it can be appropriate in training samples where

there is expected to be less structure and more noise.

4 Conclusion

Bayesian algorithms for learning class probability trees were presented and compared empirically with reimplementations of existing approaches like CART [4], C4 [19] and minimum encoding approaches. The Bayesian averaging algorithm gave significantly better accuracy on predictions for a set of learning problems, but this was at the expense of computational cost.

First, this work has a number of implications for Bayesian learning. Simple maximum posterior methods and minimum encoding methods (which here would choose the single maximum posterior tree) may not perform well in combinatorial discrete spaces if the prior is not well matched to the problem. Considerable improvement can be got by averaging over multiple high posterior models. Importance sampling, and collecting together many high posterior models helps in this task. This has implications to areas like multivariate regression and model discovery (for instance, of Bayesian networks [14]).

Second, Bayesian methods corresponded to a variety of subtasks usually done by existing tree learning approaches. This suggested several ways of improving the growing heuristic for trees and for doing learning in an incremental rather than a batch mode.

Third, given that many non-Bayesian techniques work as well Bayesian when sample sizes become larger, it is important to determine when a sample size is "large enough" so simpler techniques such as minimising error [28] can be applied.

Acknowledgements

Thanks to Peter Cheeseman, Stuart Crawford and Robin Hanson for their assistance, and to the Turing Institute and Brian Ripley at the University of Strathclyde who sponsored me while much of this research was taking shape. I am particularly indebted to Ross Quinlan, whose grasp of the tree field and insistence on experimental evaluation helped me enormously during the thesis development.

References

[1] M. Abramowitz and I.A. Stegun. *Handbook of Mathematical Functions.* Dover Publications, New York, 1972.

[2] L.R. Bahl, P.F. Brown, P.V. de Souza, and R.L. Mercer. A tree-based langauge model for natural language speech recognition. *IEEE Trans. on AS and SP*, 37(7):1001–1008, 1989.

[3] J. O. Berger. *Statistical Decision Theory and Bayesian Analysis.* Springer-Verlag, New York, 1985.

[4] L. Breiman, J.H. Friedman, R.A. Olshen, and C.J. Stone. *Classification and Regression Trees.* Wadsworth, Belmont, 1984.

[5] W.L. Buntine. *Some Experiments with Learning Classification Trees.* Technical Report, NASA Ames Research Center, 1991. In preparation.

[6] W.L. Buntine. *A Theory of Learning Classification Rules.* PhD thesis, University of Technology, Sydney, 1990. Forthcoming.

[7] C. Carter and J. Catlett. Assessing credit card applications using machine learning. *IEEE Expert*, 2(3):71–79, 1987.

[8] B. Cestnik, I. Kononenko, and I. Bratko. Assistant86: a knowledge-elicitation tool for sophisticated users. In I. Bratko and N. Lavrač, editors, *Progress in Machine Learning: Proceedings of EWSL-87*, pages 31–45, Sigma Press, Bled, Yugoslavia, 1987.

[9] P.C. Cheeseman. On finding the most probable model. In J. Shrager and P. Langley, editors, *Computational Models of Discovery and Theory Formation*, pages 73–95, Morgan Kaufmann, 1990.

[10] P.A. Chou. Optimal partitioning for classification and regression trees. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 1991.

[11] P. Clark and T. Niblett. The CN2 induction algorithm. *Machine Learning*, 3(4):261–283, 1989.

[12] S.L. Crawford. Extensions to the CART algorithm. *International Journal of Man-Machine Studies*, 31(2):197–217, 1989.

[13] S.K. Kwok and C. Carter. Multiple decision trees. In R.D. Schacter, T.D. Levitt, L.N. Kanal, and J.F. Lemmer, editors, *Uncertainty in Artificial Intelligence 4*, North-Holland, 1990.

[14] S.L. Lauritzen and D.J. Spiegelhalter. Local computations with probabilities on graphical structures and their application to expert systems. *J. Roy. Statist. Soc. B*, 50(2):240–265, 1988.

[15] J. Mingers. An empirical comparison of pruning methods for decision-tree induction. *Machine Learning*, 4(2):227–243, 1989.

[16] J. Mingers. An empirical comparison of selection measures for decision-tree induction. *Machine Learning*, 3(4):319–342, 1989.

[17] J.R. Quinlan. Generating production rules from decision trees. In *International Joint Conference on Artificial Intelligence*, pages 304–307, Milan, 1987.

[18] J.R. Quinlan. Induction of decision trees. *Machine Learning*, 1(1):81–106, 1986.

[19] J.R. Quinlan. Simplifying decision trees. In B. Gaines and J. Boose, editors, *Knowledge Acquisition for Knowledge-Based Systems*, pages 239–252, Academic Press, London, 1988.

[20] J.R. Quinlan, P.J. Compton, K.A. Horn, and L. Lazarus. Inductive knowledge acquisition: a case study. In J.R. Quinlan, editor, *Applications of Expert Systems*, Addison Wesley, London, 1987.

[21] J.R. Quinlan and R.L. Rivest. Inferring decision trees using the minimum description length principle. *Information and Computation*, 80:227–248, 1989.

[22] B.D. Ripley. An introduction to statistical pattern recognition. In *Interactions in Artificial Intelligence and Statistical Methods*, pages 176–187, Unicom, Gower Technical Press, Aldershot, UK, 1987.

[23] J. Rissanen. Stochastic complexity. *J. Roy. Statist. Soc. B*, 49(3):223–239, 1987.

[24] J. Rissanen. *Stochastic Complexity in Statistical Enquiry.* World Scientific, 1989. Section 7.2.

[25] P. Utgoff. Incremental induction of decision trees. *Machine Learning*, 4(2):161–186, 1989.

[26] C.S. Wallace and P.R. Freeman. Estimation and inference by compact encoding. *J. Roy. Statist. Soc. B*, 49(3):240–265, 1987.

[27] C.S. Wallace and J.D. Patrick. Coding decision trees. 1989. Submitted.

[28] S.M. Weiss, R.S. Galen, and P.V. Tadepalli. Optimising the predictive value of diagnostic decision rules. In *Sixth National Conference on Artificial Intelligence*, pages 521–526, Seattle, 1987.