Classifiers: A Theoretical and Empirical Study*

Wray Buntine
RIACS†
Artificial Intelligence Research Branch
NASA Ames Research Center, Mail Stop 244-17
Moffet Field, CA 94035, USA
Phone: +1 (415) 604-3389
wray@ptolemy.arc.nasa.gov

Abstract

This paper describes how a competitive tree learning algorithm can be derived from first principles. The algorithm approximates the Bayesian decision theoretic solution to the learning task. Comparative experiments with the algorithm and the several mature AI and statistical families of tree learning algorithms currently in use show the derived Bayesian algorithm is consistently as good or better, although sometimes at computational cost. Using the same strategy, we can design algorithms for many other supervised and model learning tasks given just a probabilistic representation for the kind of knowledge to be learned. As an illustration, a second learning algorithm is derived for learning Bayesian networks from data. Implications to incremental learning and the use of multiple models are also discussed.

1 Introduction

Systems for learning classification trees [Quinlan, 1986; Cestnik *et al.*, 1987] are common in machine learning, statistics and pattern recognition. Despite these successes, the study presented here is not motivated by a view that tree classifiers are inherently superior to other learning systems (see for instance, the argument against so-called universal learning algorithms in [Buntine, 1990b]). Rather, this study is motivated by the view that tree learning is an ideal benchmark problem for studying learning theories. The problem is used here to gain insight into generic problems in empirical learning, and to explore a general strategy for designing learning algorithms.

First, current tree learning methods are mature in that many modifications have been suggested in recent years, but little real gain has been made[1]. There are now many different tree learning methods and none clearly superior [Mingers, 1989]. The systems have stood the test of time and are now widely used for benchmarking and comparative studies. If an algorithm design strategy yields a superior tree learning algorithm despite this tough competition, it is likely the strategy will be successful on other learning tasks.

Second, tree learning is a common meeting ground for the different areas of inquiry outside of AI that have learning theories of some form or another such as classical statistics [Breiman *et al.*, 1984] and minimum encoding approaches [Rissanen, 1989]. This provides a rare opportunity to contrast these different learning theories theoretically as well as empirically. Comparative studies conducted to date in AI have provided little understanding about generic principles of algorithm design, partly because comparisons have been made of algorithms that use different knowledge representations, and comparisons have largely ignored the theoretical principles on which the algorithms were based.

Finally, classification trees have been the framework in which a number of discoveries have been made in machine learning: the overfitting problem (related to the accuracy-complexity tradeoff and Ockham's razor), incremental algorithms [Schlimmer and Granger Jr., 1986], and interactive induction [Shapiro, 1987]. The learning problem tree systems tackle is difficult enough to be considered "unlearnable," yet simple enough so not requiring the specialized machinery to learn relations or do constructive induction. In this spirit, this study presents a number of additional discoveries about problems generic to empirical learning.

This paper reviews the Bayesian development of a system for learning tree classifiers, from the theoretical basis, through the hacking required to make the system work, to comparative results with other systems. The development is presented as a generic algorithm design strategy in the second section, and the results discussed in the third section. More detail of the statistical aspects of the system can be found in [Buntine, 1990a; Buntine, 1990c]. Because of the large number of different trials and data sets that were used, full detail of the results are reported elsewhere [Buntine, 1990c].

This paper also discusses some insights that have come out of this study: an emerging trend in machine learning towards the use of multiple models [Kwok and Carter, 1990; Gams, 1989; Jacobs *et al.*, 1991], a generic method

*From *Int. Joint Conf. on AI*, Sydney, 1991.

†Research Institute for Advanced Computer Science

[1] Except, perhaps, in the area of learning strictly logical rules where recent techniques learn new terms; but the focus of this paper is primarily learning noisy or uncertain concepts, where gains have been more incremental.

to overcome the problem of repeated restructuring in incremental learning [Crawford, 1989], and a better understanding of learning theories and their role in helping us design learning algorithms. These are discussed in the fourth section.

We can now design algorithms for many other supervised and model learning tasks given just a probabilistic representation for the kind of knowledge to be learned. As an illustration, a method is outlined in the fifth section for learning Bayesian networks [Lauritzen and Spiegelhalter, 1988], a common representation in medical expert systems. This is a model learning task. The design strategy can also be applied to other learning tasks such as probabilistic rule systems, n-gram models (such as bigram and trigram models) used in speech recognition and natural language, and the function-finding task that is the basis of scientific discovery algorithms. The strategy has also been applied to analyse the training of feed-forward neural networks [Buntine and Weigend, 1991], where popular heuristic procedures for cost functions and network pruning were found to conform well to corresponding methods developed from Bayesian first principles.

2 The Algorithm Design Strategy

The algorithm design strategy presented here is based on approximating Bayesian decision theory. The justification for Bayesian decision theory comes from fundamental principles of how uncertain reasoning should be done [Berger, 1985]. The theory applies widely in inference and plausible reasoning and its use is continually expanding in AI. But there is not a single "Bayesian learning algorithm," as some people mistakenly believe when they learn about Bayesian classifiers [Tou and Gonzalez, 1974]. Rather, Bayesian decision theory presents computational guidelines on how learning should be done for many different learning problems, including for instance, improving an approximate theory using data.

The basic tenet of Bayesian decision theory is that if we do not know something with reasonable certainty, then we should look at some reasonable and mutually exclusive alternatives and weigh them up, to help us make a "representative" decision. A reasonable alternative is one we currently have high subjective belief in. I will first explain how this applies to trees [Buntine, 1990a], to introduce the notation. This is done for the two-class problem with discrete tests at nodes, but easily extends to the multi-class problem, and adjustments for real-valued tests at nodes exist [Buntine, 1990c, Sec.6.5.5]. The formulation is sufficiently general so that it could just as well be applied to other models such as probabilistic rules, Bayesian networks, or one of many other knowledge representations that have a probabilistic interpretation.

Class probability trees have a vector of class probabilities at their leaves [Breiman *et al.*, 1984], and they represent a conditional probability distribution of class value conditioned on example value. A particular class probability tree can be represented by its discrete component T, the *tree structure* given by the shape of the tree and the tests at the leaves, and its continuous component θ, the leaf class probabilities. This gives the conditional probability distribution $Pr(c' \,|\, x', T, \theta)$, which is the *likelihood function* for the class probability tree specified by T and θ for the training example (c', x').

Suppose we are given a training sample of examples $\vec{x}$ and their classes $\vec{c}$, together with a new example x' whose class we wish to predict. If the goal is to minimize errors in prediction (other utility functions can be handled similarly), decision theory says we should choose the class c' to maximize the posterior class probability $Pr(c' \mid x', \vec{x}, \vec{c})$. Using the tree model, this is the posterior average of the class probabilities predicted for c' from all possible class probability trees:

$$\begin{aligned} & Pr(c' \,|\, x', \vec{x}, \vec{c}) \\ &= \sum_T \int_\theta Pr(c' \,|\, x', T, \theta)\, Pr(T, \theta \,|\, \vec{x}, \vec{c})\; d\theta \\ &= \sum_T Pr(T \,|\, \vec{x}, \vec{c})\;\; \mathbf{E}_{\theta | T, \vec{x}, \vec{c}}(Pr(c' \,|\, x', T, \theta)) \quad (1) \end{aligned}$$

where the summations are over the space of all possible tree structures T, and

$$Pr(T \,|\, \vec{x}, \vec{c}) \;\; \propto \;\; \int_\theta Pr(\vec{c} \,|\, \vec{x}, T, \theta)\, Pr(T, \theta)\; d\theta \;,$$

$$\begin{aligned} & \mathbf{E}_{\theta | T, \vec{x}, \vec{c}}(Pr(c' \,|\, x', T, \theta)) \\ &= \int_\theta Pr(c' \,|\, x', T, \theta)\, Pr(\theta \,|\, T, \vec{x}, \vec{c})\; d\theta \;. \end{aligned}$$

Formula (1) simply says to average the class predictions made for each tree structure, where $Pr(T \mid \vec{x}, \vec{c})$, the *posterior* probability of the tree structure T, is the weight used in the averaging process. In this formula, $Pr(T, \theta)$ is the *prior* on the space of class probability trees, and $Pr(\vec{c} \,|\, \vec{x}, T, \theta)$ the likelihood of the training sample.

The algorithm design strategy is based on designing a heuristic procedure to find a single structure or set of structures that can be used to approximate Formula (1). This is described by the following 6 steps, and Table 1 gives the results of the analysis.

1. Precisely define the form of knowledge to be learned by representing it in terms of a parameterized *likelihood function*. A particular parameterization is referred to as a *model* which has a structural (discrete) and a continuous component as described for trees.

2. Develop a prior over the structural and continuous components of the model. The form of the prior should be flexible enough so that it can be changed from application to application. In [Buntine, 1990a], a range of priors are presented for trees. One is given in the table. The prior on the tree structures, $Pr(T)$, is not given but could, for instance, be assumed uniform. The prior on the continuous component is a product of symmetric beta distributions over the probabilities θ. The function $B(n, m)$ given there is the beta function found in many mathematical handbooks. $B(n, m) \approx 1/(n+m) \cdot 1/C_n^{n+m}$.

3. Given a training sample *Sample*, determine a suitably efficient way of computing or approximating the posterior of the structural component of the model. See

structure	T gives the tree structure, specifying the form of the tree, together with tests made at internal nodes
likelihood	conditional likelihood of class c given example x is $Pr(c \mid x, T, \theta) \;=\; \theta_{c\mid l}$ where x belongs in leaf $l \in leaves(T)$
prior	$Pr(T, \theta) \;=\; Pr(T) \prod_{l \in leaves(T)} \frac{\theta^{\alpha}_{1\mid l} \theta^{\alpha}_{2\mid l}}{B(\alpha,\alpha)}$ for $\alpha = 0.5$
sufficient statistics	$n_{i\mid l}$ = number of examples in $Sample$ which occur at leaf l and have class i
posterior	$Pr(T \mid Sample) \;\propto\; Pr(T) \prod_{l \in leaves(T)} \frac{B(n_{1\mid l}+\alpha, n_{2\mid l}+\alpha)}{B(\alpha,\alpha)}$
lookahead heuristic	we replace the leaf l in tree structure T by a test and leaves $l_1, \ldots, l_o$ at each test outcome, to give tree structure $T+$ $\frac{Pr(T+\mid Sample)}{Pr(T \mid Sample)}$ = increase in posterior by changing T to $T+$ $= \frac{Pr(T+)}{Pr(T)} \frac{\prod_{i=1}^{o} B(n_{1\mid l_i}+\alpha, n_{2\mid l_i}+\alpha)}{B(n_{1\mid l}+\alpha, n_{2\mid l}+\alpha)\; B(\alpha,\alpha)^{o-1}}$
estimates	$\mathbf{E}_{\theta \mid T, Sample}\left(\theta_{i\mid l}\right) \;=\; \frac{n_{i\mid l}+\alpha}{n_{\cdot\mid l}+2\alpha}$
smoothing	because the posterior is a product over the nodes in the tree structure, all conditional probability distributions along a branch can be averaged together using a linear-time recursive algorithm

Table 1: Bayesian analysis of learning class probability trees

[Buntine and Weigend, 1991] for an approximation in the more complex domain of feed-forward neural networks.

4. Devise a heuristic search procedure for searching the space of structures to find structures with high posterior. A simple one-ply lookahead procedure can be tried, which corresponds to the standard tree growing algorithm [Quinlan, 1986], although two-ply or three-ply versions could also be tried. Start with the trivial structure, the empty tree. Then consider extending the structure by a single ply, which for trees means growing a single node by adding a test with leaves at the outcomes of the test. A heuristic measure to evaluate the quality of the new growth is given in the table. To prevent overflow/underflow, this measure has to be calculated in log-space. The measure behaves similarly to Quinlan's information gain heuristic, but has some correction terms for multivalued attributes and small samples. This heuristic can also be used as a stopping rule [Cestnik *et al.*, 1987].

5. Given a training sample *Sample* and a structure T, determine a formula or approximation for the posterior expected values of the parameters θ, as required for Formula (1).

6. Devise a procedure for approximating the summation of Formula (1) by a small set of high posterior structures. Several suggestions are given below. This is currently an active area of research.

Minimum encoding approaches [Rissanen, 1989; Wallace and Freeman, 1987] to supervised learning and the so-called "most probable model" (Bayesian) approach are first-order approximations to Formula (1), because they attempt to find a single high posterior structure. Step 6 in the design strategy above is the main improvement suggested here, because it suggests how to improve on this.

There are three techniques for performing Step 6. These correspond to different ways of estimating the sum in Formula (1):

Smoothing: The sum can be computed in closed form if it is restricted to the set of tree structures obtained by pruning a large tree structure in all possible ways. A linear time algorithm is given in [Buntine, 1990c, Lemma 6.5.1]. This is called smoothing because it is equivalent to smoothing out the class probabilities at the leaf of a tree by averaging them with some class probabilities from interior nodes of the tree.

Averaging: The sum can be approximated by searching for and storing many dominant terms, i.e. many high posterior trees structures. We can build multiple tree structures, and combine them together efficiently in an AND-OR representation called *option trees*. Growing option trees and then applying a similar summation process to smoothing is called *tree averaging*.

Multiple Models: The sum can be approximated by using importance sampling and Monte Carlo estimation. That is, a few tree structures are generated in approximate proportion with their posterior (this is done using the tree growing heuristic [Buntine, 1990a]), and their class probability vectors uniformly averaged.

3 Experimental Results

Reimplementations of CART [Breiman *et al.*, 1984], C4 [Quinlan, 1988], and a generic minimum encoding ap-

proach were compared with the Bayesian approaches[2]. The algorithms were applied to 12 different data sets with a range of characteristics. These included Quinlan's hypothyroid and XD6 data [Quinlan, 1988], the CART digital LED problem [Breiman *et al.*, 1984], three medical domains made available by Bratko's induction group [Cestnik *et al.*, 1987], and a variety of other data sets from the Irvine Machine Learning Database such as "glass," "voting records," "hepatitis," and "mushrooms." Data sets were 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 for 4 different training set sizes, and significance of difference between two algorithms was checked using the paired t-test.

Of the algorithms tried, a generic minimum encoding approach, (re-)CART, (re-)C4, Bayesian smoothing and Bayesian averaging, the averaging approach using a uniform prior on tree structures was the only approach that consistently produced the best predictions. In most 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 yielded improvement in predictive accuracy averaged over 20 trials as often as high as 2-3%, 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. Also the growing of option trees sometimes involved extra orders of magnitude in time and space. Although this only occurred for small samples, or where trees were inappropriate for the learning problem (like XD6 which is a DNF concept poorly expressed using a tree). 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-step 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 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 [Quinlan and Rivest, 1989] introduce a parameter that allows lighter pruning. So all approaches Bayesian and non-Bayesian have parameters that allow more or less pruning. These can be set 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.

[2]The tree algorithms were written in C and integrated with an experiment control package. The entire suite is available from the author.

4 Discussion

4.1 Multiple models

Machine learning research, as with classical statistics and minimum encoding methods, has been largely concerned to date with trying to find the best single tree, the best non-redundant rule set, or the best relational rule explaining the data.

Several researchers have now reported being able to significantly improve learning performance by instead working with multiple models. This is an approximate method for doing the averaging presented in Section 2, and is different from the technique of combining independent sources of knowledge multiplicatively, using the probability formula for independence (the basis of "idiot" Bayes classifiers).

Kwok and Carter used a heuristic approximation to Bayesian decision theory [Kwok and Carter, 1990]; they built multiple decision trees and when processing a new example, processed it with each tree individually and averaged the multiple class predictions. Gams discussed the notion of "redundant knowledge" [Gams, 1989], where he weighs up the predictions of several overlapping rules when classifying a new example. Jacobs *et al.* present another approach that does adaptive mixing of multiple feed-forward networks [Jacobs *et al.*, 1991]. A survey of some related theoretical work in "aggregrating learning strategies" and "weighted majority" is given in [Haussler *et al.*, 1991].

Suggested modifications to learning algorithms rarely perform consistently better (see for instance [Mingers, 1989]). The most striking thing about the use of multiple models is that it does appear to give consistently better performance, and can often be implemented as a control module on top of an existing algorithm.

4.2 Incremental learning algorithms

Incremental learning applies when new training examples are continually being supplied and we wish to update our current hypothesis(es) given the few additional examples. The contrasting approach is batch learning where examples are supplied in one batch. The major approach for developing an incremental algorithm is to modify a batch learning algorithm. This has the advantage that the long sequence of theoretical and empirical work that led to the development of the algorithm is not wasted. Some algorithms, however, are designed to be

incremental from the beginning [Gennari *et al.*, 1990]. These algorithms can suffer from order-sensitivity [Langley and McKusick, 1990], which is an incremental manifestation of the overfitting problem, a problem which is largely solved for batch algorithms.

There are two broad cases where a batch algorithm can be easily converted to an incremental algorithm. In the first case, Bayesian classifiers and many classical statistical methods are *naturally incremental* because they are calculated from simple summary statistics such as means and variances that are readily updated with additional data. These summary statistics are an example of sufficient statistics, discussed in most advanced statistical texts. In the second case, Perceptrons, most neural net methods, Autoclass [Cheeseman *et al.*, 1988], and many classical statistical clustering algorithms use *iterative convergence* methods. These are not naturally incremental, because their performance is generally poor with only one iteration of the training sample. But because of their iterative nature, they can be easily modified to form incremental versions. For instance, one could add the new training data to the next iteration.

Some batch algorithms do not lend themselves naturally to incremental versions. In these cases, as done with trees [Schlimmer and Granger Jr., 1986], the batch learning algorithm is *differentiated*. That is, an incremental algorithm is designed that attempts to do the least amount of work to modify a tree given new data so that it looks as if the tree was constructed from the entire training sample using the corresponding batch algorithm. Crawford reports this leads to the problem of *repeated restructuring* [Crawford, 1989]. This occurs in trees when some subtree is repeatedly restructured during incremental updating due to vacillation in what is currently considered the "best" test at the root of the subtree. This occurs particularly with learning noisy concepts (earlier incremental studies looked at logical concepts). The Bayesian methods offer a generic remedy for this problem. Essentially, we only restructure if we strongly believe the new substructure will be better, rather than immediately restructuring the moment some new substructure seems slightly better. The logarithm of the lookahead measure given in Table 1 returns a measure whose units are in log-odds. So one can easily implement an algorithm that only changes the current test at a node if there is another test that has a good log-odds (e.g. > 1.0) of being better. This would prevent vacillation, and experiments show it does not unduly effect predictive accuracy.

4.3 Comparisons of learning theories

A basic tool of learning theory in pattern recognition and computational learning theory is uniform convergence. If a sample size is large enough, uniform convergence theory provides bounds on the predictive performance of classifiers learned by minimizing empirical error [Vapnik, 1982]. In practice when sample sizes are not large enough, one needs to make a tradeoff between the complexity of the hypothesis chosen and the accuracy of the fit to the data. Here, uniform convergence methods give less guide, only asymptotic (i.e. large sample) theory regarding their performance (see, for instance, the principle of structural risk minimization in [Vapnik, 1982]). These techniques have no theoretical justification that they will provide good average-case performance on smaller samples. Some experimental comparisons are given in [Buntine, 1990c]. Learning theoreticians are now using Bayesian methods [Haussler *et al.*, 1991] to analyse the smaller sample case, as suggested earlier by Buntine [Buntine, 1989].

Statisticians overcome these overfitting problems with a variety of resampling techniques (as applied to trees, see [Breiman *et al.*, 1984; Crawford, 1989]) that have good intuition and performance, but again only have asymptotic theory. The tree experiments (using cross-validation) show these techniques work well, however they can overprune, so do not have the consistency of the full Bayesian approach when sample sizes are smaller. They are quick to code in many cases and hence offer a viable alternative.

Only Bayesian decision theory is able to claim that it is the most rational alternative in the information poor environment of learning from smaller samples [Berger, 1985]. Minimum encoding approaches [Rissanen, 1989] are sometimes touted as alternatives, however they are, mathematically, an interpretation of the Bayesian "most probable model" approach, which itself is an approximation to Bayesian decision theory [Wallace and Freeman, 1987; Buntine, 1990c]. Experiments support this approximation view, and also indicate the minimum encoding approximation can degrade significantly as the sample size decreases. This happens because the encoding methods do not consider multiple models, as suggested here.

In Bayesian theory, priors are viewed as assumptions that are essential when making inference from limited information such as a small training sample. Uniform convergence theory constrains samples so they are large enough to make the effect of the prior assumptions negligible [Buntine, 1990c, Lemma 4.2.1]. According to Bayesian theory, methods applied to smaller samples have implicitly built in particular assumptions which correspond to a choice of prior. Breiman *et al.*'s cost complexity pruning with cross validation and the 0-SE rule, for instance, favors smaller trees. And the various minimum encoding approaches [Rissanen, 1989; Quinlan and Rivest, 1989] have a very strong preference towards smaller trees. Bayesian methods differ in that they make these unavoidable prior assumptions explicit, allowing them to be specified by the user, or providing fairly objective assumptions as a fall-back.

5 Learning Bayesian Networks

To illustrate the algorithm strategy again, I will outline the development of an algorithm for learning Bayesian networks. This yields a one-step lookahead heuristic search algorithm, with smoothing on the final structure. The algorithm is analogous to the tree algorithms just presented. A similar method has been independently developed and implemented by Herskovits and Cooper and they report good experimental results [Cooper and Herskovits, 1991]. The different approach of Geiger *et al.* is concerned with learning networks from known depen-

dency information [Geiger *et al.*, 1990] (for instance, as extracted from a large sample) so is not relevant to the problem of learning from a smaller sample when dependency information is uncertain.

Bayesian networks in their simplest formulation specify dependence properties between variables by using a directed acyclic graph. They describe probabilistic models useful for non-directed classification. Figure 1 shows

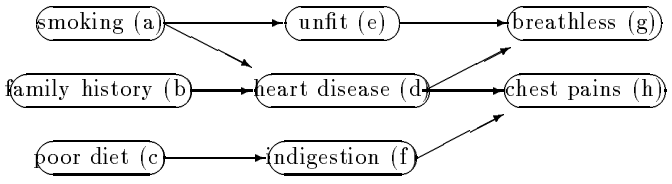

Figure 1: Bayesian network for a simple system

a simple Bayesian network. The set of variables that have outgoing arcs to a variable are called the parents of the variable. These parents specify the network structure. Each variable also has an associated conditional probability table which gives probabilities for different values of the variable given values of its parent variables. For instance for the graph in the figure, we need values for $Pr(e|a)$, $Pr(d|a,b)$, $Pr(g|e,d)$, etc. The parent structure and the conditional probabilities specify the model, as given by the likelihood function in Table 2. For ease of presentation, the learning algorithm presented here assumes variables are binary; this can be easily extended to multivalued or to real-valued variables.

The following notation is used. A Bayesian network consists of a set of binary discrete variables $\mathcal{X}$ where each variable $x \in \mathcal{X}$ has a set of parent variables Π_x. Assume variables take the values 1 or 2. For instance, for the graph in the figure, $\Pi_e = \{a\}$, $\Pi_d = \{a, b\}$, etc. The set of values for the cartesian product of variables in Π_x is $v(\Pi_x)$. The number of examples in the training sample *Sample* with $x = i$ and $\Pi_x = j$ is $n_{x=i|j}$, assuming every example in *Sample* has variable values fully specified. The conditional probabilities for variables are given by the probabilities θ as specified in the likelihood function in Table 2.

The learning algorithm is developed analogously to the tree learning algorithm. The corresponding results of the Bayesian analysis are given in Table 2. The network model has a structural component Π, the parent function, and a continuous component θ, the conditional probabilities. The prior on the continuous component is similar to the tree prior. A form for $Pr(\Pi)$ is not given in the table; it could be chosen to be uniform, or take some other simple form. A one-step lookahead heuristic search procedure for growing a high posterior structure can be developed analogous to the tree case. Start with the trivial structure where no variables have parents, and repeatedly add a new parent to maximize the posterior at each stage. Notice the adding of parents also has to be constrained so that the resultant graph has no cycles. A suitable heuristic measure for lookahead is given in the table. This gives the increase in posterior due to making y a parent of x. Notice the lookahead values calculated for a variable x remain unchanged if a new parent has just been added to another variable x'. This means at each cycle when choosing the next best parent to add, little recalculation needs to be done.

Acknowledgements

Thanks to Ross Quinlan especially, also Padhraic Smyth, Peter Cheeseman and Robin Hanson. University of Strathclyde and Turing Institute provided some support.

References

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

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

[Buntine and Weigend, 1991] W.L. Buntine and A.S. Weigend. Bayesian back-propagation. Submitted, 1991.

[Buntine, 1989] W.L. Buntine. A critique of the Valiant model. In *International Joint Conference on Artificial Intelligence*, pages 837–842, Detroit, 1989. Morgan Kaufmann.

[Buntine, 1990a] W.L. Buntine. Learning classification trees. Technical Report FIA-90-12-19-01, RIACS and NASA Ames Research Center, Moffett Field, CA, 1990. Paper presented at Third International Workshop on Artificial Intelligence and Statistics.

[Buntine, 1990b] W.L. Buntine. Myths and legends in learning classification rules. In *Eighth National Conference on Artificial Intelligence*, pages 736–742, Boston, Massachusetts, 1990.

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

[Cestnik *et al.*, 1987] 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, Bled, Yugoslavia, 1987. Sigma Press.

[Cheeseman *et al.*, 1988]
P. Cheeseman, M. Self, J. Kelly, W. Taylor, D. Freeman, and J. Stutz. Bayesian classification. In *Seventh National Conference on Artificial Intelligence*, pages 607–611, Saint Paul, Minnesota, 1988.

[Cooper and Herskovits, 1991] G. Cooper and E. Herskovits. A Bayesian method for the induction of probabilistic networks from data. Technical Report KSL-91-02, Knowledge Systems Laboratory, Medical Computer Science, Stanford University, 1991.

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

structure	Π is the parent function specifying how variables are conditionally dependent on other variables, as characterised by the likelihood
likelihood	$Pr(\mathcal{X} \mid \Pi, \theta) \;=\; \prod_{x \in \mathcal{X}} Pr(x \mid \Pi_x)$ where $Pr(x = i \mid \Pi_x = j)$ is given by $\theta_{x=i\mid j}$
prior	$Pr(\Pi, \theta) \;=\; Pr(\Pi) \prod_{x \in \mathcal{X}} \prod_{j \in v(\Pi_x)} \frac{\theta^{\alpha}_{x=1\mid j} \theta^{\alpha}_{x=2\mid j}}{B(\alpha, \alpha)}$ for $\alpha = 0.5$
sufficient statistics	$n_{x=i\mid j}$ = number of examples in *Sample* with $x = i$ and $\Pi_x = j$
posterior	$Pr(\Pi \mid Sample) \;\propto\; Pr(\Pi) \prod_{x \in \mathcal{X}} \prod_{j \in v(\Pi_x)} \frac{B(n_{x=1\mid j} + \alpha, n_{x=2\mid j} + \alpha)}{B(\alpha, \alpha)}$
lookahead heuristic	$\frac{Pr(\Pi \cup \{y \rightarrow x\} \mid Sample)}{Pr(\Pi \mid Sample)}$ = increase in posterior with y a parent of x $= \frac{Pr(\Pi \cup \{y \rightarrow x\})}{Pr(\Pi)} \prod_{j \in v(\Pi_x)} \frac{\prod_{i \in v(y)} B(n_{x=1\mid j, y=i} + \alpha, n_{x=2\mid j, y=i} + \alpha)}{B(n_{x=1\mid j} + \alpha, n_{x=2\mid j} + \alpha)\; B(\alpha, \alpha)}$
estimates	$\mathbf{E}_{\theta \mid \Pi, Sample}\left(\theta_{x=i\mid j}\right) \;=\; \frac{n_{x=i\mid j} + \alpha}{n_{x=1\mid j} + n_{x=2\mid j} + 2\alpha}$
smoothing	because the posterior is a product over the variables in $\mathcal{X}$, each conditional probability distribution $Pr(x \mid \Pi)$ can be smoothed individually without considering others

Table 2: Bayesian analysis of learning Bayesian networks

[Gams, 1989] M. Gams. New measurements highlight the importance of redundant knowledge. In K. Morik, editor, *Proceedings of the Fourth European Working Session on Learning*, pages 71–80, Montpellier, 1989. Pitman Publishing.

[Geiger *et al.*, 1990] D. Geiger, A. Paz, and J. Pearl. Learning causal trees from dependence information. In *Eighth National Conference on Artificial Intelligence*, pages 770–771, Boston, Massachusetts, 1990.

[Gennari *et al.*, 1990] J.H. Gennari, P. Langley, and D. Fisher. Models of incremental concept formation. *Artificial Intelligence*, 1990.

[Haussler *et al.*, 1991] D. Haussler, M. Kearns, and R.E. Schapire. Unifying bounds on the sample complexity of Bayesian learning using information theory and the VC dimension. In *COLT'91: 1991 Workshop on Computational Learning Theory*. Morgan Kaufmann, 1991. To appear.

[Jacobs *et al.*, 1991] R.A. Jacobs, M.I. Jordan, S.J. Nowlan, and G.E. Hinton. Adaptive mixtures of local experts. *Neural Computation*, 3(1), 1991.

[Kwok and Carter, 1990] 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.

[Langley and McKusick, 1990] P. Langley and K. McKusick. Personal communication, 1990.

[Lauritzen and Spiegelhalter, 1988] 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.

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

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

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

[Quinlan, 1988] 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.

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

[Schlimmer and Granger Jr., 1986] J.C. Schlimmer and R.H. Granger Jr. Incremental learning from noisy data. *Machine Learning*, 1(3):317–354, 1986.

[Shapiro, 1987] A. Shapiro. *Structured Induction in Expert Systems*. Addison Wesley, London, 1987.

[Tou and Gonzalez, 1974] J.T. Tou and R.C. Gonzalez. *Pattern Recognition Principles*. Addison-Wesley, USA, 1974.

[Vapnik, 1982] V. Vapnik. *Estimation of Dependencies Based on Empirical Data*. Springer-Verlag, New York, 1982.

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