Applying General Bayesian Techniques to Improve TAN Induction

Jesús Cerquides [1]
Ubilab
UBS AG
Bahnhofstrasse 45
P.O. Box, CH-8098 Zürich
Jesus.Cerquides@ubs.com

Abstract

Tree Augmented Naive Bayes (TAN) has shown to be competitive with state-of-the-art machine learning algorithms [3]. However, the TAN induction algorithm that appears in [3] can be improved in several ways. In this paper we identify three shortcomings in it and introduce two ideas to overcome those problems: the *multinomial sampling approach* to learning bayesian networks and *local bayesian model averaging*. These ideas approaches are generic and can thus be reused to improve other learning algorithms. We empirically test the new algorithms, and conclude that in most of the cases they lead to an improvement in accuracy in the classification and in the quality of the probabilities given as predictions.

1 Introduction

Tree Augmented Naive Bayes (TAN) has shown to be competitive with state-of-the-art machine learning algorithms. In this paper we analyze the TAN induction method proposed in [3]. We have identified three shortcomings where corrections can lead to a more coherent and accurate classifier:

1. The use of *ad hoc* softening of probabilities is not understandable from a theoretical point of view.
2. The algorithm selects a single model, ignoring model uncertainty.
3. The algorithm tries to find the TAN that maximizes likelihood, while in order to have better classification accuracy conditional likelihood should be maximized.

[1] The work was partially developed at the Institut de Investigació en Intel.ligència Artificial, CSIC, Spain.

In the following, we propose solutions that try to overcome these problems. We first introduce bayesian networks and propose a new approach (the *multinomial sampling approach*) to the problem of learning bayesian networks in Section 2. In Section 3 we discuss TAN into detail and use the *multinomial sampling approach* to derive an algorithm for learning maximum likelihood networks that uses a unique coherent probability distribution at every step and competes in accuracy with the one proposed in [3]. In order to overcome problems two and three, we consider bayesian model averaging (BMA) in 4.1; we introduce *local bayesian model averaging* (LBMA) in 4.2 and we make the application of LBMA to TAN induction in 4.3. We empirically evaluate the results of our improvements in Section 5, to finish up pointing some conclusions in Section 6. A longer version of this paper is available [2].

2 Learning Bayesian Networks

Let $\mathbf{U} = \{X_1, \ldots, X_n\}$ be a set of discrete random variables. A bayesian network is an annotated directed acyclic graph that encodes a joint probability distribution over $\mathbf{U}$. Formally it is a pair $B = \langle G, \Theta \rangle$. G is a DAG whose vertices correspond to the random variables and whose edges represent direct dependencies between the variables. Θ represents the set of parameters that quantify the network. It contains a parameter $\theta_{x_i|\Pi_{x_i}} = P_B(x_i|\Pi_{x_i})$ for each possible value x_i of X_i and Π_{x_i} of Π_{X_i}, where Π_{X_i} denotes the set of parents of X_i in G. A bayesian network B defines a unique joint probability distribution over $\mathbf{U}$ given by

$$P_B(X_1, \ldots, X_n) = \prod_{i=1}^{n} P_B(X_i|\Pi_{X_i}) = \prod_{i=1}^{n} \theta_{X_i|\Pi_{X_i}} \quad (1)$$

The problem of learning a bayesian network has been informally stated as:

Statement 1 (Friedman et al., 1997) *Given a training set $D = \{u_1, \ldots, u_N\}$ of instances of $\mathbf{U}$ find the network B that best matches D.*

For classification purposes, the fact of trying to match the data perfectly usually causes overfitting. That is why, in order to use bayesian networks for classification purposes we prefer the following informal statement:

Statement 2 *Given a sample $S = \{u_1, \ldots, u_N\}$ of a probability distribution P^* find the network B that best matches P^*.*

One of the measures used to learn bayesian networks is the *log likelihood*:

$$LL(B|D) = \sum_{i=1}^{N} \log(P_B(u_i)) \tag{2}$$

This measure has the property that can be decomposed according to the structure of B. Let $Count_D(X)$ stand for the number of observations in our sample that fulfill condition X and $Freq_D(X) = \frac{Count_D(X)}{N}$. $Val(X_i)$ is the set of possible states of the random variable X_i and $\#Val(X_i)$ the number of different possible states X_i can be in.
If $Freq_D(x_i, \Pi_{x_i})$ is strictly positive and defined everywhere (i.e. we have at least one observation for each possible pair x_i, Π_{x_i} in our dataset) it is easy to see that $LL(B|D)$ is maximized when

$$\theta_{x_i|\Pi_{x_i}} = \frac{Freq_D(x_i, \Pi_{x_i})}{Freq_D(\Pi_{x_i})} \tag{3}$$

This results allow us to search separately the space of network structures and the space of network parameters.

2.1 The multinomial sampling approach to learning bayesian networks

The dataset D of Statement 1 can be seen as a sample of a probability distribution P^*. We can assume that P^* is a multinomial distribution with a number of possible states equal to: $States(P^*) = \prod_{i=1}^{n} \#Val(X_i)$. Each instance in the dataset is equivalent to the observation of a concrete state as an outcome of a multinomial trial. In order to learn a bayesian network we should take two steps:

1. Approximate the distribution P^* from the information we have in the sample S by using multinomial sampling methods. This generates P_S^*.
2. Find the bayesian network which better fits P_S^*.

We need a way to calculate P_S^*. We adhere to the *principle of indifference*, which says that if we lack better information, we should assign an equal probability to each possible success. Our prior is then a Dirichlet distribution with $States(P^*)$ equiprobable possible states. We still have to fix one more parameter, the relevance we are giving to the prior, namely λ. Hence:

$$P_S^*(x_1, \ldots, x_n) = \frac{Count_S(x_1, \ldots, x_n) + \frac{\lambda}{States(P^*)}}{N + \lambda} \tag{4}$$

3 Tree Augmented Naive Bayes

From our point of view TAN are the most coherent and best performing enhancement to *Naive Bayes* up to now. To talk of the classification problem we will use the common notation of for distinguishing between the random variable we want to predict (the class, C) and all the rest (the attributes, $A_1, \ldots, A_n$).

3.1 Learning TAN

TAN are a restricted family of bayesian networks in which the class variable has no parents and each attribute has the class variable and at most one other attribute as parents. The interesting property of this family is that we have an efficient procedure for identifying the structure of the network with maximum likelihood. In [3] we can find a procedure that builds a TAN B_T that maximizes $LL(B_T|D)$ and has time complexity $O(n^2 N)$. The search for the maximum likelihood structure is done using the result of Equation 3. Surprisingly, once the structure is found, they empirically noticed that softening the probabilities instead of using exactly the maximum likelihood estimates lead to an improvement in classification accuracy [3]. They take:

$$\theta^s_{x|\Pi_x} = \alpha \cdot Freq_D(x|\Pi_x) + (1 - \alpha) \cdot Freq_D(x)$$
$$\alpha = \frac{N \cdot Freq_D(\Pi_x)}{N * Freq_D(\Pi_x) + s} \tag{5}$$

The explanation given there (summarizing, that the number of observations is not enough to estimate reliably the probabilities) is not theoretically sound, because the results are not assymptotic. A plausible explanation to this phenomenon could be related to the fact (usually disregarded) that the results only hold when $Freq_D$ is strictly positive and defined everywhere. From our point of view, the most likely explanation comes from the fact that, by following Statement 1 we are focusing on fitting the data, and not on predicting future events. That is why we propose the usage of the *multinomial sampling approach* to provide a solution to the problem.

3.2 Applying the multinomial sampling approach to TAN induction

In this case, instead of looking for maximum likelihood, we would like to minimize the *cross entropy* or Kullback-Leibler divergence [6] between the P_S^* and the probability distribution generated by the TAN (following

the spirit of Statement 2). *Cross entropy* is minimized when:

$$\theta_{x_i|\Pi_{x_i}} = P^*_S(x_i|\Pi_{x_i}) = \frac{P^*_S(x_i,\Pi_{x_i})}{P^*_S(\Pi_{x_i})} =$$

$$= \frac{\sum\limits_{\overline{\Pi_{x_i}} \in Val(\overline{\Pi_{X_i}})} P^*_S(x_i,\Pi_{x_i},\overline{\Pi_{x_i}})}{\sum\limits_{\substack{x_i \in Val(X_i) \\ \overline{\Pi_{x_i}} \in Val(\overline{\Pi_{X_i}})}} P^*_S(x_i,\Pi_{x_i},\overline{\Pi_{x_i}})} \quad (6)$$

where $\overline{\Pi_{X_i}}$ stands for the set of variables which are not parents of X_i in the network. $\overline{\Pi_{X_i}}$ does not include X_i. We define $SC(X_i) = \prod\limits_{X_j \in \overline{\Pi_{X_i}}} \#Val(X_j)$. $SC(X_i)$ is the number of different states of the multinomial for which you have to add up the probability in order to calculate $P^*_S(x_i,\Pi_{x_i})$. From this definition and Equation 6 we get:

$$\theta_{x_i|\Pi_{x_i}} = \frac{Count_D(x_i,\Pi_{x_i}) + \lambda\frac{SC(X_i)}{States(P^*)}}{Count_D(\Pi_{x_i}) + \lambda\frac{\#Val(X_i)SC(X_i)}{States(P^*)}} \quad (7)$$

In the case of TAN induction, we have to set three different kind of parameters: $\theta_{a_i|c,a_j}$, $\theta_{a_i|c}$ and θ_c. In these concrete three cases Equation 7 simplifies to:

$$\theta_{a_i|c,a_j} = \frac{Count_D(a_i,c,a_j)+\frac{\lambda}{\#Val(C)\#Val(A_i)\#Val(A_j)}}{Count_D(c,a_j)+\frac{\lambda}{\#Val(C)\#Val(A_j)}}$$
$$\theta_{a_i|c} = \frac{Count_D(a_i,c)+\frac{\lambda}{\#Val(C)\#Val(A_i)}}{Count_D(c)+\frac{\lambda}{\#Val(C)}} \quad (8)$$
$$\theta_c = \frac{Count_D(c)+\frac{\lambda}{\#Val(C)}}{N+\lambda}$$

These probabilities estimates are used consistently both in the tree structure induction and in the weight determination. Empirical results are given and explained in Section 5.

4 Local Bayesian Model Averaging

The second shortcoming in the TAN induction algorithm of [3] is that uncertainty in model selection is ignored. *Bayesian model averaging* (BMA) [5] provides a coherent mechanism for accounting for uncertainty in modelling.

4.1 Bayesian Model Averaging

A coherent approach to solve the classification problem is calculating the probability of each class given the data. If we assume that the data has been generated from a model that is contained in a class of models $\mathcal{M}$, the probability distribution of the class given data, is:

$$P(C|I,S) = \sum_{M\in\mathcal{M}} P(C|M,I)P(M|S) \quad (9)$$

where $P(C|M,I)$ is the probability distribution of the class when we know the model M that generated the data, and the value of the attributes for this instance I. $P(M|S)$ is the probability that M is the model that generated the data given the sample S.
Equation 9 tells us not to use a single model to classify the data, but instead to use all the models from the class of models, weighting each model prediction by the probability of the model given the sample of data we are analyzing. Using Equation 9 to predict is known as *Bayesian Model Averaging* or BMA for short. BMA produces optimally accurate predictions within the chosen model family from the probability theory point of view.
In order to use BMA in practice, we need to further develop Equation 9. It can be expanded as:

$$P(M|S) = \frac{P(S|M)P(M)}{\sum\limits_{M'\in\mathcal{M}} P(S|M')P(M')} \quad (10)$$

Here $P(M)$ is the prior probability that M is the real model and $P(S|M)$ is the probability that model M generates the data in S.

4.2 Local Bayesian Model Averaging

In practice, the usage of BMA presents some problems, coming from:

- The computational cost of calculating Equation 9.
- The difficulty in the specification of $P(M)$, the prior distribution over competing models.

In order to handle the first of these problems, we propose LBMA, an heuristic approach to approximate BMA. The idea is similar in spirit to the Occam's Window method described in [5, 7]. To apply LBMA we should have an heuristic $h(M,S)$ such that

$$h(M,S) \approx P(M|S) \quad (11)$$

In order to approximate the summation in Equation 9, and given that we have $h(M,S)$, we define our set of *interesting models* $\mathcal{M}'$ as:

$$\mathcal{M}' = \{M \in \mathcal{M} | h(M,S) \geq \gamma\} \quad (12)$$

γ represents a compromise between the prediction accuracy and its computational cost. It should be big enough to make $\#\mathcal{M}' \ll \#\mathcal{M}$, but small enough in order for

$$P(M|S) \approx P'(M|S) = \frac{P(S|M)P(M)}{\sum\limits_{M'\in\mathcal{M}'} P(S|M')P(M')} \quad (13)$$

$$P(C|I,S) \approx \sum_{M\in\mathcal{M}'} P(C|M,I)P'(M|S) \quad (14)$$

to be accurate approximations. It is interesting to note that maximum likelihood prediction is a concrete case of LBMA where $h(M,S) = P(S|M)$ and γ is implicitly set in order for $\mathcal{M}'$ to contain only a model. Once we have the resulting weighted set of models calculated, we can use it to classify by calculating Equation 14 for each class and choosing the one with higher probability.

4.3 Local Bayesian Model Averaging for TAN induction

For this concrete case, our class of models $\mathcal{M}$ is

$$\mathcal{M} = \{\langle G, \Theta\rangle | G \in TANStructs, \Theta \in Params(G)\} \tag{15}$$

We perform a first reduction of $\mathcal{M}$ by using the results in Equation 8 or the *softened* method proposed in [3] depending on whether we decide to use the *multinomial sampling approach* or the ad hoc adjustment proposed in [3]. In any case, we will only average over the structures, fixing the parameters by using the corresponding equation in each case. Our heuristic over structures will be given by the algorithm `Construct-TAN`, just modifying the step where a maximum spanning tree is induced, to generate a set containing the K maximum spanning trees by using Gabow algorithm [4]. In order to calculate $P'(M)$, we set a prior over tree structures that assigns the same probability to each possible tree structure (since they can be considered of a similar complexity). We also have to provide an implementation for `Predict`, that is, we have to know how to calculate $P(C = c|M, u)$. In a TAN:

$$P_M(c|u_{i,1},\ldots,u_{i,n}) = \frac{P_M(c,u_{i,1},\ldots,u_{i,n})}{\sum_{c'\in C} P_M(c',u_{i,1},\ldots,u_{i,n})} = \\ = \frac{\prod_{j=1}^{n} \theta_{u_{i,j}|c,u_{i,\pi(A_j)}}}{\sum_{c'\in C}\prod_{j=1}^{n} \theta_{u_{i,j}|c',u_{i,\pi(A_j)}}} \tag{16}$$

where $u_{i,\pi(A_j)}$ is adequately set to the value of the parent of A_j in the tree or nothing if A_j does not has parents in it.

Applying LBMA to TAN induction we are simultaneously providing a solution for the second and third shortcomings in [3] noticed in the introduction. It is clear that LBMA is an approximation to BMA, and hence focuses on the second point, i. e., taking into account the uncertainty in model selection. The third point was that what should be maximized is the conditional likelihood instead of the likelihood. This shortcoming was already noticed in their paper, where they stated that it was an open question whether good heuristic approaches can be found in order to induce TAN models that maximize conditional likelihood. The way LBMA weights the different models is exactly multiplying each model by its conditional likelihood. We are then trusting more those models with a higher conditional likelihood. The problem is far from being solved, but we think this is a good first step.

5 Experimental results

In order to use the algorithm described in Section 4.3, we need to set some parameters. In our experimental setting, we took:

$$\begin{array}{c} k = \min(10, n) \\ \lambda = 10 \end{array} \tag{17}$$

We tested three algorithms over 11 datasets from the Irvine repository [1] plus our own credit screening database. To discretize continuous attributes we used equal frequency discretization with 5 intervals. For each dataset and algorithm we evaluated accuracy and *LogScore*. *LogScore* gives an idea of how well the classifier is estimating probabilities.

$$LogScore(M, S') = \sum_{i=1}^{N'} -\log(P_M(u'_i)) \tag{18}$$

For both evaluations we used 10 fold cross validation. The error rates appear in Table 1, with the best method for each dataset in cursive. *LogScore*'s appear in Table 2. The meaning of the column headers are:

- TAN+MS+BMA, is the method described in Section 4.3.
- sTAN is the softened TAN algorithm as described in [3]
- TAN+MS is the maximum likelihood TAN induction using the *multinomial sampling approach*

Dataset	TAN+MS+BMA	sTAN	TAN+MS
DCredits	14.16 ± 1.58	14.29 ± 1.27	*13.98 ± 1.49*
flare	20.02 ± 3.16	*18.86 ± 3.11*	20.09 ± 3.07
hep	20.96 ± 12.56	*20.14 ± 8.54*	21.81 ± 13.94
mushroom	*0.12 ± 0.00*	0.14 ± 0.04	*0.12 ± 0.00*
votes	*7.60 ± 3.33*	8.49 ± 4.40	8.26 ± 4.18
car	*5.85 ± 1.37*	6.25 ± 1.36	*5.85 ± 1.37*
iris	13.39 ± 5.82	*12.84 ± 5.18*	13.61 ± 4.92
nursery	*4.90 ± 1.15*	6.76 ± 1.33	6.90 ± 1.39
pima	*26.16 ± 4.85*	26.19 ± 4.99	26.17 ± 5.13
adult	16.43 ± 0.61	*16.34 ± 0.63*	16.42 ± 0.62
chess	*7.53 ± 1.33*	8.15 ± 1.52	7.63 ± 1.44
soybean	*7.31 ± 4.22*	8.01 ± 3.58	7.90 ± 3.92

Table 1: Averages and standard deviations of error rates

5.1 Interpretation of the results

We plot in Figure 1 the percentage of improvement in error rate between using TAN+MS+BMA and sTAN, and in Figure 2 the improvement in *LogScore*. Both figures favor TAN+MS+BMA.

Dataset	TAN+MS+BMA	sTAN	TAN+MS
DCredits	160.10 ± 20.07	158.81 ± 13.21	***158.24 ± 18.14***
flare	***84.37 ± 18.43***	∞	86.35 ± 19.78
hep	2.38 ± 2.24	***1.73 ± 1.20***	3.04 ± 2.94
mushroom	***0.01 ± 0.01***	0.36 ± 0.51	0.02 ± 0.03
votes	8.58 ± 7.12	8.74 ± 6.30	***8.46 ± 7.04***
car	31.98 ± 5.69	37.47 ± 6.35	***31.96 ± 5.68***
iris	***2.34 ± 1.45***	2.58 ± 1.22	2.87 ± 1.65
nursery	***199.97 ± 18.18***	211.23 ± 19.89	206.03 ± 19.40
pima	41.31 ± 9.35	***39.48 ± 8.88***	41.29 ± 10.08
adult	1186.25 ± 33.48	***1170.94 ± 32.20***	1186.36 ± 33.17
chess	58.43 ± 5.86	60.46 ± 6.15	***58.38 ± 5.75***
soybean	16.04 ± 9.78	***12.37 ± 5.30***	16.69 ± 9.77

Table 2: Averages and standard deviations of *LogScore*

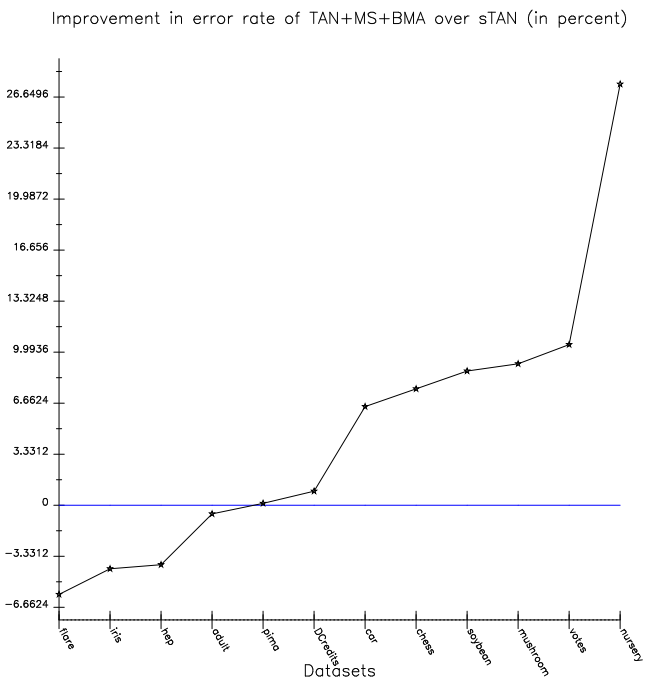

Figure 1: Comparison of the error rate of TAN+MS+BMA and sTAN

6 Conclusions

We have proposed solutions for correcting the three shortcomings we noticed in TAN induction as it is done in [3]. We have introduced the *multinomial sampling approach*, a new approach to learning bayesian networks that provides a coherent way of estimating probabilities, and have used it to develop a theoretically coherent maximum likelihood TAN induction algorithm. We have introduced *local bayesian model averaging* and have used it to account for uncertainty in the selection of the model. By using LBMA, we have weighted each model by its conditional likelihood, instead of by its likelihood, thus providing a partial solution to the fact that conditional likelihood and not likelihood is what should be maximized.

We have provided empirical evidence that shows that in most of the cases the resulting new method provides more accurate predictions and probability estimates.

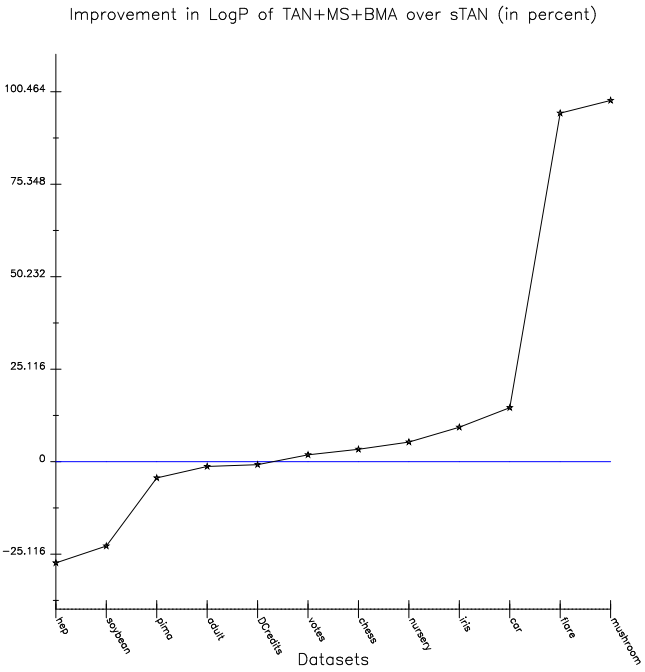

Figure 2: Comparison of the *LogScore* of TAN+MS+BMA and sTAN

7 Acknowledgements

I would like to thank Maria Luisa Barja and Ramon López de Mántaras for carefully reviewing the preliminary versions of this paper. I would like to specially thank Maria Luisa for accepting an additional workload in order to give me the time to write it.

References

[1] C. Blake, E. Keogh, and C. J. Merz. UCI repository of machine learning databases, 1998.

[2] Jesús Cerquides. Applying General Bayesian Techniques to Improve TAN Induction. Long Version. Technical report, Ubilab, www.ubs.com/ubilab, 1999.

[3] Nir Friedman, Dan Geiger, and Moises Goldszmidt. Bayesian network classifiers. *Machine Learning*, 29:131–163, 1997.

[4] Harold N. Gabow. Two algorithms for generating weighted spanning trees in order. *SIAM J. COMPUT.*, 6(1):139–150, March 1977.

[5] Jennifer A. Hoeting, David Madigan, Adrian E. Raftery, and Chris T. Volinsky. Bayesian model averaging. Technical Report 9814, Department of Statistics. Colorado State University, 1998.

[6] S. Kullback and R. A. Leibler. On information and sufficiency. *Annals of Mathematical Statistics*, 22:76–86, 1951.

[7] David Madigan and Adrian. E. Raftery. Model selection and accounting for model uncertainty in graphical models using occam's window. *J. American Statistical Association*, 89:1535–1549, 1994.