Knowledge-Based Systems 26 (2012) 239–245

Contents lists available at SciVerse ScienceDirect

Knowledge-Based Systems

journal homepage: www.elsevier.com/locate/knosys

Improving Tree augmented Naive Bayes for class probability estimation

Liangxiao Jiang [a], Zhihua Cai [a,*], Dianhong Wang [b], Harry Zhang [c]

[a] *Department of Computer Science, China University of Geosciences, Wuhan, Hubei 430074, China*
[b] *Department of Electronic Engineering, China University of Geosciences, Wuhan, Hubei 430074, China*
[c] *Faculty of Computer Science, University of New Brunswick Fredericton, New Brunswick, Canada E3B5A3*

ARTICLE INFO

Article history:
Received 23 February 2011
Received in revised form 16 August 2011
Accepted 19 August 2011
Available online 30 August 2011

Keywords:
Naive Bayes
Tree Augmented Naive Bayes
Class probability estimation
Conditional log likelihood
Ensemble learning

ABSTRACT

Numerous algorithms have been proposed to improve Naive Bayes (NB) by weakening its conditional attribute independence assumption, among which Tree Augmented Naive Bayes (TAN) has demonstrated remarkable classification performance in terms of classification accuracy or error rate, while maintaining efficiency and simplicity. In many real-world applications, however, classification accuracy or error rate is not enough. For example, in direct marketing, we often need to deploy different promotion strategies to customers with different likelihood (class probability) of buying some products. Thus, accurate class probability estimation is often required to make optimal decisions. In this paper, we investigate the class probability estimation performance of TAN in terms of conditional log likelihood (CLL) and present a new algorithm to improve its class probability estimation performance by the spanning TAN classifiers. We call our improved algorithm Averaged Tree Augmented Naive Bayes (ATAN). The experimental results on a large number of UCI datasets published on the main web site of Weka platform show that ATAN significantly outperforms TAN and all the other algorithms used to compare in terms of CLL.

1. Introduction

Classification is one of the most important tasks in data mining. In classification, a classifier is built from a set of training instances with class labels. The predictive ability of a classifier is typically measured by its classification accuracy or error rate on the testing instances. In fact, probability-based classifiers can also produce probability estimates or “confidence” of the class prediction. Unfortunately, this information is often ignored in classification.

In many real-world applications, however, the accuracy or error rate of a classifier is not enough, because they cannot express the information how “far-off” (be it 0.99 or 0.01?) is the prediction of each instance from its target. For example [1], in target marketing, the estimated probability that a customer will respond to an offer is combined with the estimated profit to evaluate various offer propositions. For another example [2], in cost-sensitive learning, the class probability estimation is used to minimize the conditional risk [3]. Besides, the estimated class membership probabilities are often used for probability-based ranking [4–6]. Thus, accurate class probability estimation is required, instead of just a classification [2,7].

A natural question is how to evaluate a probability-based classifier in terms of its class probability estimation performance, rather than classification accuracy or error rate. Recently, conditional log likelihood [7–9], or simply CLL, has been used for this purpose and received a considerable attention. In this paper, our task is to learn a probability-based classifier with maximal CLL score when the accurate class probability estimation is needed.

Given a classifier G and a set of test instances $T = \{e_1, e_2, \ldots, e_i \ldots, e_t\}$, where t is the number of test instances. Let c_i be the true class label of e_i. Then, the conditional log likelihood $CLL(G|T)$ of the classifier G on the test instance set T is defined as:

$$CLL(G|T) = \sum_{i=1}^{t} log\widehat{P}_G(c_i|e_i). \tag{1}$$

Let e, represented by an attribute vector $\langle a_1, a_2, \ldots, a_m \rangle$, be a test instance and the true class label of it be c, then we can use the built classifier G to estimate the probability that e belongs to c. This resulting probability is generally called the predicted probability, denoted by $\widehat{P}(c|e)$. On the other hand, we use $P(c|e)$ to denote the true probability that e belongs to c. Therefore, we can compute the difference between the true probability $P(c|e)$ and the predicted probability $\widehat{P}(c|e)$. This difference is just the class probability loss (simply CPL), defined as follows.

$$CPL(e) = P(c|e) - \widehat{P}(c|e). \tag{2}$$

In general, however, the true probability $P(c|e)$ is difficult to know in real-world data mining applications, because the information we have for c is only the class label. For simplicity, in this paper, we assume that $P(c|e) = 1$ when e is just in class c. Please note that this assumption may not be held if data can not be separated

* Corresponding author. Tel.: +86 27 67883714.
E-mail address: zhcai@cug.edu.cn (Z. Cai).

0950-7051/$ - see front matter © 2011 Elsevier B.V. All rights reserved.
doi:10.1016/j.knosys.2011.08.010

completely [10]. Now, we can see that the classifiers with higher CLL trend to have better class probability estimation performance, because the predicted probability $\widehat{P}(c|e)$ by them is more close to 1 and thus the estimated class probability loss is lower.

Now, the only question left to answer is how to estimate the class probability $\widehat{P}(c|e)$ via constructing probability-based classifiers. Bayesian network classifiers are typical probability-based classifiers, which estimate $\widehat{P}(c|e)$ using Eq. (3).

$$\widehat{P}(c|e) = \frac{\widehat{P}(c)\widehat{P}(a_1, a_2, \ldots, a_m|c)}{\widehat{P}(a_1, a_2, \ldots, a_m)}. \tag{3}$$

Assume all attributes are independent given the class. Then, the resulting classifier is called Naive Bayes (NB). NB estimates above probability using Eq. (4).

$$\widehat{P}(c|e) = \frac{\widehat{P}(c)\prod_{j=1}^{m}\widehat{P}(a_j|c)}{\sum_c \widehat{P}(c)\prod_{j=1}^{m}\widehat{P}(a_j|c)}, \tag{4}$$

where the prior probability $\widehat{P}(c)$ with Laplace correction is defined using Eq. (5), and the conditional probability $\widehat{P}(a_j|c)$ with Laplace correction is defined using Eq. (6).

$$\widehat{P}(c) = \frac{\sum_{i=1}^{n}\delta(c_i, c) + 1}{n + k}, \tag{5}$$

$$\widehat{P}(a_j|c) = \frac{\sum_{i=1}^{n}\delta(a_{ij}, a_j)\delta(c_i, c) + 1}{\sum_{i=1}^{n}\delta(c_i, c) + n_j}, \tag{6}$$

where n is the number of training instances, k is the number of classes, c_i is the class label of the ith training instance, n_j is the number of values of the jth attribute, a_{ij} is the jth attribute value of the ith training instance, a_j is the jth attribute value of the test instance, and $\delta(\bullet)$ is a binary function, which is one if its two parameters are identical and zero otherwise. Thus, $\sum_{i=1}^{n}\delta(c_i, c)$ is the frequency that the class label c occurrs in the training data and $\sum_{i=1}^{n}\delta(a_{ij}, a_j)\delta(c_i, c)$ is the frequency that the class label c and the attribute value a_j occurr simultaneously in the training data.

Fig. 1 shows graphically an example of Naive Bayes. In Naive Bayes, each attribute node has the class node as its parent, but does not have any parent from other attribute nodes. Because the values of $\widehat{P}(c)$ and $\widehat{P}(a_j|c)$ can be easily estimated from the training instances, Naive Bayes is easy to construct.

Naive Bayes is the simplest form of Bayesian network classifiers. It is obvious that the conditional independence assumption in Naive Bayes is rarely true in reality, which would harm its performance in the applications with complex attribute dependencies. Numerous algorithms have been proposed to improve Naive Bayes by weakening its conditional attribute independence assumption, among which Tree Augmented Naive Bayes (TAN) [11] has demonstrated remarkable classification performance and is competitive with the general Bayesian network classifiers [12] in terms of classification accuracy or error rate, while maintaining efficiency and simplicity.

However, its class probability estimation performance is unknown. Therefore, in this paper, we investigate its class probability estimation performance in terms of conditional log likelihood (CLL) and find that its class probability estimation performance also outperforms Naive Bayes. In order to make its class probability estimation performance stronger, we then present an improved algorithm by averaging all of the spanning TAN classifiers. We call the improved algorithm Averaged Tree Augmented Naive Bayes (ATAN). The experimental results on a large number of UCI datasets show that the ATAN significantly outperforms TAN in terms of CLL.

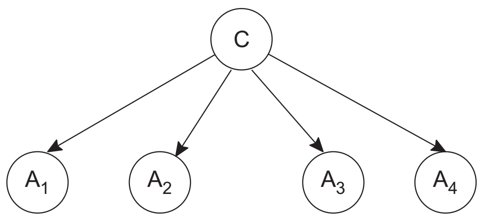

Fig. 1. An example of Naive Bayes.

The rest of the paper is organized as follows. At first, we briefly introduce Tree Augmented Naive Bayes (TAN) and some other state-of-the-art algorithms based on the structure extension approach. Then, we present our improved algorithm Averaged Tree Augmented Naive Bayes (ATAN). Followed by the description of our experiments and results in detail, we draw conclusions.

2. Tree Augmented Naive Bayes

In order to weaken the conditional independence assumption of Naive Bayes effectively, an appropriate language and efficient machinery to represent and manipulate independence assertions are needed [11]. Both are provided by Bayesian networks [13]. Unfortunately, it has been proved that learning an optimal Bayesian network is NP-hard [14]. In order to avoid the intractable complexity for learning Bayesian networks, learning improved Naive Bayes has attracted much attention from researchers, and several improved approaches have been proposed [15,16].

Among these improved approaches, structure extension is the most direct way to improve Naive Bayes, since attribute dependencies can be explicitly represented by arcs. Tree Augmented Naive Bayes (TAN) [11] is an extended tree-like Naive Bayes, in which the class node directly points to all attribute nodes and an attribute node only has at most one parent from another attribute node. Fig. 2 shows an example of TAN. TAN is a specific case of general Bayesian network classifiers [12,17], in which the class node also directly points to all attribute nodes, but there is no limitation on the arcs among attribute nodes (except that they do not form any directed cycle). In practice, TAN is a good trade-off between the model complexity and learnability. Assume that $A_1, A_2, \ldots, A_m$ are m attributes and C is the class variable, the learning algorithm of TAN [11] is depicted as:

```
Algorithm TAN (D)
Input: a training instance set D
Output: the built TAN
 1. Compute the conditional mutual information $I_{\widehat{P}_D}(A_i; A_j|C)$
    between each pair of attributes, $i \neq j$.
 2. Build a complete undirected graph in which nodes are
    attributes $A_1, \ldots, A_m$. Annotate the weight of an edge con-
    necting $A_i$ to $A_j$ by $I_{\widehat{P}_D}(A_i; A_j|C)$.
 3. Build a complete undirected maximum weighted spanning
    tree. An example of it is shown in Fig. 3.
 4. Transform the built undirected tree to a directed one by
    choosing a root attribute and setting the direction of all
    edges to be outward from it. For example, if $A_1$ is chosen
    as the root node, the resulting directed tree can be shown
    in Fig. 4.
 5. Build a TAN model by adding a node labeled by C and add-
    ing an arc from C to each $A_i$. Fig. 2 shows graphically the
    built TAN.
 6. Return the built TAN.
```

Keogh and Pazzani [18] present an algorithm called SuperParent to improve the classification accuracy of TAN. It applies a greedy heuristic search algorithm to add an arc of achieving the highest

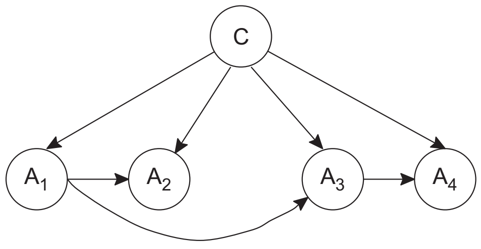

Fig. 2. An example of TAN.

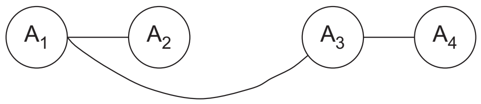

Fig. 3. An example of the undirected tree.

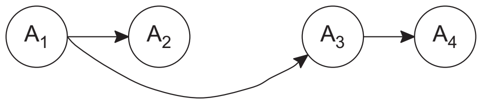

Fig. 4. The directed tree resulted by choosing A_1 as the root node.

classification accuracy improvement in each iteration until the adding of any one possible arc does not lead to the improvement of classification accuracy.

In 2001, Zhang and Ling [19] observed that the dependence among attributes tends to cluster into groups in many real-world domains with a large number of attributes. Based on their observation, they present an improved algorithm called StumpNetwork. Their motivation is to enhance the efficiency of the Superparent algorithm while maintaining a similar classification accuracy. Next year, they present another algorithm called AUC-Superparent [20] to improve the ranking performance of Superparent. In AUC-Superparent, AUC [21,22] is used for evaluating the current network instead of classification accuracy.

Besides, Averaged One-Dependence Estimators (AODE) [23] can also be viewed as an improved TAN algorithm. In AODE, a special TAN is firstly built for each attribute, in which the attribute is directly set to be the parent of all the other attributes. Then, AODE produces the prediction by directly averaging the predictions of all qualified TAN classifiers. Thus, AODE is an ensemble learning algorithm.

3. Averaged Tree Augmented Naive Bayes

The classification performance of TAN, in terms of classification accuracy or error rate, is well known. However, its class probability estimation performance, in terms of conditional log likelihood (CLL), is unknown. Therefore, in this paper, we investigate its class probability estimation performance and find that its class probability estimation performance is also much better than Naive Bayes.

Table 1
Descriptions of UCI datasets used in the experiments.

Dataset	Instances	Numeric attributes	Nominal attributes	Classes	Missing values
Anneal	898	6	32	5	N
Anneal.ORIG	898	6	32	6	Y
Audiology	226	0	69	24	Y
Autos	205	15	10	7	Y
Balance-scale	625	4	0	3	N
Breast-cancer	286	0	9	2	Y
Breast-w	699	9	0	2	Y
Colic	368	7	15	2	Y
Colic.ORIG	368	7	20	2	Y
Credit-a	690	6	9	2	Y
Credit-g	1000	7	13	2	N
Diabetes	768	8	0	2	N
Glass	214	9	0	7	N
Heart-c	303	6	7	5	Y
Heart-h	294	6	7	5	Y
Heart-statlog	270	13	0	2	N
Hepatitis	155	6	13	2	Y
Hypothyroid	3772	23	6	4	Y
Ionosphere	351	34	0	2	N
Iris	150	4	0	3	N
kr-vs-kp	3196	0	36	2	N
Labor	57	8	8	2	Y
Letter	20000	16	0	26	N
Lymph	148	3	15	4	N
Mushroom	8124	0	22	2	Y
Primary-tumor	339	0	17	21	Y
Segment	2310	19	0	7	N
Sick	3772	7	22	2	Y
Sonar	208	60	0	2	N
Soybean	683	0	35	19	Y
Splice	3190	0	61	3	N
Vehicle	846	18	0	4	N
Vote	435	0	16	2	Y
Vowel	990	10	3	11	N
Waveform-5000	5000	40	0	3	N
Zoo	101	1	16	7	N

We download these data sets from the main web site of Weka.

The detailed compared results can be found in Section 4. In order to make its class probability estimation performance stronger, we present an improved algorithm by averaging all of the spanning TAN classifiers. We call the improved algorithm Averaged Tree Augmented Naive Bayes (ATAN).

At the training time, ATAN respectively chooses each attribute as the root of the tree to directly built a complete directed maximum weighted spanning tree, and then constructs a TAN classifier using each built tree. At the test time, ATAN estimates the class-membership probabilities using each built TAN classifier, and then the estimated class-membership probabilities are averaged. Thus, the whole algorithm of ATAN can be partitioned into a training algorithm (**ATAN-Training**) and a test algorithm (**ATAN-Test**). They are depicted below respectively.

```
Algorithm ATAN-Training (D)
Input: a training instance set D
Output: $TAN_1, TAN_2, \ldots, TAN_m$
1. Compute the conditional mutual information $I_{\widehat{P}_D}(A_i; A_j|C)$ between each pair of attributes, $i \neq j$.
2. For each attribute $A_o$ ($o = 1, 2, \ldots, m$)
   (a) Choose $A_o$ as the root node to build a complete directed maximum weighted spanning tree in which nodes are attributes $A_1, A_2, \ldots, A_m$. Annotate the weight of an edge connecting $A_i$ to $A_j$ by $I_{\widehat{P}_D}(A_i; A_j|C)$.
   (b) Build $TAN_o$ by adding a node labeled by $C$ and adding an arc from $C$ to each attribute $A_i$.
3. Return the built $TAN_1, TAN_2, \ldots, TAN_m$.
```

```
Algorithm ATAN-Test ($TAN_1, TAN_2, \ldots, TAN_m, e$)
Input: the built $TAN_1, TAN_2, \ldots, TAN_m$ and a test instance $e$
Output: the probabilities $\widehat{P}(c_1|e), \widehat{P}(c_2|e), \cdots, \widehat{P}(c_k|e)$
1. For each class label $c_p$ ($p = 1, 2, \ldots, k$)
   (a) For each $TAN_o$ ($o = 1, 2, \ldots, m$), use $TAN_o$ to estimate the probability $\widehat{P}(c_p|e)_o$ that $e$ belongs to class $c_p$.
   (b) Average all of the probabilities $\widehat{P}(c_p|e) = \frac{1}{m}\sum_{o=1}^{m}\widehat{P}(c_p|e)_o$.
2. Return the estimated $\widehat{P}(c_1|e), \widehat{P}(c_2|e), \ldots, \widehat{P}(c_k|e)$.
```

Notice that ATAN does not need to build the complete undirected maximum weighted spanning tree and the choosing of the root attribute is prior to the process of building the complete directed maximum weighted spanning tree, which are totally different from the previous learning algorithm of TAN. Therefore, the complete undirected maximum weighted spanning trees corresponding to these direct trees may be different. This assures that the class probability estimation performance of the learned TAN classifiers may be different, since only those TAN classifiers with different undirected trees have different class probability estimation performance. Moreover, it exactly meets the need of the ensemble learning. The experimental results in Section 4 show that our ATAN significantly outperforms the original TAN indeed.

Similar to TAN, ATAN needs to generate a three-dimensional table of probability estimates for each attribute-value, conditioned by each other attribute-value and each class, which takes the time complexity of $O(nm^2)$. Then, ATAN needs to calculate the conditional mutual information between each pair of attributes, requiring consideration for each pair of attributes, every pairwise combination of their respective values in conjunction with each class value, which takes the time complexity of $O(km^2v^2)$, where v is the average number of values for all attributes, namely $v = \frac{1}{m}\sum_{j=1}^{m} n_j$. At last, ATAN needs to construct m directed maximum weighted spanning trees, which takes the time complexity of $O(m^3 \log m)$. In summary, the time complexity of training ATAN is $O(nm^2 + km^2v^2 + m^3 \log m)$, which is a little higher than that of TAN $O(nm^2 + km^2v^2 + m^2 \log m)$. Besides, its time complexity of estimating the probability $\widehat{P}(c|e)$ that a test instance e belongs to the class c is $O(km^2)$, which is m times of that of TAN $O(km)$.

In the ATAN agorithm depicted above, the class-membership probabilities produced by different TAN classifiers are directly averaged. In fact, ATAN can be further improved by the use of some weighted average learning methods and its advantage would be stronger. Namely, the equation $\widehat{P}(c_p|e) = \frac{1}{m}\sum_{o=1}^{m}\widehat{P}(c_p|e)_o$ in our algorithm can be improved by the equation $\widehat{P}(c_p|e) = \frac{1}{\sum w_o}\sum_{o=1}^{m} w_o \widehat{P}(c_p|e)_o$, where w_o is the weight of TAN_o. To our knowledge, of numerous proposals can be used to achieve this goal. For example, we can use the mutual information $I_{\widehat{P}_D}(A_o; C)$ between the root attribute A_o and the class attribute C to define the weight w_o [24]. The detailed Equation is:

$$I_{\widehat{P}_D}(A_o; C) = \sum_{a,c} \widehat{P}(a,c) \log \frac{\widehat{P}(a,c)}{\widehat{P}(a)\widehat{P}(c)}, \tag{7}$$

where a is one of the attribute values of the attribute variable A_o and c is one of the class values of the class variable C. We simply call the resulting model Weighted ATAN (WATAN). The experimental results in Section 4 validate the effectiveness of this weighting approach.

4. Experiments and results

The purpose of these experiments is to investigate the class probability estimation performance of TAN and the proposed ATAN algorithm. Therefore, we conduct our experiments to compare the class probability estimation performance of TAN with NB, AODE, and ATAN in terms of conditional log likelihood (CLL).

We ran our experiments on the whole 36 UCI datasets [25] published on the main web site of Weka platform [26], which represent a wide range of domains and data characteristics. The description of the 36 datasets is shown in Table 1. In our experiments, Missing values are replaced with the modes and means of the corresponding attribute values from the available data. For example, if the sex of someone is missing, it can be replaced by the mode (the value with the highest frequency) of the sexes of all the others. In the same way, if the age of someone is missing, it can be replaced by the mean of the ages of all the others. Numeric attribute values are discretized using the unsupervised ten-bin discretization implemented in Weka platform. Besides, we manually delete three useless attributes: the attribute "Hospital Number" in the dataset "colic.ORIG", the attribute "instance name" in the dataset "splice", and the attribute "animal" in the dataset "zoo".

In our experiments, the conditional log likelihood scores of each algorithm on each data set are obtained via 10 runs of 10-fold cross-validation. Runs with the various algorithms are carried out on the same training sets and evaluated on the same test sets. In particular, the cross-validation folds are the same for all the experiments on each dataset. Finally, we conducted a corrected paired two-tailed t-test with the $p = 0.05$ significance level [27] to compare TAN with NB, AODE, and ATAN.

Table 2 shows the CLL scores of each algorithm on each data set, and the symbols ∘ and • in the table respectively denote statistically significant improvement or degradation over TAN with a 95% confidence level. Besides, the $w/t/l$ values are summarized at the bottom of the tables. Each entry $w/t/l$ in the table means that NB, AODE, and ATAN win on w datasets, tie on t datasets, and lose

Table 2
Conditional log likelihood comparisons for TAN versus NB, AODE, and ATAN.

Dataset	TAN	NB	AODE	ATAN
Anneal	−11.03 ± 6.65	−14.22 ± 6.16	−8.59 ± 5.12	−10.73 ± 6.37
Anneal.ORIG	−24.00 ± 7.45	−23.58 ± 5.60	−22.53 ± 5.34	−23.64 ± 7.35 ○
Audiology	−82.89 ± 22.19	−65.91 ± 24.28 ○	−64.85 ± 23.80 ○	−82.21 ± 22.15 ○
Autos	−35.33 ± 17.07	−45.48 ± 18.07	−27.44 ± 12.54	−32.76 ± 16.26 ○
Balance-scale	−33.61 ± 2.39	−31.75 ± 1.51	○ −33.26 ± 1.60	−33.27 ± 2.18
Breast-cancer	−18.81 ± 3.62	−18.37 ± 4.49	−16.73 ± 3.37 ○	−18.09 ± 3.42 ○
Breast-w	−9.62 ± 5.51	−18.28 ± 14.16 ●	−7.43 ± 4.84 ○	−8.50 ± 4.77 ○
Colic	−24.49 ± 8.74	−30.63 ± 11.38 ●	−20.37 ± 6.92 ○	−23.71 ± 8.44 ○
Colic.ORIG	−35.23 ± 9.88	−21.24 ± 5.74 ○	−19.96 ± 4.71 ○	−33.38 ± 9.19 ○
Credit-a	−30.42 ± 6.84	−28.79 ± 8.10	−26.50 ± 6.94 ○	−29.43 ± 6.60 ○
Credit-g	−59.02 ± 7.91	−52.79 ± 6.35 ○	−51.48 ± 5.80 ○	−57.45 ± 7.54 ○
Diabetes	−40.87 ± 7.49	−40.78 ± 7.49	−38.28 ± 6.10	−39.93 ± 7.02 ○
glass	−27.23 ± 5.50	−24.08 ± 5.42 ○	−22.17 ± 4.67 ○	−25.88 ± 5.03○
Heart-c	−14.97 ± 5.54	−13.91 ± 6.71	−12.95 ± 5.64	−14.42 ± 5.30 ○
Heart-h	−14.07 ± 3.97	−13.49 ± 5.37	−12.71 ± 4.48	−13.64 ± 3.87 ○
Heart-statlog	−13.42 ± 4.15	−12.25 ± 4.96	−11.95 ± 4.60	−13.02 ± 4.07 ○
Hepatitis	−7.60 ± 4.31	−8.53 ± 5.98	−6.65 ± 4.29	−7.30 ± 4.08
Hypothyroid	−95.75 ± 13.40	−97.14 ± 13.29	−88.24 ± 11.14 ○	−94.66 ± 13.24 ○
Ionosphere	−19.01 ± 13.06	−34.79 ± 19.94 ●	−20.98 ± 13.71	−17.26 ± 12.33 ○
Iris	−2.44 ± 1.77	−2.56 ± 2.35	−2.45 ± 1.41	−2.56 ± 1.45
kr-vs-kp	−56.91 ± 7.74	−93.48 ± 7.65 ●	−78.14 ± 5.97 ●	−57.03 ± 7.56
Labor	−2.38 ± 2.91	−0.71 ± 0.99	−0.87 ± 1.12	−2.13 ± 2.57
Letter	−1283.36 ± 67.78	−2505.15 ± 98.42 ●	−956.97 ± 41.93 ○	−1211.10 ± 64.09 ○
Lymph	−7.38 ± 4.58	−6.22 ± 3.96	−5.67 ± 3.58	−6.94 ± 4.25
Mushroom	−0.33 ± 1.11	−105.77 ± 23.25 ●	−1.02 ± 1.27	−0.44 ± 1.04
Primary-tumor	−69.69 ± 8.40	−65.56 ± 8.27○	−64.07 ± 7.68 ○	−68.92 ± 8.22 ○
Segment	−48.82 ± 15.42	−124.32 ± 33.74●	−50.97 ± 12.86	−45.79 ± 14.36○
Sick	−29.05 ± 9.20	−46.05 ± 11.99 ●	−32.25 ± 8.94	−28.34 ± 9.07 ○
Sonar	−21.83 ± 9.74	−22.67 ± 11.47	−14.10 ± 7.40 ○	−19.91 ± 8.95 ○
Soybean	−9.35 ± 4.29	−26.25 ± 11.03●	−15.64 ± 6.57 ●	−9.14 ± 4.27
Splice	−49.24 ± 12.15	−46.53 ± 12.85	−38.50 ± 10.52 ○	−48.36 ± 11.88 ○
Vehicle	−60.31 ± 10.20	−172.12 ± 27.55●	−67.17 ± 9.77 ●	−57.82 ± 9.28 ○
Vote	−7.73 ± 5.12	−27.25 ± 13.85 ●	−7.51 ± 4.68	−7.56 ± 4.95
Vowel	−25.40 ± 7.55	−89.80 ± 11.38 ●	−30.84 ± 6.78 ●	−24.69 ± 7.01
Waveform-5000	−241.89 ± 21.64	−378.00 ± 32.64 ●	−174.56 ± 15.90 ○	−234.65 ± 20.93 ○
Zoo	−1.06 ± 1.28	−1.22 ± 1.06	−1.04 ± 0.93	−0.98 ± 1.11
w/t/l	–	6/17/13	14/18/4	24/12/0

○, ● statistically significant improvement or degradation.

on *l* datasets, compared to TAN. From our experimental results, we can see that:

1. In terms of CLL, TAN is also much better than NB. Compared to NB, TAN wins in 13 datasets and only losses in 6 datasets.
2. ATAN significantly outperforms TAN with 24 wins and surprisingly 0 loss. This advantage is much stronger than the comparison (14 wins and 4 losses) between AODE and TAN.

In our another group of experiments, we compare ATAN with the weighted ATAN (WATAN) and the single TAN with the highest CLL score (We denote it TAN-H for simplicity.). Besides, we include LWC4.4 [7], a recently published decision tree algorithm for class probability estimation, into our experiments. From the detailed compared results in Table 3, we can find that:

1. ATAN is a little worse than WATAN (the weighted ATAN) with 1 win and 7 losses. This result validates the effectiveness of the weighting approach.
2. ATAN significantly outperforms TAN-H (the single TAN with the highest CLL score) in 16 datasets and surprisingly loses in 0 dataset.
3. Compared to LWC4.4, ATAN wins in 9 datasets and losses in 11 datasets. Considering that LWC4.4 is a lazy learning algorithm, ATAN is overall more effective in real-world applications.

Finally, we also observe the classification performance of ATAN in terms of classification accuracy. Interestingly, ATAN almost ties the original TAN (1 win and 0 loss). Due to the limit of space, we have not presented the detailed experimental results here.

5. Conclusions

In many real-world applications, accurate class probability estimation is often required to make optimal decisions. In this paper, we experimentally investigate the class probability estimation performance of Tree Augmented Naive Bayes (TAN) in terms of conditional log likelihood (CLL) and find that its class probability estimation performance is also much better than Naive Bayes (NB).

In order to make its class probability estimation performance stronger, we present a new algorithm ATAN to improve the class probability estimation performance of TAN by averaging all of the spanning TAN classifiers. ATAN respectively chooses each attribute as the root of the tree to directly built a complete directed maximum weighted spanning tree and the class-membership probabilities produced by the resulting TAN classifiers are averaged. The experimental results on a large number of UCI datasets published on the main web site of Weka platform validate its effectiveness.

Table 3
Conditional log likelihood comparisons for ATAN versus WATAN, TAN-H, and LWC4.4.

Dataset	ATAN	WATAN	TAN-H	LWC4.4
Anneal	−10.73 ± 6.37	−10.66 ± 6.34	−11.81 ± 6.78 •	−6.22 ± 1.86 ○
Anneal.ORIG	−23.64 ± 7.35	−23.40 ± 7.44	−25.97 ± 8.14 •	−20.92 ± 4.09
Audiology	−82.21 ± 22.15	−81.34 ± 21.99 ○	−83.71 ± 22.28 •	−14.90 ± 3.52 ○
Autos	−32.76 ± 16.26	−32.61 ± 16.26 ○	−34.69 ± 16.98 •	−11.76 ± 2.46 ○
Balance-scale	−33.27 ± 2.18	−33.29 ± 2.18	−33.99 ± 2.35	−48.83 ± 5.15 •
Breast-cancer	−18.09 ± 3.42	−18.11 ± 3.51	−18.06 ± 3.67	−18.75 ± 3.02
Breast-w	−8.50 ± 4.77	−8.54 ± 4.79	−9.53 ± 5.33	−11.11 ± 3.45 •
Colic	−23.71 ± 8.44	−23.69 ± 8.46	−25.83 ± 9.22 •	−18.53 ± 4.70 ○
Colic.ORIG	−33.38 ± 9.19	−32.54 ± 9.03 ○	-35.35 ± 9.50 •	−17.84 ± 3.06 ○
Credit-a	−29.43 ± 6.60	−29.39 ± 6.65	−30.57 ± 6.99	−29.22 ± 5.01
Credit-g	−57.45 ± 7.54	−57.23 ± 7.57	−58.42 ± 7.89	−62.49 ± 5.98
Diabetes	−39.93 ± 7.02	−39.85 ± 7.06	−40.58 ± 7.46	−43.07 ± 4.96
Glass	−25.88 ± 5.03	−25.79 ± 5.04	−28.12 ± 5.81 •	−20.75 ± 2.87 ○
Heart-c	−14.42 ± 5.30	−14.42 ± 5.35	−14.80 ± 5.61	−15.89 ± 3.74
Heart-h	−13.64 ± 3.87	−13.51 ± 4.04	−14.24 ± 4.89	−14.86 ± 3.23
Heart-statlog	−13.02 ± 4.07	−12.98 ± 4.06	−13.66 ± 4.40	−14.10 ± 3.39
Hepatitis	−7.30 ± 4.08	−7.30 ± 4.08	−7.61 ± 4.34	−6.77 ± 2.50
Hypothyroid	−94.66 ± 13.24	−94.55 ± 13.38	−95.39 ± 13.01	−91.50 ± 6.20
Ionosphere	−17.26 ± 12.33	−17.15 ± 12.30	−18.69 ± 13.05	−10.77 ± 3.00
Iris	−2.56 ± 1.45	−2.38 ± 1.42 ○	−3.17 ± 1.65 •	−3.45 ± 1.42 •
kr-vs-kp	−57.03 ± 7.56	−57.48 ± 7.61	−62.56 ± 8.73 •	−7.31 ± 3.37 ○
Labor	−2.13 ± 2.57	−2.17 ± 2.62	−2.41 ± 2.96	−2.19 ± 1.26
Letter	−1211.10 ± 64.09	−1212.50 ± 64.10	−1340.02 ± 72.61 •	−982.61 ± 32.95 ○
Lymph	−6.94 ± 4.25	−7.00 ± 4.26	−7.31 ± 4.52	−7.48 ± 2.42
Mushroom	−0.44 ± 1.04	−0.47 ± 1.12	−2.49 ± 3.67	−1.47 ± 0.14 •
Primary-tumor	−68.92 ± 8.22	−68.84 ± 8.21	−69.71 ± 8.15	−49.44 ± 4.13 ○
Segment	−45.79 ± 14.36	−45.61 ± 14.39	−50.38 ± 16.58 •	−41.60 ± 6.34
Sick	−28.34 ± 9.07	−26.69 ± 9.16 ○	−29.69 ± 9.54 •	−20.79 ± 5.75 ○
Sonar	−19.91 ± 8.95	−19.97 ± 8.94	−21.43 ± 9.49 •	−12.01 ± 2.43 ○
Soybean	−9.14 ± 4.27	−9.11 ± 4.25	−9.93 ± 4.56	−17.26 ± 3.58 •
Splice	−48.36 ± 11.88	−47.72 ± 11.74 ○	−49.39 ± 12.20	−66.52 ± 8.41 •
Vehicle	−57.82 ± 9.28	−58.54 ± 9.45 •	−61.26 ± 10.72 •	−54.56 ± 4.90
Vote	−7.56 ± 4.95	−7.53 ± 4.89	−7.80 ± 4.96	−7.01 ± 3.85
Vowel	−24.69 ± 7.01	−24.41 ± 6.85	−27.41 ± 8.48 •	−65.70 ± 6.23 •
Waveform-5000	−234.65 ± 20.93	−232.83 ± 20.98 ○	−242.90 ± 22.73 •	−319.69 ± 14.12 •
Zoo	−0.98 ± 1.11	−0.97 ± 1.08	−1.19 ± 1.27	−2.63 ± 1.32 •
w/t/l	–	7/28/1	0/20/16	11/16/9

○, • statistically significant improvement or degradation.

Acknowledgment

We thank anonymous reviewers for their valuable comments and suggestions. The work was partially supported by the National Natural Science Foundation of China (No. 60905033 and No. 61075063), the Provincial Natural Science Foundation of Hubei (No. 2009CDB139), and the Fundamental Research Funds for the Central Universities (No. CUG090109).

References

[1] M. Saar-Tsechansky, F. Provost, Active sampling for class probability estimation and ranking, Machine Learning 54 (2) (2004) 153–178.

[2] L. Jiang, H. Zhang, Learning Naive Bayes for probability estimation by feature selection, in: Proceedings of the 19th Canadian Conference on Artificial Intelligence, CAI 2006, LNAI 4013, Springer Press, pp. 503–514.

[3] J. Hu, N. Zhong, Y. Shi, Developing mining-grid centric E-finance portals for risk management and decision making, International Journal of Pattern Recognition and Artificial Intelligence 21 (4) (2007) 639–658.

[4] F.J. Provost, P. Domingos, Tree induction for probability-based ranking, Machine Learning 52 (3) (2003) 199–215.

[5] C. Ling, R. Yan, Decision tree with better ranking, in: Proceedings of the Twentieth International Conference on Machine Learning, AAAI Press, 2003, pp. 480–487.

[6] L. Jiang, C. Li, Z. Cai, Learning decision tree for ranking, Knowledge and Information Systems 20 (1) (2009) 123–135.

[7] L. Jiang, C. Li, Z. Cai, Decision tree with better class probability estimation, International Journal of Pattern Recognition and Artificial Intelligence 23 (4) (2009) 745–763.

[8] D. Grossman, P. Domingos, Learning Bayesian network classifiers by maximizing conditional likelihood, in: Proceedings of the Twenty-First International Conference on Machine Learning, ICML, ACM Press, 2004, pp. 361–368.

[9] Y. Guo, R. Greiner, Discriminative Model Selection for Belief Net Structures, in: Proceedings of the Twentieth National Conference on Artificial Intelligence, AAAI Press, AAAI 2005, pp. 770–776.

[10] J. Su, H. Zhang, C.X. Ling, S. Matwin, Discriminative parameter learning for Bayesian networks, in: Proceedings of the Twenty-Fifth International Conference on Machine Learning, ACM Press, Helsinki, Finland, 2008, pp. 1016–1023.

[11] N. Friedman, N. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine Learning 29 (1997) 131–163.

[12] M.G. Madden, On the classification performance of TAN and general Bayesian networks, Knowledge-Based Systems 22 (7) (2009) 489–495.

[13] J. Pearl, Probabilistic Reasoning in Intelligent Systems, Morgan Kaufmann, San Francisco, CA, 1988.

[14] D.M. Chickering, Learning Bayesian networks is NP-Complete, in: D. Fisher, H. Lenz (Eds.), Learning from Data: Artificial Intelligence and Statistics, V, Springer, Press, 1996, pp. 121–130.

[15] L. Jiang, H. Zhang, Z. Cai, A Novel Bayes model: hidden Naive Bayes, IEEE Transactions on Knowledge and Data Engineering 21 (10) (2009) 1361–1371.

[16] A.A. Balamurugan, R. Rajaram, S. Pramala, et al., NB+: An improved Naive Bayesian algorithm, Knowledge-Based Systems 24 (5) (2011) 563–569.

[17] J. Xiao, C. He, X. Jiang, Structure identification of Bayesian classifiers based on GMDH, Knowledge-Based Systems 22 (6) (2009) 461–470.

[18] E. Keogh, M. Pazzani, Learning augmented Bayesian classifiers: A comparison of distribution-based and classification-based approaches. In: Proceedings of the International Workshop on Artificial Intelligence and Statistics, 1999, pp. 225–230.

[19] H. Zhang, C.X. Ling, An improved learning algorithm for augmented Naive Bayes, in: Proceedings of the Fifth Pacific-Asia Conference on KDD, Springer Press, PAKDD 2001, pp.581–586.

[20] C.X. Ling, H. Zhang, Toward Bayesian classifiers with accurate probabilities, in: Proceedings of the Sixth Pacific-Asia Conference on KDD, Springer Press, PAKDD 2002, pp.123–134.

[21] A.P. Bradley, The use of the area under the ROC curve in the evaluation of machine learning algorithms, Pattern Recognition 30 (1997) 1145–1159.

[22] D.J. Hand, R.J. Till, A simple generalisation of the area under the ROC curve for multiple class classification problems, Machine Learning 45 (2) (2001) 171–186.

[23] G.I. Webb, J. Boughton, Z. Wang, Not so Naive Bayes: aggregating one-dependence estimators, Machine Learning 58 (2005) 5–24.
[24] L. Jiang, H. Zhang, Weightily averaged one-dependence estimators, in: Proceedings of the 9th Biennial Pacific Rim International Conference on Artificial Intelligence, PRICAI 2006, LNAI 4099, Springer Press, pp.970–974.
[25] C. Merz, P. Murphy, D. Aha, UCI repository of machine learning databases, in Dept of ICS, University of California, Irvine, 1997. <http://www.ics.uci.edu/mlearn/MLRepository.html>.
[26] I.H. Witten, E. Frank, Data Mining: Practical machine learning tools and techniques, second ed., Morgan Kaufmann, San Francisco, 2005. <http://prdownloads.sourceforge.net/weka/datasets-UCI.jar>.
[27] C. Nadeau, Y. Bengio, Inference for the generalization error, Machine Learning 52 (2003) 239–281.