Learning Tree Augmented Naive Bayes for Ranking

Liangxiao Jiang[1] *, Harry Zhang[2], Zhihua Cai[1], and Jiang Su[2]

[1] Department of Computer Science, China University of Geosciences
Wuhan, China 430074
[2] Faculty of Computer Science, University of New Brunswick
P.O. Box 4400, Fredericton, NB, Canada E3B 5A3

Abstract. Naive Bayes has been widely used in data mining as a simple and effective classification algorithm. Since its conditional independence assumption is rarely true, numerous algorithms have been proposed to improve naive Bayes, among which tree augmented naive Bayes (TAN) [3] achieves a significant improvement in term of classification accuracy, while maintaining efficiency and model simplicity. In many real-world data mining applications, however, an accurate ranking is more desirable than a classification. Thus it is interesting whether TAN also achieves significant improvement in term of ranking, measured by AUC(the area under the Receiver Operating Characteristics curve) [8, 1]. Unfortunately, our experiments show that TAN performs even worse than naive Bayes in ranking. Responding to this fact, we present a novel learning algorithm, called forest augmented naive Bayes (FAN), by modifying the traditional TAN learning algorithm. We experimentally test our algorithm on all the 36 data sets recommended by Weka [12], and compare it to naive Bayes, SBC [6], TAN [3], and C4.4 [10], in terms of AUC. The experimental results show that our algorithm outperforms all the other algorithms significantly in yielding accurate rankings. Our work provides an effective and efficient data mining algorithm for applications in which an accurate ranking is required.

Keywords: data mining and knowledge discovery, learning algorithms, Bayesian networks, decision trees.

1 Introduction

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

* This work was done when the author was a visiting scholar at University of New Brunswick.

In many data mining applications, however, the classifier's accuracy are not enough, because they cannot express the information how "far-off" (be it 0.45 or 0.01?) is the prediction of each example from its target. For example, in direct marketing, we often need to promote the top X% of customers during gradual roll-out, or we often deploy different promotion strategies to customers with different likelihood of buying some products. To accomplish these tasks, we need more than a mere classification of buyers and non-buyers. We often need a ranking of customers in terms of their likelihood of buying. Thus, a ranking is more desirable than just a classification.

A natural question is how to evaluate a classifier in terms of its ranking performance, rather than classification accuracy. Recently, the area under the Receiver Operating Characteristics curve [8, 1], or simply AUC, has been used for this purpose and received a considerable attention. AUC compares the classifiers' performance cross the entire range of class distributions and error costs and is a good "summary" for comparing two classifiers. Hand and Till [4] show that, for binary classification, AUC is equivalent to the probability that a randomly chosen example of class $-$ will have a smaller estimated probability of belonging to class $+$ than a randomly chosen example of class $+$. They present a simple approach to calculating the AUC of a classifier G below.

$$\hat{A} = \frac{S_0 - n_0(n_0+1)/2}{n_0 n_1}, \tag{1}$$

where n_0 and n_1 are the numbers of negative and positive examples respectively, and $S_0 = \sum r_i$, where r_i is the rank of i_{th} positive example in the ranked list. From Equation 1, it is clear that AUC is essentially a measure of the quality of a ranking. For example, the AUC of a ranking is 1 (the maximum value of AUC) if there is no positive example preceding a negative example.

In classification, an example $E = (a_1, a_2, \cdots, a_n)$, where a_i is the value of attribute A_i, is classified into the class C with the maximum posterior class probability $P(C|E)$ (or simply, class probability), as shown below.

$$C_{pb}(E) = \arg\max_{C} P(C|E). \tag{2}$$

Assume that all the attributes are independent given the value of class, called conditional independence assumption and shown in Equation 3. The resulting classifier, called naive Bayes, is shown in Equation 4. Figure 1 shows an example of naive Bayes.

$$P(a_1, \cdots, a_n|C) = \prod_{i=1}^{n} P(a_i|C). \tag{3}$$

$$C_{nb}(E) = \arg\max_{C} P(C) \prod_{i=1}^{n} P(a_i|C). \tag{4}$$

The structure of naive Bayes can be extended to represent the dependences among attributes. Tree Augmented naive Bayes (TAN) is an extended tree-like

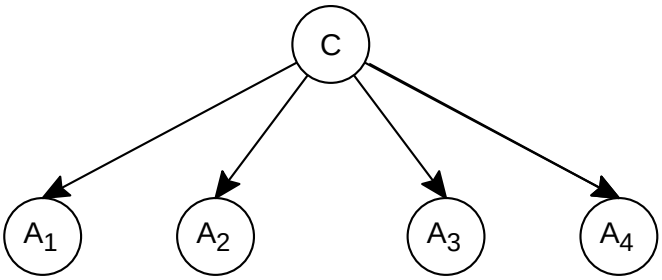

Fig. 1. An example of NB

naive Bayes [3], in which the class node directly points to all attribute nodes and an attribute node can have only one parent from another attribute node (in addition to the class node). Figure 2 shows an example of TAN. In TAN, each node has at most two parents (one is the class node). TAN outperforms naive Bayes in terms of accuracy [3] and still maintains a considerably simple structure.

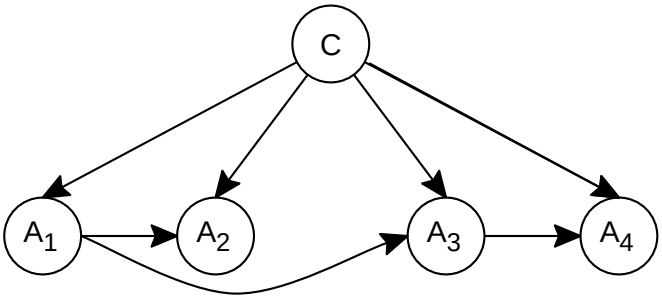

Fig. 2. An example of TAN

One interesting question is whether TAN is also a good model for ranking. In this paper, we investigate the ranking performance of TAN. Unfortunately, the traditional TAN learning algorithm does not produce high quality ranking. We propose a new TAN learning algorithm and our experiments show that our algorithm performs better not only than the traditional TAN learning algorithm, but also other popular state-of-the-art algorithms designed for yielding accurate ranking.

The rest of the paper is organized as follows. In Section 2, we introduce the related work on improving naive Bayes and on improving decision tree for ranking. In Section 3, we present our new algorithm. In Section 4, we describe the experimental setup and results in detail. In Section 5, we make a conclusion.

2 Related Work

It is obvious that the conditional independence assumption in naive Bayes is rarely true in many applications. Therefore, researchers have made a substantial amount of effort to improve naive Bayes in classification. Research work to improve the naive Bayes can be broadly divided into two approaches below.

1. Select attributes subsets in which attributes are conditionally independent. For example, Langley and Sage [6] presented an algorithm, called Selective Bayesian Classifiers (simply SBC), to improve naive Bayes. They used a forward greedy search method to select a subset of attributes.
2. Relax the conditional independence assumption by extending the structure of naive Bayes to represent the dependences among attributes. TAN is an example of this approach. TAN is a specific case of general augmented naive Bayesian networks (ANB), in which the class node also directly points to all attribute nodes, but there is no limitation on the links among attribute nodes (except that they do not form any directed cycle).

Unfortunately, learning an optimal ANB is intractable. Thus, TAN is a good trade-off between the model complexity and learnability in practice. A number of TAN learning algorithms have been proposed, among which the ChowLiu algorithm (CL-TAN) [3] and the SuperParent algorithm (SP-TAN) [5] performs significantly better than naive Bayes in classification. SP-TAN is a greedy heuristic search algorithm in which an arc of achieving the highest accuracy improvement is selected in each step. One disadvantage of SP-TAN is its time complexity of $O(mn^3)$, where m is the number of training examples and n is the number of attributes. However, CL-TAN has the time complexity of $O(mn^2)$, a considerable advantage over SP-TAN. CL-TAN is depicted below, which is the base of our work.

Algorithm CL-TAN

1. Compute $I_{\hat{P}_D}(A_i, A_j|C)$ between each pair of attributes, $i \neq j$.
2. Build a complete undirected graph in which nodes are attributes $A_1, \cdots, A_n$. Annotate the weight of an edge connecting A_i to A_j by $I_{\hat{P}_D}(A_i; A_j|C)$.
3. Build a maximum weighted spanning tree.
4. Transform the resulting undirected tree to a directed one by choosing a root attribute and setting the direction of all edges to be outward from it.
5. Construct a TAN model by adding a node labeled by C and adding an arc from C to each A_i.

In the preceding algorithm, $I_{\hat{P}_D}(A_i, A_j|C)$ is an estimate of the conditional mutual information which will be defined in Section 3.

Both SP-TAN and CL-TAN outperforms naive Bayes significantly in classification. Moreover, the ranking performance of SP-TAN has been studied [14]. Since CL-TAN is more efficient than SP-TAN, it is more practical in data mining applications. In this paper, we focus on the ranking performance of CL-TAN.

Decision tree learning algorithms are a major type of effective learning algorithms in data mining. However, traditional decision tree algorithms, such as C4.5 [11], have been observed to produce poor estimations of probabilities [10]. Aiming at this fact, Provost and Domingos [10] presented an algorithm, called C4.4, to improve C4.5's performance in ranking measured by AUC. In detail, they used two techniques to improve the AUC of C4.5: smooth probability estimates by Laplace correction and turn off pruning. Their experiments show that C4.4 performs significantly better than C4.5 in ranking. In this paper, we compare our new algorithm with C4.4.

3 Forest Augmented Naive Bayes:FAN

At first, let us introduce the definitions of mutual information and conditional mutual information used in this paper.

Definition 1. *Let X,Y are two variables,then the mutual information between X and Y is defined by the following equation [3].*

$$I_P(X;Y) = \sum_{x,y} P(x,y) \log \frac{P(x,y)}{P(x)P(y)}. \tag{5}$$

Roughly speaking, this function measures how much information Y provides about X.

Definition 2. *Let X,Y,Z are three variables,then the conditional mutual information between X and Y given Z is defined by the following equation [3].*

$$I_P(X;Y|Z) = \sum_{x,y,z} P(x,y,z) \log \frac{P(x,y,z)P(z)}{P(x,z)P(y,z)}. \tag{6}$$

Roughly speaking, this function measures the information that Y provides about X when the value of Z is known.

In a TAN, the class probability $P(C|E)$ is estimated by the following equation:

$$P(C|E) = P(C) \prod_{i=1}^{n} P(A_i|A_{ip}, C) \tag{7}$$

where A_{ip} is the parent of A_i and

$$P(A_i|A_{ip}, C) = \begin{cases} P(A_i|A_{ip}, C) & \{A_{ip}\} \neq \emptyset \\ P(A_i|C) & \{A_{ip}\} = \emptyset \end{cases} \tag{8}$$

An instance is classified into the class with the maximum class probability.

We experimentally investigate the ranking performance of CL-TAN, measured by AUC. Unfortunately, CL-TAN yields poor AUC (see Table 1 and 2, although its accuracy is higher than naive Bayes (see Table 3 and 4). By experiments, we observe that there are two factors contributing this fact:

1. The directions of edges in a TAN are crucial. In Step 4 of the CL-TAN algorithm, an attribute is randomly chosen as the root of the tree and the directions of all edges are set outward from it. Notice that the selection of the root attribute actually determines the structure of the resulting TAN, since a TAN is a directed graph. It is interesting that the directions of edges in a TAN do not affect the classification accuracy significantly. In contrast, however, AUC is quite sensitive to it. Thus the selection of the root attribute is important for building a TAN with accurate ranking.
2. Irrelevant edges may exist in a CL-TAN. In Step 3 of the CL-TAN, a maximum weighted spanning tree is built. Thus, the number of the edges is fixed to $n-1$. Sometimes, it might overfit the data, since some edges may not be necessary to exist in the TAN.

Based on the preceding observations, we modify the CL-TAN algorithm correspondingly as follows.

1. We choose the attribute A_{root} with the maximum mutual information with class, defined by Equation 1, as the root. That is,

$$A_{root} = \arg\max_{A_i} I_P(A_i; C), \tag{9}$$

   where $i = 1, \cdots, n$. It is natural to use this strategy, since intuitively the attribute which has the greatest influence on classification should be the root of the tree.
2. We filter out the edges that have a conditional mutual information less than a threshold. To our understanding, those edges have a high risk to overfit the training data, and thus undermine the probability estimation. More precisely, we use the average conditional mutual information I_{avg}, defined in Equation 10, as the threshold. All the edges with the conditional mutual information less than I_{avg} are removed.

$$I_{avg} = \frac{\sum_i \sum_{j,j \neq i} I_P(A_i; A_j|C)}{n(n-1)}, \tag{10}$$

   where n is the number of attributes.

Since the structure of the resulting model is not a strict tree, we call our algorithm forest augmented naive Bayes (FAN), depicted in detail as follows.

Algorithm FAN

1. Calculate the conditional mutual information $I_P(A_i; A_j|C), j \neq i$ between each pair of attributes, and calculate the average conditional mutual information I_{avg}, defined in Equation 10.

2. Build a complete undirected graph in which nodes are attributes $A_i, i = 1, 2, \ldots, n$. Annotate the weight of an edge connecting A_j to A_i by $I_P(A_i; A_j|C)$.
3. Search a maximum weighted spanning tree.
4. Calculate the mutual information $I_P(A_i; C)$, $i = 1, 2, \ldots, n$ between each attribute and the class, and find the attribute A_{root} that has the maximum mutual information with class, according to Equation 9.
5. Transform the resulting undirected tree to a directed one by setting A_{root} as the root and setting the directions of all edges to be outward from it.
6. Delete the directed edges with the weight of the conditional mutual information below the average conditional mutual information I_{avg}.
7. Construct a FAN model by adding a vertex labeled by C and adding an directed arc from C to each $A_i, i = 1, 2, \ldots, n$.

The time complexity and space complexity of FAN are $O(n^2 \cdot N)$ and $O(|C|(n|V|)^2)$, respectively, where n is the number of attributes, N is the number of training instances, $|C|$ is the number of classes, and $|V|$ is the average number of values for an attribute. Both of them are same as the CL-TAN algorithm. However, our experiments, described in next section (Section 4) that FAN improves the ranking performance of CL-TAN significantly.

4 Experimental Methodology And Results

We conduct our experiments on all the 36 data sets recommended by Weka [13], which come from the UCI repository [7]. We download these data sets in format of arff from main web of Weka. All the preprocessing stages of data sets were carried out by the Weka system. They mainly include the following three processes:

1. We use the filter of ReplaceMissingValues in Weka to replace the missing values of attributes.
2. We use the filter of Discretize in Weka to discretize numeric attributes.
3. It is well-known that, if the number of values of an attribute is almost equal to the number of instances in the data set, this attribute does not contribute any information to classification. So we use the filter of Remove in Weka to delete these attributes. In these 36 data sets, there only exists three this type of attributes, namely Hospital Number in colic.ORIG, Instance Name in Splice and Animal in zoo.

We conduct experiments to compare our algorithm (FAN) with naive Bayes, SBC [6], TAN [3], and C4.4 [10] in AUC . All algorithms are implemented within the Weka framework. Multi-class AUC has been calculated by M-measure in [4]. The AUC of each classifier is measured via the ten-fold cross validation for all data sets. Runs with the various classifiers were 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 data set. Throughout, we compare

our algorithm with each other algorithm via two-tailed t-test with significantly different probability of 0.95, because we speak of two results for a data set as being "significantly different" only if the difference is statistically significant at the 0.05 level according to the corrected two-tailed t-test.

Table 1 shows the AUC and standard deviations of each classifier on the test sets of each data set, and the average AUC and deviation are summarized at the bottom of the table. Table 2 shows the results of two-tailed t-test between each pair of algorithms, and each entry $w/t/l$ means that the algorithm at the corresponding row wins in w data sets, ties in t data sets, and loses in l data sets, compared to the algorithm at the corresponding column.

The detailed results displayed in Table 1 and Table 2 show that our algorithm outperforms significantly all the other algorithms in AUC. Now, we summarize the highlights as follows:

1. FAN outperforms naive Bayes significantly: It wins in 9 data sets, ties in 27 data sets and loses in 0 data set. The average AUC for FAN is 89.95%, it is slightly higher than the average AUC 89.61% of naive Bayes. This fact is understandable, since the conditional independence among attributes have been relaxed and represented in FAN. Thus, the class probability estimates of FAN are expected to be more accurate than those of naive Bayes.
2. FAN also outperforms C4.4 significantly: It wins in 13 data sets, ties in 20 data sets and loses in 3 data sets. The average AUC for C4.4 is 85.92%, lower than that of FAN. Since C4.4 is the state-of-the-art decision tree algorithm designed specifically for yielding accurate rankings, this comparison also provides evidence to support FAN.
3. FAN outperforms significantly SBC and TAN. It wins in 11 data sets, ties in 25 data sets and loses in 0 data set, compared with SBC; and it wins in 24 data sets, ties in 12 data sets and loses in 0 data set, compared with TAN. Notice that, although SBC and TAN improve naive Bayes' performance in classification, they do not improve naive Bayes' performance in ranking.

In our experiments, we also observe the classification accuracy of each algorithm, shown in Table 3, and Table 4 shows the results of two-tailed t-test with confidence level of 95% between each pair of algorithms in terms of accuracy. We can see that our experiment repeats experimental results of SBC [6] and CL-TAN [3], both of which improve the classification performance of naive Bayes. It is also interesting to notice that FAN also slightly outperforms all the algorithms in terms of accuracy.

5 Conclusions

In this paper, we investigate the ranking performance of the CL-TAN learning algorithm, and find that CL-TAN performs even worse than naive Bayes in ranking. Responding to this problem, we present a novel TAN learning algorithm FAN to build a TAN for accurate ranking. We experimentally test our algorithm measured by AUC, using all the 36 data sets recommended by Weka, and compare

Table 1. Experimental results on AUC. FAN: Forest Augmented naive Bayes; NB: naive Bayes; SBC: Selective Bayesian Classifiers; CL-TAN : Tree Augmented naive Bayes with smoothed parameter of 5.0; C4.4: C4.5 with Laplace correction and without tree pruning.

Data set	FAN	NB	SBC	CL-TAN	C4.4
anneal	96.4±0.51	95.9±1.3	94.7±3.92	92.97±2.51	93.78±2.9
anneal.ORIG	95.1±2.93	94.49±3.67	94.35±4.31	85.42±7.04	92.69±3.15
audiology	70.92±0.59	70.96±0.73	70.98±0.67	70.16±0.55	70.58±0.63
autos	92.13±5.24	89.18±4.93	90.43±3.43	90.28±2.59	90.73±4.52
balance-scale	84.46±4.1	84.46±4.1	84.46±4.1	76.47±7.56	63.06±6.18
breast-cancer	68.04±12.43	69.71±15.21	67.67±12.63	67.4±10.4	59.3±12.03
breast-w	99.15±0.94	99.19±0.87	99.16±0.62	98.74±1.32	97.85±1.86
colic	85.25±6.16	83.71±5.5	84.86±7.13	50.6±8.29	85.02±7.03
colic.ORIG	74.91±9.77	80.67±6.98	81.82±4.9	62.89±7.73	80.56±8.94
credit-a	91.3±3.36	92.09±3.43	87±3.75	63.3±13.3	89.42±3.1
credit-g	78.25±6.42	79.27±4.74	77.41±4.67	60.18±6.84	69.62±5
diabetes	82.71±5.65	82.31±5.17	82.79±5.04	74.18±5.87	75.5±5.76
glass	79.03±7.02	80.5±6.65	80.97±8.37	84.79±4.34	82.36±4.38
heart-c	83.95±0.71	84.1±0.54	83.87±0.64	82.96±1.12	83.1±1.19
heart-h	83.66±0.8	83.8±0.7	82.83±1.38	82.69±0.72	83.04±0.85
heart-statlog	90.42±5.36	91.3±4.19	87.98±6.91	80.12±11.94	81.36±9.15
hepatitis	85.91±11.52	88.99±8.99	83.62±12.29	53.83±14.97	82.03±14.04
hypothyroid	86.69±9.61	87.37±8.52	85.25±8.16	84.03±12.22	81.58±8.8
ionosphere	98.48±1.47	93.61±3.36	92.26±5.26	72.05±7.4	93.1±3.76
iris	98.58±2.67	98.58±2.67	99±1.46	94.17±5.51	97.33±2.63
kr-vs-kp	98.12±0.9	95.17±1.29	96.41±0.78	87.21±1.49	99.95±0.06
labor	93.33±14.05	98.33±5.27	65.83±32.5	68.33±40.41	74.17±31.04
letter	98.28±0.19	96.86±0.24	97.03±0.23	94.5±0.25	95.39±0.39
lymph	89.95±1.57	89.69±1.49	88.14±3.35	85.56±6.98	87.26±3.75
mushroom	100±0	99.79±0.04	99.98±0.02	99.87±0.04	100±0
primary-tumor	78.9±1.03	78.85±1.35	78.88±1.45	76.39±1.9	75.48±2.33
segment	99.55±0.27	98.51±0.46	98.93±0.42	95.35±1.06	98.85±0.32
sick	98.22±0.77	95.91±2.35	94.5±4.28	73.25±2.73	99.07±0.35
sonar	85.96±10.19	85.48±10.82	79.89±13.1	67.4±13.83	77.01±8.59
soybean	99.61±0.64	99.53±0.6	99.08±0.74	96.73±1.59	91.43±2.6
splice	99.47±0.32	99.41±0.22	99.14±0.36	97.72±0.68	98.14±0.72
vehicle	89.05±2.99	80.81±3.51	81.31±4.02	76.86±3.8	86.5±2.28
vote	98.03±1.51	96.56±2.09	94.26±4.14	93.49±1.38	96.77±2.96
vowel	99.51±0.26	95.81±0.84	96.12±0.59	92.33±1.23	91.28±2.46
waveform-5000	94.92±0.63	95.27±0.58	95.12±0.76	78.9±2.03	80.83±1.24
zoo	89.88±4.05	89.88±4.05	89.06±4.49	89.88±4.05	88.88±4.5
Mean	89.95±3.795	89.61±3.54	87.92±4.746	80.58±5.991	85.92±4.708

Table 2. Results of two-tailed t-test on AUC. An entry $w/t/l$ means that the algorithm at the corresponding row wins in w data sets, ties in t data sets, and loses in l data sets, compared to the algorithm at the corresponding column. The significantly different probability of two-tailed t-test is **0.95**.

	NB	SBC	CL-TAN	C4.4
FAN	9/27/0	11/25/0	24/12/0	13/20/3
NB	-	1/31/4	23/12/1	13/20/4
SNB	-	-	20/16/0	9/22/5
CL-TAN	-	-	-	4/20/12

our algorithm FAN with naive Bayes, SBC, TAN, and C4.4. The experimental results show that our algorithm improves significantly naive Bayes' performance in ranking, and outperforms some widely used extended naive Bayes algorithms, such as SBC and CL-TAN and the state-of-the-art decision tree learning algorithm C4.4. In a word, our work provides an effective and efficient data mining algorithm especially when a ranking is more desirable than just a classification.

References

1. Bradley, A. P.: The use of the area under the ROC curve in the evaluation of machine learning algorithms. Pattern Recognition **30** (1997) 1145-1159
2. Cohen, W. W., Schapire, R. E., Singer, Y.: Learning to order things. Journal of Artificial Intelligence Research, **10** 1997 243-270
3. Friedman, N., Greiger, D., Goldszmidt, M.: Bayesian Network Classifiers. Machine Learning **29** (1997) 103–130
4. Hand, D. J., Till, R. J.: A simple generalisation of the area under the ROC curve for multiple class classification problems. Machine Learning **45** (2001) 171-186
5. Keogh, E., Pazzani, M. : Learning augmented bayesian classifiers. Proceedings of Seventh International Workshop on AI and Statistics. (1999) Ft. Lauderdale.
6. Langley, P., Sage, S.: Induction of selective Bayesian classifiers. in Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, 1994, pp. 339-406.
7. Merz, C., Murphy, P., Aha, D.: UCI repository of machine learning databases. Dept of ICS, University of California, Irvine (1997). http://www.ics.uci.edu/ mlearn/MLRepository.html
8. Provost, F., Fawcett, T.: Analysis and visualization of classifier performance: comparison under imprecise class and cost distribution. Proceedings of the Third International Conference on Knowledge Discovery and Data Mining. AAAI Press (1997) 43-48
9. Provost, F., Fawcett, T., Kohavi, R.: The case against accuracy estimation for comparing induction algorithms. Proceedings of the Fifteenth International Conference on Machine Learning. Morgan Kaufmann (1998) 445-453
10. Provost, F. J., Domingos, P.: Tree Induction for Probability-Based Ranking. Machine Learning **52(3)** (2003) 199-215
11. Quinlan, J. R.: C4.5: Programs for Machine Learning. Morgan Kaufmann: San Mateo, CA (1993)
12. http://prdownloads.sourceforge.net/weka/datasets-UCI.jar

Table 3. Experimental results on accuracy. FAN: Forest Augmented naive Bayes; NB: naive Bayes; SBC: Selective Bayesian Classifiers; CL-TAN: Tree Augmented naive Bayes with smoothed parameter of 5.0; C4.4: C4.5 with Laplace correction and without post pruning.

Data set	FAN	NB	SBC	CL-TAN	C4.4
anneal	97.1±1.5	94.32±2.38	96.88±2.5	96.66±2.35	99±0.98
anneal.ORIG	90.98±3.64	87.53±4.69	88.75±3.72	87.98±3.62	91.76±3.07
audiology	71.19±5.14	71.23±7.03	76.01±7.05	75.16±8.45	78.3±8
autos	77.55±8.11	64.83±11.18	67.71±11.27	76.07±10.01	81.45±7.48
balance-scale	91.36±1.38	91.36±1.38	91.36±1.38	86.08±3.18	69.3±4.25
breast-cancer	68.21±5.11	72.06±7.97	73.45±8.91	66.82±7.01	68.57±7.49
breast-w	97.13±2.03	97.28±1.84	96.42±2.26	96.71±1.79	92.99±3.66
colic	81.25±5.31	78.81±5.05	81.77±4.89	77.18±7.04	80.17±5.95
colic.ORIG	72.57±6.5	75.26±5.26	75.53±6.15	75.51±7.15	76.08±8.74
credit-a	84.49±3.99	84.78±4.28	85.51±4.16	84.64±5.03	83.19±3.5
credit-g	75.6±5.15	76.3±4.76	74.1±3.87	73.4±4.12	68.6±4.3
diabetes	75.4±6.61	75.4±5.85	75.53±5.07	75.13±4.71	69.54±5.12
glass	59.87±7.98	60.32±9.69	57.99±6.89	55.71±10.81	58.83±7.73
heart-c	81.13±7.8	84.14±4.16	82.47±7.61	77.53±7.41	74.26±11.46
heart-h	82±5.94	84.05±6.69	79±9.77	79.97±6.39	72.78±11
heart-statlog	82.59±6.77	83.7±5	79.26±9.75	81.11±3.68	75.93±8.95
hepatitis	83.17±9.66	83.79±8.79	80.63±6.8	83.83±8.05	81.25±11.52
hypothyroid	93.19±0.78	92.79±1.02	93.53±0.66	92.79±1.06	92.5±0.58
ionosphere	92.61±4.64	90.89±3.49	91.17±4.12	90.6±3.83	84.63±4.45
iris	94.67±8.2	94.67±8.2	97.33±4.66	90.67±11.42	92.67±5.84
kr-vs-kp	92.52±2.09	87.89±1.81	94.34±1.23	93.18±1.6	99.41±0.45
labor	88.33±15.81	93.33±11.65	77±11.91	88±11.46	77.67±15.64
letter	76.77±0.78	70±0.81	70.57±0.88	80.45±0.91	80.56±0.87
lymph	83.14±7.22	85.67±9.55	79±6.84	84.38±9.1	74.29±12.56
mushroom	99.4±0.27	95.57±0.45	99.67±0.23	99.77±0.12	100±0
primary-tumor	46.31±2.33	46.89±4.32	46.02±5.19	48.37±5.83	38.91±4.97
segment	94.37±1.59	88.92±1.95	90.43±1.96	86.36±2.36	92.86±1.39
sick	97.67±0.47	96.74±0.53	97.59±0.69	97±0.4	97.83±0.61
sonar	78.5±16	77.5±11.99	70.71±12.97	71.62±12.64	67.69±10.94
soybean	95.76±1.61	92.08±2.34	91.79±2.72	93.41±2.1	92.68±1.56
splice	95.3±1.48	95.36±1	94.76±1.6	95.39±1.35	91.57±1.37
vehicle	69.98±3.29	61.82±3.54	60.65±4.73	69.86±3.47	69.03±2.63
vote	92.66±4.65	90.14±4.17	95.18±3.93	93.12±4.02	94.96±3.83
vowel	92.42±2.2	67.07±4.21	68.69±3.47	83.43±3.84	75.66±5.18
waveform-5000	82±1.24	79.96±1.92	81.32±1.54	81.52±1.21	64.86±1.83
zoo	97.09±4.69	94.18±6.6	93.18±7.93	97.09±4.69	92.18±8.94

Table 4. Results of two-tailed t-test on accuracy. the results of two-tailed t-test between each pair of algorithms, each entry $w/t/l$ means that the algorithm at the corresponding row wins in w data sets, ties in t data sets, and loses in l data sets, compared to the algorithm at the corresponding column. The significantly different probability of two-tailed t-test is **0.95**.

	NB	SBC	CL-TAN	C4.4
FAN	10/26/0	5/29/2	4/30/2	12/21/3
NB	-	1/29/6	4/26/6	11/15/10
SNB	-	-	4/28/4	7/22/7
CL-TAN	-	-	-	7/15/4

13. Witten, I. H., Frank, E.: Data Mining –Practical Machine Learning Tools and Techniques with Java Implementation. Morgan Kaufmann (2000)
14. Ling, C. X., Zhang, H.: Toward Bayesian classifiers with accurate probabilities. Proceedings of the Sixth Pacific-Asia Conference on KDD. Springer (2002) 123-134