Pre-publication draft of a paper awhich appeared in the Proceedings of AI'03 LNAI 2903 pp 453 - 465 Publisher: Springer

# Adjusting Dependence Relations for Semi-Lazy *TAN* Classifiers

Zhihai Wang[1], Geoffrey I. Webb[1], Fei Zheng[2]

[1] School of Computer Science and Software Engineering,
Monash University, Clayton Campus,
Clayton, Victoria, 3800, Australia
{zhihai.wang, webb}@infotech.monash.edu.au
[2] Department of Computer Science, Nanchang University,
Jiangxi Province, 330029, China
zhengfei_z@163.com

**Abstract.** The naive Bayesian classifier is a simple and effective classification method, which assumes a Bayesian network in which each attribute has the class label as its only one parent. But this assumption is not obviously hold in many real world domains. Tree-Augmented Naive Bayes ($TAN$) is a state-of-the-art extension of the naive Bayes, which can express partial dependence relations among attributes. In this paper, we analyze the implementations of two different $TAN$ classifiers and their tree structures. Experiments show how different dependence relations impact on accuracy of $TAN$ classifiers. We present a kind of semi-lazy $TAN$ classifier, which builds a $TAN$ identical to the original $TAN$ at training time, but adjusts the dependence relations for a new test instance at classification time. Our extensive experimental results show that this kind of semi-lazy classifier delivers lower error than the original $TAN$ and is more efficient than Superparent $TAN$.

## 1 Introduction

Classification learning seeks to build a classifier that can assign a suitable class label to an unlabelled instance described by a set of attributes. The naive Bayesian classifier is widely used in interactive applications due to its computational efficiency, competitive accuracy, direct theoretical base, and its ability to integrate prior information with data sample information [1–7]. It is based on Bayes' theorem and an assumption that all attributes are mutually independent within each class. Assume $X$ is a finite set of instances, and $A = \{A_1, A_2, \cdots, A_n\}$ is a finite set of $n$ attributes. An instance $x \in X$ is described by a vector $\langle a_1, a_2, \cdots, a_n \rangle$, where $a_i$ is a value of attribute $A_i$. $C$ is called the class attribute. Prediction accuracy will be maximized if the predicted class

$$Label(\langle a_1, a_2, \cdots, a_n \rangle) = argmax_c(P(c \mid \langle a_1, a_2, \cdots, a_n \rangle)). \quad (1)$$

Unfortunately, unless $\langle a_1, a_2, \cdots, a_n \rangle$ occurs many times within $X$, it will not be possible to directly estimate $P(c \mid \langle a_1, a_2, \cdots, a_n \rangle)$ from the frequency with

which each class $c \in C$ co-occurs with $\langle a_1, a_2, \cdots, a_n \rangle$ within a given set of training instances. Bayes' theorem provides an equality that might be used to help estimate the posterior probability $P(c_i \mid x)$ in such a circumstance:

$$P(c_i \mid x) = \frac{P(c_i)P(\langle a_1, a_2, \cdots, a_n \rangle \mid c_i)}{P(\langle a_1, a_2, \cdots, a_n \rangle)} \tag{2}$$

$$= \alpha \cdot P(c_i) \cdot P(\langle a_1, a_2, \cdots, a_n \rangle \mid c_i) \tag{3}$$

where $P(c_i)$ is the prior probability of class $c_i \in C$, $P(\langle a_1, a_2, \cdots, a_n \rangle \mid c_i)$ is the conditional probability of $x \in T$ given the class $c_i$, and $\alpha$ is a normalization factor. According to the Bayes Theorem and the chain rule, equation 3 can be written as:

$$P(c_i \mid x) = \alpha \cdot P(c_i) \cdot \prod_{k=1}^{n} P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i) \tag{4}$$

Therefore, an approach to Bayesian estimation is to seek to estimate each $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$.

If the $n$ attributes are mutually independent within each class value, then the conditional probability can be calculated in the following way:

$$P(\langle a_1, a_2, \cdots, a_n \rangle \mid c_i) = \prod_{k=1}^{n} P(a_k \mid c_i). \tag{5}$$

Classification selecting the most probable class as estimation using formula 3 and formula 5 is the well-known naive Bayesian classifier.

The attribute independence assumption makes the application of Bayes' theorem to classification practical in many domains, but this assumption rarely holds in real world problems. Where some dependence relations do exist among attributes, the probability estimate of the naive Bayesian classifier may be incorrect. In such circumstances, comparing equation 4 with equation 5, we cannot use $P(a_k \mid c_i)$ instead of $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$, where $k = 1, 2, \cdots, n$. Notwithstanding Domingos and Pazzani analysis that demonstrates that some violations of the independence assumption are not harmful to classification accuracy [1], it is clear that many are, and there is an increasing body of work developing techniques to retain naive Bayesian classifiers' desirable simplicity and efficiency while delivering improved accuracy [2–5, 8–12].

Of numerous proposals to improve the accuracy of naive Bayesian classifiers by weakening its attribute independence assumption, Tree Augmented Naive Bayes($TAN$) [9, 3, 4] has demonstrated remarkable error performance [7]. Friedman, Geiger and Goldszmidt [9, 3] compared the naive Bayesian method and Bayesian network, and showed that using unrestricted Bayesian networks did not generally lead to improvements in accuracy and even reduced accuracies in some domains . They presented a compromise representation, called tree-augmented naive Bayes ($TAN$, called Friedman's $TAN$ in our paper), in which the class node directly points to all attribute nodes and an attribute node can

have only at most one additional parent to the class node. Based on this representation, they utilized a scoring measurement, called conditional mutual information, to efficiently find a maximum weighted spanning tree as a classifier. Keogh & Pazzani [4] took a different approach to constructing tree-augmented Bayesian networks(called Keogh and Pazzani's $TAN$ in our paper). They used the same representation, but used leave-one-out cross validation to estimate the classification accuracy of the network when an arc was added. The two methods mainly differ in two aspects. One is the criterion of attribute selection used to select dependence relations among the attributes while building a tree-augmented Bayesian network. Another is the structure of the classifiers. The first one always tends to construct a tree including all attributes, the second one always tends to construct a tree with fewer dependence relations among attributes and better classification accuracy.

Friedman's $TAN$ and Keogh & Pazzani's $TAN$ are eager classifiers. They build tree-augmented Bayesian classifiers based on a given set of training instances at training time, and classify a new unlabelled instance directly using the classifiers at classification time.We analyze these two different approaches to $TAN$ classifiers and their tree classifier structures. We show experimentally how different dependence relations impact on the accuracy of $TAN$ classifiers. As a result of this study we present a new semi-lazy $TAN$ classification algorithm. At training time, it builds a $TAN$ identical to Friedman's $TAN$, but at classification time we adjust the dependence relations for each new test instance. Different Bayesian networks may apply to different unlabelled instances. Therefore, this approach can be thought of as a semi-lazy or partially-lazy classifier. Our extensive experimental results have shown that this kind of semi-lazy classifier has better accuracy than the previous $TAN$ classifiers.

## 2 Restricted Bayesian Network Classifiers

Bayesian network classification is a probability classification method that can describe probability distributions over the training data. However, learning unrestricted Bayesian networks is very time consuming and quickly becomes intractable as the number of attributes increases [3, 13]. Previous research also shows that some scoring metrics used in learning unsupervised Bayesian networks do not necessarily optimize the performance of the learned networks in classification [9, 3]. Therefore, restricting the structure of Bayesian networks has become an active research area. The naive Bayesian classifier can be regarded as a highly restricted Bayesian network, which assumes that each attribute has the class label as its only one interdependent variable. TAN classifiers allow each attribute to depend on the class and at most one additional attribute. In this section, we will more formally describe $TAN$ classifiers and show some issues in the implementations of the $TAN$ classifiers.

### 2.1 The Basic *TAN* classifiers

A basic $TAN$ classifier, i.e. a Friedman's $TAN$ classifier, is a restricted Bayesian network classification model, which uses a tree-structure imposed on the naive Bayesian structure. In its Bayesian network, the class node is the root and has no parents, i.e. $\Pi(C) = \emptyset$ (here $\Pi(A_i)$ represents the set of parents of variable or attribute $A_i$). The class variable is a parent of each attribute variables, i. e. $C \in \Pi(A_i)$. And except for the class node, each attribute variable node has at most one other attribute variable node as its a parent, i.e. $|\Pi(A_i)| \leq 2$. Therefore $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$ in equation 4 can be simplified as follows.

$$P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i) = P(a_k \mid \pi(a_k)). \tag{6}$$

Note that $\Pi(A_i)$ represents the set of parents of attribute $A_i$. $\pi(a_i)$ represents the set of parents of attribute value $a_i$. An example of Bayesian network structure for a $TAN$ structure is shown in Figure 1, where class variable node $C$ and all dependences from it to all attribute nodes $A_i$ are omitted for simplicity.

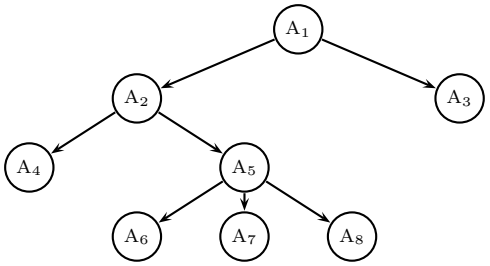


**Fig. 1.** A $TAN$ classifier's tree structure

The algorithm for building a basic $TAN$ classifier consists of five main steps [3]:

1. Compute conditional mutual information $I_T(A_i, A_j \mid C)$ between each pair of attributes as follows($i \neq j$):

$$I_T(A_i, A_j \mid C) = \sum_{A_i, A_j, C} P(A_i, A_j \mid C) log \frac{P(A_i, A_j \mid C)}{P(A_i \mid C) \cdot P(A_j \mid C)} \tag{7}$$

2. Build a complete undirected graph in which the vertices are attributes $A_1, \cdots, A_n$. Annotate the weight of an edge connecting $A_i$ to $A_j$ by $I_T(A_i, A_j \mid C)$.

3. Build a maximum weighted spanning tree.

4. Transform the resulting undirected tree to a directed one by choosing a root variable and setting the direction of all edges to be outward from it.

5. Build a $TAN$ model by adding a vertex labelled by $C$ and adding an arc from $C$ to each $A_i$.

**Table 1.** Descriptions of Data

| Domain | ♯Instances | ♯Classes | ♯ Attributes |
|---|---|---|---|
| 1 Adult | 48842 | 2 | 14 |
| 2 Annealing Processes | 898 | 6 | 38 |
| 3 Breast Cancer(Wisconsin) | 699 | 2 | 9 |
| 4 Credit Screening(Australia) | 690 | 2 | 15 |
| 5 German | 1000 | 2 | 20 |
| 6 Glass Identification | 214 | 7 | 10 |
| 7 Heart Disease(Cleveland) | 303 | 2 | 13 |
| 8 Hepatitis Prognosis | 155 | 2 | 19 |
| 9 House Votes 84 | 435 | 2 | 16 |
| 10 Hypothyroid Diagnosis | 3163 | 2 | 25 |
| 11 Iris Classification | 150 | 3 | 4 |
| 12 LED 24(noise level=10%) | 1000 | 10 | 24 |
| 13 Letter Recognition | 20000 | 26 | 16 |
| 14 Liver Disorders(bupa) | 345 | 2 | 6 |
| 15 Lung Cancer | 32 | 3 | 56 |
| 16 New-Thyroid | 215 | 3 | 5 |
| 17 Pen Digits | 10992 | 10 | 16 |
| 18 Pima Indians Diabetes | 768 | 2 | 8 |
| 19 Pioneer | 9150 | 57 | 36 |
| 20 Post-Operative Patient | 90 | 3 | 8 |
| 21 Promoter Gene Sequences | 106 | 2 | 57 |
| 22 Segment | 2310 | 7 | 19 |
| 23 Solar Flare | 1389 | 3 | 10 |
| 24 Sonar Classification | 208 | 2 | 60 |
| 25 Soybean Large | 683 | 19 | 35 |
| 26 Splice Junction Gene Sequences | 3177 | 3 | 60 |
| 27 Syncon | 600 | 6 | 60 |
| 28 Tic-Tac-Toe End Game | 958 | 2 | 9 |
| 29 Vehicle | 846 | 4 | 18 |
| 30 Zoology | 101 | 7 | 16 |

### 2.2 Some Issues in the Implementation

All the experiments in this paper are performed in the *Weka* system [14]. Now, we discuss some extended issues in our implementation of the basic $TAN$ classifier.

One issue is related to the probability estimation assumption. In $TAN$, for each attribute we assess the conditional probability given the class variable and another attribute. This means that the number of instances used to estimate the conditional probability is reduced as it is estimated from the instances that share three specific values (the class value, the parent value and the child value). Thus it is not surprising to encounter unreliable estimates, especially in small datasets. Friedman, Geiger and Goldszmidt dealt with this problem by introducing a

smoothing operation [3]. The estimation formula used by them is as follows.

$$\theta^s(x \mid \pi(x)) = \frac{N \cdot \widehat{P}_T P(\pi(x))}{N \cdot \widehat{P}_T P(\pi(x)) + N^0} \cdot \widehat{P}_T(x \mid \pi(x) + \frac{N^0}{N \cdot \widehat{P}_T P(\pi(x)) + N^0} \cdot \theta^0(x \mid \pi(x)) \tag{8}$$

where $\theta^0(x \mid \pi(x))$ is the prior estimate of $P(x \mid \pi(x))$, and $N^0$ is the confidence associated with that prior. In their experiments, $N^0 = 5$. A problem arises when an attribute value does not occur in the training data, a situation often occurs in cross validation tests. In this case the value of the estimate will be zero. To address this problem, in our implementation, we also apply a normal Laplace estimation to $\widehat{P}_T P(x)$. We use both smoothing functions to estimate any conditional probability, and only Laplace estimation to estimate a non-conditional probability.

Secondly, regarding the problem of missing values, in $TAN$ classifiers, instances with missing values were deleted from the set of training instances by Friedman, et al. We keep all the instances, but ignore missing values from the counts for attribute values. Also, when we estimate a conditional probability $P(a_k \mid c_i)$, for a prior probability of class value $c_i$ we exclude the occurrences of class value $c_i$ with missing values on attribute $A_k$. Obviously, this makes the estimation of the condition more reliable while estimating $P(a_k \mid c_i)$.

Thirdly, although the choice of root variable does not change the log-likelihood of the $TAN$ network, we have to set the direction of all edges for classification. When each edge $(Ai, Aj)$ is added to the current tree structure, we always set the direction from $Ai$ to $Aj$ $(i < j)$ at once.

### 2.3 Keogh and Pazzani's *TAN* Classifiers

Keogh and Pazzani present another approach to learn $TAN$ classifiers [4], called $SuperParent$,which searches heuristically for a $TAN$ guided by cross-validation accuracy. They show that their algorithm consistently predicts more accurately than naive Bayes. It consists of two steps. The first step searches for a super parent that has the best cross-validation accuracy. A super parent is a node with arcs pointing to all others nodes without a parent except for the class label. The second step determines one favorite child for the super parent chosen at the first step, again based on the cross-validation accuracy. After this iteration of the two steps, one arc is added on the tree structure, and this process repeats until no improvement is achieved, or $n - 1$ arcs are added into the tree.

We also implemented Keogh and Pazzani's $TAN$ classifiers in $Weka$ system. They follow Friedman and Goldszmidt's assumption about missing values [3], i.e., instances with missing values were deleted from the datasets. In their experiments, they replace zero probabilities with a small epsilon (0.0001). However, for consistency, our implementation utilises the smoothing techniques described above for our implementation of the basic $TAN$ classifiers.

## 3 Adjusting Dependence Relations in Semi-Lazy Way

In this section, we discuss how to select dependence relations among attribute values given a test instance based on the basic $TAN$ network. Experimentally we demonstrate that the tree structure is a useful description for the given set of training instances, and on the other hand, most of these conditional probabilities have extremely high variance and lead to poor predictions. We investigate a method of adjusting the dependence relation for a given conditional probability. At training time, we derive the basic $TAN$ classifiers as Friedman's $TAN$. At classification time, we reinterpret the dependence relations for a given unlabelled instance. As the interpretation is done at classification time, we can regard this kind of classification model as a semi-lazy or partially-lazy classifier. Finally, we also experimentally demonstrate that building a $TAN$ classifier in a totally lazy way is not effective. Before describing the new algorithms, we first describe the data sets used in our experiments and our experimental methodology.

### 3.1 Experimental Domains and Methodology

The thirty natural domains used in our experiments are shown in Table 1 [15]. All the experiments were performed in the Weka system [14], which provides a workbench that includes full and working implementations of many popular learning schemes that can be used for practical data mining or for research. The error rate of each classification model on each domain is determined by running 10-fold cross validation on a dual-processor 1.7Ghz Pentium 4 Linux computer with 2Gb RAM. All the data sets were used in previous research [9, 4, 5]. We also use the default discretization method "weka.filters.DiscretizeFilter" to discretize continuous attributes, which is based on Fayyad and Irani's method [16].

### 3.2 Applying Higher-Order Conditional Probabilities

A $TAN$ structure is a kind of restricted Bayesian network, which combines some of Bayesian networks' ability to represent dependence relations with the simplicity of naive Bayes. A $TAN$ structure means the way of using $P(a_k \mid c_i)$ *or* $P(a_k \mid \pi(a_k))$ instead of $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$ in equation 4. It is clear that in some situations, it would be useful to model correlations among attributes that cannot be captured by a $TAN$ structure. This will be significant when there is a sufficient number of training instances to robustly estimate higher-order conditional probabilities [9]. However, estimating higher-order conditional probabilities will cost much more computation. Our alternative is to seek to find a better estimate instead of $P(a_k \mid \pi(a_k))$ as the estimate of $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$ based on a known $TAN$ structure. Each node in a $TAN$ structure always has the strongest mutual information with its parent. The performance of estimation for node $A_k$ depends on the estimate for its parent node. This suggests using all the ancestors as the condition of node $A_k$. The result is an algorithm for classification based on the following equation:

$$P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i) = P(a_k \mid Ancestors(a_k)). \tag{9}$$

**Table 2.** Applying Higher-Order Conditional Probabilities

| Domain | $TAN^sT_1$ | $TAN^sT_2$ | $HOCPT_1$ | $HOCPT_2$ |
|---|---|---|---|---|
| 1 Adult | 15.96 | 16.31 | 14.70 | 16.27 |
| 2 Annealing Processes | 3.90 | 4.34 | 3.79 | 4.14 |
| 3 Breast Cancer(Wisconsin) | 0.86 | 3.58 | 0.00 | 6.01 |
| 4 Credit Screening(Australia) | 11.88 | 14.20 | 4.35 | 24.64 |
| 5 German | 16.90 | 27.70 | 1.90 | 27.20 |
| 6 Glass Identification | 0.93 | 9.34 | 0.93 | 7.94 |
| 7 Heart Disease(Cleveland) | 10.56 | 18.48 | 4.29 | 24.09 |
| 8 Hepatitis Prognosis | 3.87 | 18.71 | 1.29 | 21.94 |
| 9 House Votes 84 | 5.29 | 7.13 | 1.38 | 5.98 |
| 10 Hypothyroid Diagnosis | 2.09 | 2.53 | 1.58 | 2.81 |
| 11 Iris Classification | 2.67 | 10.00 | 2.00 | 12.67 |
| 12 LED 24(noise level=10%) | 24.80 | 26.30 | 24.30 | 26.30 |
| 13 Letter Recognition | 15.86 | 19.35 | 4.26 | 20.66 |
| 14 Liver Disorders(bupa) | 20.29 | 39.71 | 6.09 | 40.58 |
| 15 Lung Cancer | 9.37 | 46.88 | 0.00 | 43.75 |
| 16 New-Thyroid | 2.33 | 6.98 | 1.40 | 10.23 |
| 17 Pen Digits | 3.34 | 5.06 | 0.32 | 16.59 |
| 18 Pima Indians Diabetes | 13.93 | 25.78 | 8.46 | 29.43 |
| 19 Pioneer | 2.96 | 4.71 | 2.91 | 4.79 |
| 20 Post-Operative Patient | 23.33 | 41.11 | 16.67 | 35.56 |
| 21 Promoter Gene Sequences | 0.00 | 18.86 | 0.00 | 34.91 |
| 22 Segment | 11.68 | 13.20 | 1.13 | 10.22 |
| 23 Solar Flare | 0.94 | 1.01 | 0.86 | 0.94 |
| 24 Sonar Classification | 2.40 | 29.81 | 0.00 | 44.71 |
| 25 Soybean Large | 5.27 | 8.49 | 3.22 | 12.59 |
| 26 Splice Junc. Gene Sequences | 2.80 | 4.60 | 0.03 | 41.33 |
| 27 Syncon | 0.00 | 5.17 | 0.00 | 64.33 |
| 28 Tic-Tac-Toe End Game | 22.23 | 26.10 | 21.82 | 26.20 |
| 29 Vehicle | 23.52 | 32.39 | 1.30 | 37.35 |
| 30 Zoology | 0.00 | 6.93 | 1.98 | 5.94 |

Note that this implies that the attribute subscripts are ordered so that $\forall a_j \in Ancestors(a_k)$, $j < k$, an ordering that need only be imposed implicitly. In Table 2, we list the experimental results of our implementation of Friedman's $TAN$ and the classifier based on above formula 9. $TAN^sT_1$ represents the results of the basic $TAN$ algorithm classifying all the training instances. $TAN^sT_2$ represents the results of the basic $TAN$ algorithm using 10-fold cross validations. $HOCPT_1$ represents the results of applying higher-order conditional probabilities formula shown in equation 9 to classify all the training instances. $HOCPT_2$ represents corresponding results using 10-fold cross validations. The results are surprising. Most of the results of new estimation using 10-fold cross validations are worse than the $TAN$'s, but most of the results of new estimation classifying the training instances are better than the $TAN$'s. That tells us the new algorithm is overfitting.

### 3.3 Adjusting Dependence Relations Algorithm

Finding the dependence relations among the attributes is an important way to relax the attribute independent assumption made by naive Bayes. The main difference among Bayesian classification models of this kind is in way they calculate $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$. The above results experimentally show there is some possibility to find another attribute value instead of the value of the parent attribute. If $P(a_k \mid \pi(a_k))$ is a poor choice for $P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i)$, we should first try to use $P(a_k \mid \pi(\pi(a_k)))$, because this attribute has the strongest dependence relations with its parent attribute. We use the following equation for adjusting the original dependence relation in the $TAN$ structure.

$$P(a_k \mid a_1, a_2, \cdots, a_{k-1}, c_i) = P(a_k \mid MAX\{Ancestors(a_k)\}, c_i). \qquad (10)$$

where $MAX\{Ancestors(a_k)\}$ represents the attribute value of its ancestors which has the maximum conditional mutual information with attribute value $a_k$. In this case, the conditional mutual information between two attribute values can be calculated as follows:

$$I_T(a_k, a_j \mid c) = \sum_c P(a_k, a_j \mid c) log \frac{P(a_k, a_j \mid c)}{P(a_k \mid c) \cdot P(a_j \mid c)} \qquad (11)$$

because we are interested only in the specific values, no the full range of values for each attribute. At training time, we still build a $TAN$ model in the same way, but at classification time, we will use formula 10 to classify an unlabelled instance. This is a semi-lazy classifier. Our algorithm also tests the tree structure. When a $TAN$ structure is a single chain, we always use naive Bayes directly.

We compare the classification performance of four learning algorithms by running 10-fold cross validations. In the Table 3, we list the experimental results. We use the naive Bayes classifier implemented in the $Weka$ system, simply called $Naive$. We implemented in Weka Friedman's smoothed $TAN$, called $TAN^s$, Keogh and Pazzani's $TAN$, called $SP$, and our semi-lazy $S - Lazy$. The mean accuracy and running time over all data sets for each algorithm is also given in Table 3. Table 4 presents the WIN/LOSS/DRAW records for the semi-lazy $TAN$ model together with the result of a binomial sign test which indicates the probabilities of obtaining the observed result or more extreme if WINS and LOSSES were equiprobable. This is a record of the number of data sets for which the nominated algorithm achieves lower, higher, and equal error to the comparison algorithm, measured to two decimal places. The semi-lazy $TAN$ demonstrates significantly better classification performance than the original $TAN$ models, and worse(albeit not significantly) than Keogh and Pazzani's $TAN$ models, but is much more efficient than Keogh and Pazzani's $TAN$ models.

### 3.4 Building $TAN$ structures in Totally-Lazy Ways

Previous experimental results have shown that, in most cases, adjusting dependence relations can improve the performance of the basic $TAN$ classifiers. Can

**Table 3.** Experimental Results of Comparing Algorithms

| | Naive | $TAN^s$ | | S-Lazy | | SP | |
|---|---|---|---|---|---|---|---|
| Domain | Error | Error | Time | Error | Time | Error | Time |
| Adult | 18.03 | 16.31 | 1.05 | 16.23 | 1.02 | 15.77 | 178.59 |
| Annealing Processes | 5.46 | 4.34 | 0.12 | 4.57 | 0.19 | 4.01 | 1.49 |
| Breast Cancer(Wisconsin) | 2.58 | 3.58 | 0.13 | 2.86 | 0.02 | 2.58 | 0.13 |
| Credit Screening(Australia) | 15.07 | 14.20 | 0.03 | 15.22 | 0.04 | 14.35 | 0.98 |
| German | 24.60 | 27.70 | 0.06 | 25.70 | 0.06 | 24.80 | 3.14 |
| Glass Identification | 11.68 | 9.35 | 0.03 | 7.01 | 0.06 | 6.07 | 0.10 |
| Heart Disease(Cleveland) | 16.50 | 18.48 | 0.01 | 17.49 | 0.02 | 15.19 | 0.23 |
| Hepatitis Prognosis | 16.13 | 18.71 | 0.02 | 11.61 | 0.02 | 16.13 | 0.11 |
| House Votes 84 | 9.89 | 7.13 | 0.01 | 9.66 | 0.01 | 6.90 | 0.65 |
| Hypothyroid Diagnosis | 2.94 | 2.53 | 0.13 | 2.53 | 0.13 | 2.88 | 15.51 |
| Iris Classification | 6.00 | 10.00 | 0.01 | 6.00 | 0.01 | 6.00 | 0.00 |
| LED 24(noise level=10%) | 26.20 | 26.30 | 0.01 | 26.10 | 0.01 | 25.90 | 0.23 |
| Letter Recognition | 29.99 | 19.35 | 1.23 | 23.15 | 1.77 | 16.47 | 464.11 |
| Liver Disorders(bupa) | 36.81 | 39.71 | 0.01 | 39.13 | 0.01 | 40.29 | 0.05 |
| Lung Cancer | 46.88 | 46.88 | 0.11 | 46.88 | 0.18 | 50.00 | 1.18 |
| New-Thyroid | 8.37 | 6.98 | 0.00 | 6.05 | 0.02 | 7.44 | 0.02 |
| Pen Digits | 12.92 | 5.06 | 0.53 | 6.03 | 0.69 | 3.50 | 105.77 |
| Pima Indians Diabetes | 25.00 | 25.78 | 0.01 | 25.26 | 0.02 | 25.39 | 0.21 |
| Pioneer | 9.77 | 4.71 | 4.44 | 5.42 | 8.91 | 3.66 | 1256.99 |
| Post-Operative Patient | 28.89 | 41.11 | 0.00 | 33.33 | 0.00 | 30.00 | 0.03 |
| Promoter Gene Sequences | 8.49 | 18.87 | 0.15 | 14.15 | 0.19 | 8.49 | 7.35 |
| Segment | 11.08 | 13.20 | 0.20 | 9.26 | 0.31 | 6.28 | 17.06 |
| Solar Flare | 3.89 | 1.01 | 0.02 | 0.86 | 0.02 | 1.01 | 0.49 |
| Sonar Classification | 25.48 | 29.81 | 0.53 | 25.00 | 0.81 | 23.56 | 21.59 |
| Soybean Large | 7.17 | 8.49 | 0.18 | 7.76 | 0.27 | 7.03 | 6.24 |
| Splice Junc. Gene Sequences | 4.66 | 4.60 | 1.40 | 4.60 | 1.32 | 4.69 | 217.70 |
| Syncon | 3.00 | 5.17 | 1.61 | 3.00 | 2.44 | 3.00 | 64.71 |
| Tic-Tac-Toe End Game | 29.54 | 26.10 | 0.01 | 24.95 | 0.01 | 28.81 | 0.71 |
| Vehicle | 39.48 | 32.39 | 0.10 | 35.22 | 0.15 | 31.68 | 11.24 |
| Zoology | 5.94 | 6.93 | 0.00 | 4.95 | 0.01 | 5.94 | 0.05 |
| The Mean | 16.41 | 16.49 | 0.40 | 15.33 | 0.62 | 14.59 | 79.22 |

**Table 4.** Comparison of $Semi-LazyTAN$ to others

| | $WIN$ | $LOSS$ | $DRAW$ | $P$ |
|---|---|---|---|---|
| $Naive$ | 18 | 9 | 3 | 0.122 |
| $TAN^s$ | 20 | 7 | 3 | 0.019 |
| $SP$ | 10 | 18 | 2 | 0.184 |

we get lower error using a totally-lazy way? Can we build a better $TAN$ structure for a given test instance? For many classification tasks classifier accuracy is more important than consideration of computational expense. In such a circumstance, building a classifier in a lazy way may be a better choice. To evaluate the promise of truly lazy $TAN$, we also implemented two ways for building a $TAN$ structure using the measurement of conditional mutual information between attribute values. One is based on a given class value, another is based on all class values. Neither of them reduces error. Our previous research [7] also showed the implementation of Keogh and Pazzani's $TAN$ in a lazy way did not improve classification performance. Probably, there should be some different measurement to show dependence relations among attribute values.

## 4 Conclusions

There are several contributions in this paper. The first one is that we have examined and implemented two different $TAN$ classifiers and their tree classifier structures. Secondly, we experimentally show how different dependence relations impact on the accuracy of $TAN$ classifiers. Thirdly, we mainly present a semi-lazy $TAN$ classification model, which builds the same tree structure as the basic $TAN$ model at training time, but adjusts the dependence relations for a new test instance at classification time. This approach can be thought of as a semi-lazy or partially-lazy method. Our extensive experimental results have shown that these semi-lazy classifiers have higher accuracy than the original $TAN$ and are more efficient than Keogh and Pazzani's $TAN$.

It is remarkable that all our research is based on the assumption that the conditional mutual information can really reflect dependence relations among attributes. Because the measurements of conditional mutual information between attributes do not specify the direction of the dependence, this is also a reason that we can improve classification performances by adjusting dependence relations among attribute values. Although the semi-lazy $TAN$ demonstrates better classification performance than the original $TAN$ models, it is worse than Keogh and Pazzani's $TAN$ models. These results may suggest a way to better restrict dependence relations based on the $TAN$ structure.

## References


1. P. Domingos and M. Pazzani.: Beyond Independence: Conditions for the Optimality of the Simple Bayesian Classifier. In: Proceedings of the Thirteenth International Conference on Machine Learning, Morgan Kaufmann Publishers, Inc., San Francisco, CA(1996)105–112,
2. Kohavi, R.: Scaling up the Accuracy of Naive-Bayes Classifiers: A Decision-Tree Hybird, In: Simoudis, E.,Han, J. W., Fayyad, U. M.(eds.): Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, The AAAI Press, Menlo Park, CA(1996)202-207
3. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian Network Classifiers. Machine Learning, Kluwer Academic Publishers, Boston, 29 (1997) 131–163

4. Keogh, E. J., Pazzani, M. J.: Learning Augmented Bayesian Classifiers: A Comparison of Distribution-Based and Classification-Based Approaches. In: Proceedings of the Seventh International Workshop on Artificial Intelligence and Statistics. (1999) 225-230
5. Zheng, Z., Webb, G. I.: Lazy Learning of Bayesian Rules. Machine Learning, Kluwer Academic Publishers, Boston, 41(2000) 53-84
6. Webb, G. I., Boughton, J., Wang, Z: Averaged One-Dependence Estimators: Preliminary Results. In: S. J. Simoff, G. J. Williams and M. Hegland (Eds.). Proceedings of Australian Data Mining Workshop(ADM 2002), Canberra, Australia: University of Technology Sydney, (2002) 65-73
7. Wang. Z., Webb, G. I.: Comparison of Lazy Bayesian Rule and Tree-Augmented Bayesian Learning. In: Kumar, V., Tsumoto, S., Zhong, N., Yu, P. S., Wu, X. (eds.): Proceedings of 2002 IEEE International Conference on Data Mining, IEEE Computer Society, Los Alamitos, CA(2002) 490-497
8. Kononenko, I.: Semi-Naive Bayesian Classifier. In: Proceedings of European Conference on Artificial Intelligence, (1991) 206-219
9. Friedman, N., Goldszmidt, M.: Building Classifiers Using Bayesian Networks. In: Proceedings of the Thirteenth National Conference on Artificial Intelligence, The AAAI Press, Menlo Park, CA, 1996 (1277-1284)
10. Langley, P., Sage, S.: Induction of Selective Bayesian Classifiers. In: Proceedings of the Tenth Conference on Uncertainty in Artificial Intelligence, Morgan Kaufmann Publishers, Seattle, WA, 1994 (339–406)
11. Sahami, M.: Learning Limited Dependence Bayesian Classifiers. In: Proceedings of the Second International Conference on Knowledge Discovery and Data Mining, AAAI Press, Portland, OR, 1996 (335–338)
12. Webb, G. I., Pazzani, M. J.: Adjusted Probability Naive Bayesian Induction. In: Proceedings of the Eleventh Australian Joint Conference on Artificial Intelligence, Springer-verlag, Brisbane, 1998 (285–295)
13. Chickering D. M.: Learning Bayesian Networks is NP-Hard. Technical Report MSR-TR-94-17, Microsoft Research Advanced Technology Division, Microsoft Corporation, 1994
14. Witten, I. H., Frank, E.: Data Mining: Practical Machine Learning Tools and Techniques with Java Implementations. Seattle, WA: Morgan Kaufmann Publishers. (2000)
15. Merz, C., Murphy, P., Aha, D.: UCI Repository of Machine Learning Databases. Department of Information and Computer Science, University of California, Irvine. http://www.ics.uci.edu/mlearn/ MLRepository.html
16. Fayyad, U., Irani, K.: Multi-Interval Discretization of Continuous-valued Attributes for Classification Learning. In: Proceedings of the 13th International Joint Conference on Artificial Intelligence, Seattle, WA: Morgan Kaufmann Publishers, (1993), 1022-1027