IEEE Access
Multidisciplinary ⋮ Rapid Review ⋮ Open Access Journal

Received January 6, 2020, accepted January 18, 2020, date of publication February 5, 2020, date of current version February 14, 2020.

Digital Object Identifier 10.1109/ACCESS.2020.2971706

Self-Adaptive Attribute Value Weighting for Averaged One-Dependence Estimators

LIMIN WANG[1,2], JIE CHEN[1], YANG LIU[1], AND MINGHUI SUN[1,2]
[1]College of Computer Science and Technology, Jilin University, Changchun 130012, China
[2]Key Laboratory of Symbolic Computation and Knowledge Engineering of Ministry of Education, Jilin University, Changchun 130012, China

Corresponding author: Minghui Sun (smh@jlu.edu.cn)

This work was supported in part by the National Natural Science Foundation of China under Grant 61272209 and Grant 61872164.

ABSTRACT Of numerous proposals for weakening the attribute independence assumption of Naive Bayes, averaged one-dependence estimators (AODE) learns by extrapolation from marginal to full-multivariate probability distributions, and has demonstrated reasonable improvement in terms of classification performance. However, all the one-dependence estimators in AODE are assigned with the same weight, and their probability estimates are combined linearly. This work presents an efficient and effective attribute value weighting approach that assigns discriminative weights to different super-parent one-dependence estimators for different instances by identifying the differences among these one-dependence estimators in terms of log likelihood. The proposed approach is validated on widely used benchmark datasets from UCI machine learning repository. Experimental results show that the proposed approach achieves bias-variance trade-off and is a competitive alternative to state-of-the-art Bayesian and non-Bayesian learners (e.g., tree augmented Naive Bayes and logistic regression).

INDEX TERMS Attribute value weighting, averaged one-dependence estimators, log likelihood, entropy.

I. INTRODUCTION

Bayesian network (BN) [1]–[4] provides a powerful tool for knowledge representation and inference under conditions of uncertainty. Since the 1990s, the study of Bayesian network classifier (BNC) for classification has attracted tremendous attention after the success of Naive Bayes (NB) [5]–[8]. To relax the unrealistic attribute independence assumption of NB, researchers proposed to learn the conditional dependencies among attributes. The addition of augmented edges to the topology of NB resulted in BNCs, such as tree-augmented NB (TAN) [9] and k-dependence Bayesian classifier (KDB) [10], that achieved significant advantage over NB in terms of classification performance while retaining the simplicity and efficiency. To avoid the intractable computational complexity for learning BNC and still take the influences from all attributes into account, Jiang et al. [11] proposed to create a hidden parent for each attribute that combined the influences from all the other attributes.

Attribute weighting, in which various weights are assigned to different attributes, is another important method for improving NB. Jiang et al. [6] proposed to discriminatively assign each attribute a specific weight for each class. Two objective functions, namely, the conditional log likelihood (CLL) and the mean squared error (MSE), are introduced to obtain the optimized weight matrix. Yu et al. [8] assumed that highly predictive attribute values should be strongly associated with the class but not correlated with other attribute values, and different weights were assigned to attribute values by computing the difference between relevance and average redundancy. Jiang et al. [12] assumed that highly predictive attributes had similar characteristics, and the weight for each attribute was a sigmoid transformation of the difference between mutual relevance and average mutual redundancy. Zhang et al. [7] proposed to exploit attribute dependencies by considering the horizontal granularity of attribute values and the vertical granularity of class labels. Each attribute value is assigned a specific weight for each class discriminatively.

Ensemble learning provides a powerful machine learning paradigm that has exhibited excellent generalization ability by using multiple learners, especially "weak" ones [13], [14]. Averaged one-dependence estimators (AODE) [15] is an ensemble of super-parent one-dependence estimators (SPODEs). AODE achieves high classification accuracy

The associate editor coordinating the review of this manuscript and approving it for publication was Yeliz Karaca.

while decreasing variance. Each SPODE can be considered a weak learner because it selects a single attribute as the super-parent of all the other attributes, and its implicit independence assumption is unrealistic. AODE predicts by averaging the predictions of all these estimators, that is, all estimators in AODE are considered equal and are assigned with the same weight. Numerous approaches have been proposed to refine AODE, and they can be divided into four main categories:

- Attribute selection [16] performs attribute (parent or children or both) eliminations in each iteration to reduce zero-one loss.
- Attribute weighting [17] assigns discriminative weights to super-parent nodes.
- Model selection [18] selects specific SPODEs that are the most effective models within a model space.
- Model weighting [4], [17] computes the weight associated with each SPODE to combine their probability estimates linearly.

Ideally, the estimate of the joint probability distribution that corresponds to the learned BNC should approximate the true data distribution. However, overfitting to the training data may result in high variance, thereby harming the generalization performance. To accommodate the trade-off between bias and variance for different data quantities, scalable learners are highly appealing, especially for large data learning. We argue that overfitting to the testing instance will help improve rather than harm the generalization performance, and the significance of each SPODE should vary while classifying different instances, especially for highly predictive SPODEs. Yu et al. [4] considered the specific characteristics of each testing instance and adjusted the weights to different SPODEs adaptively by computing the correlation between the root attribute value and the class. However, the notion that the weight of each SPODE is irrelevant to non-root attribute values is not convincing. In this work, log likelihood is introduced to measure the extent to which each SPODE fits specific testing instance and based on this, appropriate weights are assigned to SPODEs in AODE while dealing with different testing instances.

The contributions of this paper are presented as follows:

- We prove theoretically and experimentally in terms of log likelihood that, the same SPODE in AODE may perform differentially while classifying different instances, and different SPODEs may perform differentially while classifying the same instance.
- We take each unlabeled testing instance as a target and propose a self-adaptive weighting approach that assigns discriminative weights to different SPODEs. For different instances, the weights adaptively change to enable the final ensemble to fit the instance remarkably.
- We report the results of an empirical evaluation comparing our algorithm, targeted AODE (TAODE), with state-of-the-art machine learning algorithms on 32 publicly available datasets. Our experiments show that TAODE displays comparable or better classification performance than a range of Bayesian and non-Bayesian learners.

The remainder of this paper is organized as follows. Section 2 provides a survey of related approaches. Section 3 describes our novel weighting approach for refining AODE. Section 4 presents the experimental evaluation of our proposed algorithm and comparisons with related approaches. Section 5 presents the conclusions and directions for future research.

II. BACKGROUND THEORY AND RELATED RESEARCH

Consider a finite attribute set $\boldsymbol{X} = \{X_1, \cdots, X_n\}$ and class variable Y, the classifier learned from training data is required to predict the class of an unlabeled instance $\boldsymbol{x} = (x_1, \cdots, , x_n)$, where lower case letter x_i denotes any possible value of attribute X_i. Among numerous classification techniques, BNCs provide compact and natural representation, effective inference, and efficient learning [20]. BNCs minimize error by selecting $\arg\max_y P(y|\boldsymbol{x})$, where $y \in \{y_1, \cdots, y_m\}$ is one of the m classes. A BNC refers to an annotated directed acyclic graph $\mathcal{G}$ that encodes a joint probability distribution over $\{\boldsymbol{X}, Y\}$. In the acyclic graph $\mathcal{G}$, vertices correspond to the variables $\{X_1, \cdots, X_n, Y\}$, and edges represent direct dependencies among these variables. Let Π_i denote the set of parents of X_i in $\mathcal{G}$, the joint probability distribution can be factorized according to the chain rule as follows,

$$P(\boldsymbol{x}, y) = P(y) \prod_{i=1}^{n} P(x_i|\Pi_i, y). \tag{1}$$

From Eq.(1), a key issue for learning BNC is the identification of significant conditional dependencies between attribute X_i and its parents Π_i.

A. NAIVE BAYES

Among all the BNCs, as shown in Fig.1 NB has the simplest topology and relatively stable classification efficiency. NB assumes that each attribute is independent from the rest of attributes given the class variable, and the independence assumption can be described as

$$P_{\mathrm{NB}}(x_1, \cdots, x_n|y) = \prod_{i=1}^{n} P(x_i|y). \tag{2}$$

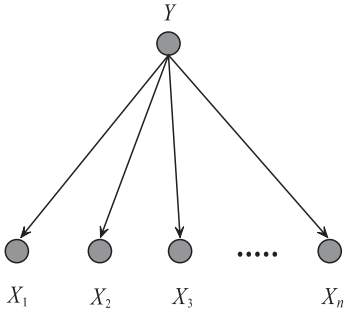


FIGURE 1. The topology of Naive Bayes.

Correspondingly, the joint probability distribution becomes

$$P_{\mathrm{NB}}(\boldsymbol{x}, y) = P(y) \prod_{i=1}^{n} P(x_i|y). \tag{3}$$

From Eq.(3), to compute $P_{\mathrm{NB}}(\boldsymbol{x}, y)$, NB needs only one single scan of the training data to generate two probability tables, one for estimating the prior class probability, and another for estimating the conditional probability for each attribute. Given that the topology of NB is definite and insensitive to the variation in training data, NB enjoys significantly low variance when compared with other BNCs. However, the unrealistic independence assumption is often questioned, and researchers believe that the assumption brings certain restrictions to the correct classification of NB. Therefore, relaxing the independence assumption has long been an important research direction for learning BNC.

B. AODE

Fig.2 shows that SPODE [21] is a three-dimensional probability estimator that relaxes the independence assumption by making all other attributes independent of each other given the class and one shared attribute X_α, the super-parent. That is,

$$P(\boldsymbol{x}, y) = P(y, x_\alpha) \prod_{i=1, i\neq\alpha}^{n} P(x_i|y, x_\alpha). \tag{4}$$

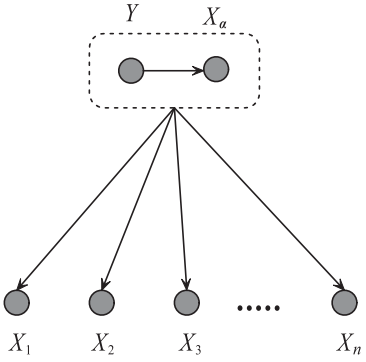


FIGURE 2. The topology of SPODE.

This weaker independence assumption is necessarily true if NB's is true and may also be true when NB's is not [22]. AODE utilizes a restricted class of SPODEs and aggregates the predictions of all qualified estimators. For a training dataset with n attributes, n candidate SPODEs can be considered, each taking a different attribute as its super-parent. Thus, AODE avoids model selection and maintains the robustness of NB.

Ensemble learning helps mitigate the negative effect caused by the possible biased independence assumption in one SPODE. For one SPODE in AODE that takes X_1 as the super-parent, X_2 and X_3 are conditionally independent. For another SPODE that takes X_2 as the super-parent, X_2 and X_3 are conditionally dependent. SPODEs in AODE are complementary in nature and can clarify the reason why AODE often exhibits excellent classification performance. Therefore, SPODE has the potential to be a substitute for NB while dealing with large data.

C. APPROACHES FOR OPTIMIZING AODE

SPODEs in AODE are assigned with the same weight and thus are treated equally. Although the topologies of these SPODEs seem similar, the conditional independence assumptions implicated vary greatly and represent different sets of conditional dependencies. If we can identify the difference among these SPODEs and assign appropriate weights to them, then the classification performance of AODE may be improved. Weights can be learned by experimental study in two ways. First is to conduct greedy search through a given interval. Although this approach can help AODE achieve significant advantage in classification performance, training such model on datasets with large number of attributes will result in great computational overhead. The reason is that the search space of weights will grow exponentially as the number of attributes increases. Second is to use different criteria, which can be proven relevant to classification, to weigh the discriminative characteristics of the SPODEs. Researchers proposed different weighting approaches, and the weights are respectively measured by the mutual information $I(X_i; Y)$ between the super-parent X_i and the class variable Y [17], classification accuracy of each SPODE [17], conditional log likelihood [23], [24] and area under the ROC curve [25], [26]. The comparison results prove that mutual information is effective despite its insignificant advantage [17].

The interdependence between attributes may vary greatly for different instances, and the identification of strong interdependence will help simplify the topology and enhance the estimates of conditional probabilities. Zheng and Webb [27] proposed Subsumption Resolution (SR) to address this issue. Given two attribute values x_i and x_j, if $P(x_j|x_i) = 1.0$, then x_i can subsume x_j, and the estimate of joint probability $P(y, \boldsymbol{x})$ can be simplified as follows,

$$P(y, \boldsymbol{x}) = P(y, x_1, \cdots, x_{j-1}, x_{j+1}, \cdots, x_n).$$

Variants of SR, that is, Near Subsumption Resolution, Lazy Subsumption Resolution and Eager subsumption resolution, were also proposed to detect interdependence and eliminate subsumed attribute-values at training time or classification time even if the dataset is ''polluted'' by noisy or erroneous data.

III. INFORMATION THEORY AND AODE

A. RELATIONSHIP BETWEEN ENTROPY AND BNC TOPOLOGY

Definition 1 [28]: Entropy measures the extent of unpredictability or uncertainty of a discrete random variable X and is defined as follows :

$$H(X) = -\sum_{X} P(x) log_2 P(x). \tag{5}$$

The greater the uncertainty of variable X, the larger the entropy $H(X)$, and the more information it needs to figure it out.

Definition 2 [28]: Joint entropy $H(X, Y)$ measures the extent of uncertainty of a pair of random variables X and Y and is defined as:

$$H(X, Y) = -\sum_{X}\sum_{Y} P(x, y) log_2 P(x, y). \tag{6}$$

Definition 3 [28]: Conditional entropy $H(X|Y)$ measures the extent of uncertainty of variable X when all possible values of variable Y are known. It is defined as:

$$H(X|Y) = -\sum_{X}\sum_{Y} P(x, y) log_2 P(x|y). \tag{7}$$

A BNC is a graphical representation of the joint probability distribution $P(y, \boldsymbol{x})$, which can be factorized and estimated according to the topology $\mathcal{B}$ learned from training data $\mathcal{D}$. Given the true probability distribution $P(y, \boldsymbol{x})$ and estimated distribution $P_{\mathcal{B}}(y, \boldsymbol{x})$, the average number of bits encoded in $\mathcal{B}$ for each instance in $\mathcal{D}$ can be computed by entropy function $H_{\mathcal{B}}$ as follows [29],

$$\begin{aligned} H_{\mathcal{B}} &= -\sum_{Y,X} P(y, x) \log P_{\mathcal{B}}(y, x) \\ &= -\sum_{Y} P(y) \log P(y) \\ &\quad -\sum_{i=1}^{n}\sum_{Y,X_i,\Pi_i} P(y, x_i, \Pi_i) \log P(x_i|y, \Pi_i^{\mathcal{B}}) \\ &= H(Y) + \sum_{i=1}^{n} H(X_i|Y, \Pi_i^{\mathcal{B}}), \end{aligned} \tag{8}$$

where $\Pi_i^{\mathcal{B}}$ denotes the set of parents of X_i in $\mathcal{B}$.

For any SPODE in AODE that takes X_α as the super-parent, $\Pi_\alpha = Y$ and $\Pi_i = \{X_\alpha, Y\}$ when $i \neq \alpha$. Thus the corresponding entropy function $H^{\alpha}_{\text{SPODE}}$ can be computed by

$$\begin{aligned} H^{\alpha}_{\text{SPODE}} &= H(Y) + H(X_\alpha|Y) + \sum_{i=1, i\neq\alpha}^{n} H(X_i|X_\alpha, Y) \\ &= H(Y, X_\alpha) + \sum_{i=1, i\neq\alpha}^{n} H(X_i|X_\alpha, Y). \end{aligned} \tag{9}$$

Obviously, for different SPODEs the topologies vary greatly and from Eq.(9), the extents to which SPODEs fit training data $\mathcal{D}$ also vary.

B. TARGETED AODE

If each SPODE in AODE is robust and performs similarly while dealing with different instances, then assigning fixed weights to individual SPODEs is appropriate. However, the discriminative independence assumptions cannot hold simultaneously. When different SPODEs deal with the same instance or the same SPODE deals with different instances, fixed weights may result in biased estimate of joint probability distribution $P(y, \boldsymbol{x})$ and unreliable classification results. For example, given dataset **Car** with six attributes $\{X_1, X_2, \cdots, X_6\}$ (see detail in Table 2) and six corresponding SPODEs, the estimates of $P(y, \boldsymbol{x})$ for each instance are shown in Fig.3. The estimates of $P(y, \boldsymbol{x})$ varies greatly for different

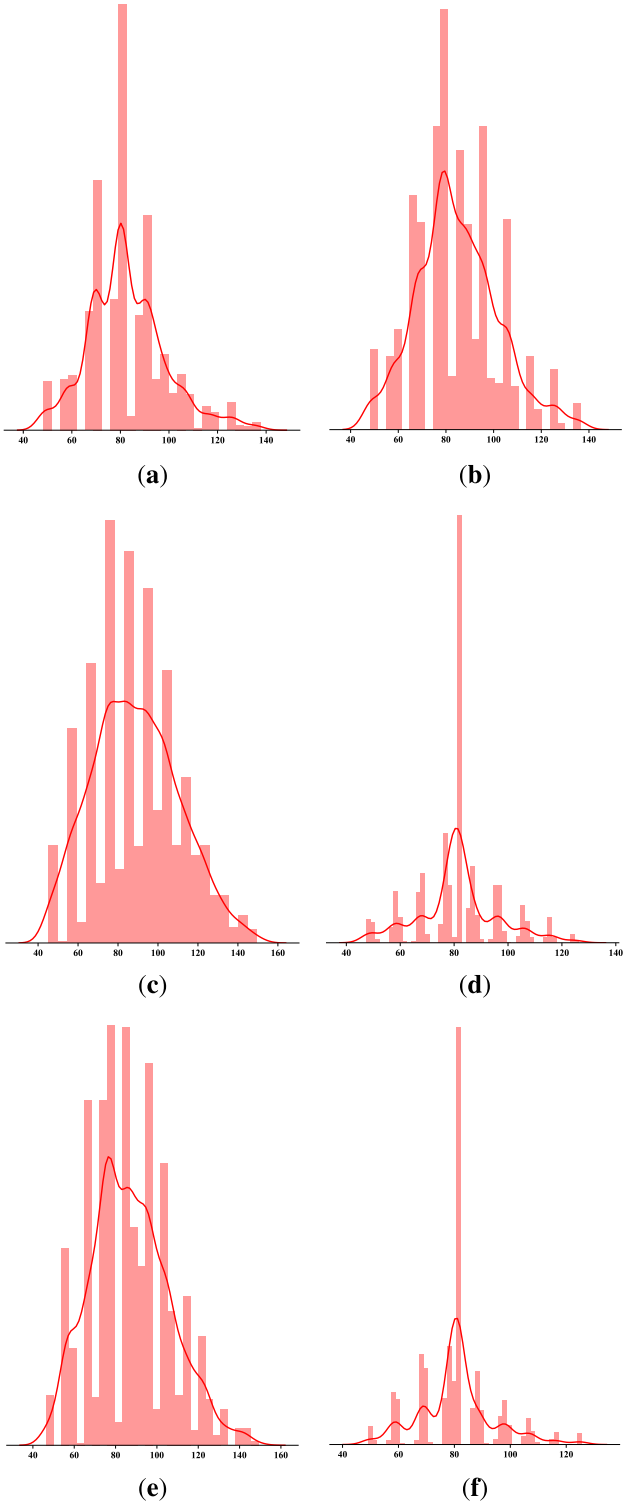


FIGURE 3. Estimates of joint probability distribution $P(y, x)$ for SPODEs on instances from dataset **Car**. These SPODEs respectively take (a) X_1, (b) X_2, (c) X_3, (d) X_4, (e) X_5 and (f) X_d as the super-parent.

SPODEs. Moreover, the SPODE, which takes X_4 as the super-parent, performs poorly, and corresponding independence assumptions may not hold for dataset **Car**(Fig.3(d)). Even for the best SPODE that takes X_3 as the super-parent (Fig.3(c)) its fitness to different instances also varies greatly. Thus, its independence assumption is effective in most cases but does not always hold.

Given testing instance $\boldsymbol{x} = \{x_1, \cdots, x_n\}$, its class label may take any one of the m possible values of variable Y. By assuming that the probability that $\boldsymbol{x}$ is in class y is $1/m$ for each $y \in \{y_1, \cdots, y_m\}$, $\boldsymbol{x}$ is transformed into a pseudo training set $\mathcal{T}$ as follows [30],

$$\mathcal{T} = \begin{cases} t_1 = \{x_1, \cdots, x_n, y_1\} \\ t_2 = \{x_1, \cdots, x_n, y_2\} \\ \cdots \\ t_m = \{x_1, \cdots, x_n, y_m\}. \end{cases} \tag{10}$$

Given training data $\mathcal{D}$ with N instances, $\boldsymbol{x}$ never appears in $\mathcal{D}$ and only appears $1/m$ times in $\mathcal{T}$. By adding $\mathcal{T}$ to $\mathcal{D}$, the joint probability $P(\boldsymbol{x}, y_i)$ or $P(t_i)$ can be estimated as follows

$$P(t_i) = \frac{\frac{1}{m}}{N+1} = \frac{1}{Nm+m} (1 \le i \le m).$$

Given topology $\mathfrak{b}$ that models the conditional dependencies between attribute values in $\boldsymbol{x}$, the log likelihood function $h_{\mathfrak{b}}(\mathcal{T})$ corresponds to the average number of bits encoded for each instance in $\mathcal{T}$ and can be computed by

$$\begin{aligned} h_{\mathfrak{b}}(\mathcal{T}) &= -\sum_{j=1}^{m} P(t_j) \log P_{\mathfrak{b}}(t_j) \propto -\sum_{j=1}^{m} \log P_{\mathfrak{b}}(t_j) \\ &= -\sum_{j=1}^{m} \log\{P(y_j) \prod_{i=1}^{n} P(x_i|y_j, \Pi_i^{\mathfrak{b}})\} \\ &= -\sum_{j=1}^{m} \log P(y_j) - \sum_{i=1}^{n} \sum_{j=1}^{m} \log P(x_i|y_j, \Pi_i^{\mathfrak{b}}) \\ &= h(Y) + \sum_{i=1}^{n} h(x_i|Y, \Pi_i^{\mathfrak{b}}). \end{aligned}$$

For any SPODE in AODE that takes X_α as the super-parent, $h_{\mathfrak{b}}(\mathcal{T})$ turns to be

$$\begin{aligned} h^{\alpha}_{\mathrm{SPODE}}(\mathcal{T}) &= -\sum_{j=1}^{m} \log P(y_j, x_\alpha) - \sum_{i=1, i\neq\alpha}^{n} \sum_{j=1}^{m} \log P(x_i|y_j, x_\alpha) \\ &= h(Y, x_\alpha) + \sum_{i=1, i\neq\alpha}^{n} h(x_i|Y, x_\alpha), \end{aligned} \tag{11}$$

where $h^{\alpha}_{\mathrm{SPODE}}(\mathcal{T})$ can be used to measure the extent to which the given SPODE fits testing instance $\boldsymbol{x}$. Subsequently, we take $\boldsymbol{x}$ as the target and apply $h^{\alpha}_{\mathrm{SPODE}}(\mathcal{T})$ as the benchmark for assigning weights. The class label of instance $\boldsymbol{x}$ is unknown or uncertain based on this observation and in Eq.(11). We can infer the corresponding definitions of entropy for classification as follows.

Definition 4: Targeted joint entropy measures the extent of uncertainty of a pair of random variables X_i and Y in the context of $\{X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n\}$ where $1 \le i \le n$, and is defined as:

$$h(Y, x_i) = -\sum_{Y} log_2 P(x_i, y). \tag{12}$$

Definition 5: Targeted conditional entropy measures the extent of uncertainty of random variables X_i given Y and X_j in the context of $\{X_1 = x_1, X_2 = x_2, \cdots, X_n = x_n\}$ where $1 \le i, j \le n$, and is defined as:

$$h(x_i|Y, x_j) = -\sum_{Y} log_2 P(x_i|y, x_j). \tag{13}$$

AODE is a linear combination of multiple SPODEs' joint probability estimates. To avoid overfitting and improve the generalization ability of AODE, the contributions of strong SPODEs to the aggregation of joint probability should be enhanced. Thus, if the SPODE fits instance well, then we need to increase its weight. TAODE seeks $\arg\max_y P(y, x)$ by using $P(y, x) = \sum_{\alpha=1}^{m} w_\alpha P_\alpha(y, x)$, where $P_\alpha(y, x)$ is the joint probability for SPODE with super-parent being X_α and the corresponding weight w_α is $1/h^{\alpha}_{\mathrm{SPODE}}(\mathcal{T})$.

To perform accurate calculations for assigning weight, the critical issue is to opt for suitable distributions for the probabilities. The prior joint probabilities in Eq.(11) will be estimated as follows

$$\begin{cases} \hat{P}(y_j, x_\alpha) = \dfrac{1}{N+1} \left[\sum_{k=1}^{N} \delta_k(y_j, x_\alpha) + \dfrac{1}{m} \right] \\ \hat{P}(x_i, y_j, x_\alpha) = \dfrac{1}{N+1} \left[\sum_{k=1}^{N} \delta_k(x_i, y_j, x_\alpha) + \dfrac{1}{m} \right], \end{cases} \tag{14}$$

where $\delta_k(\cdot)$ is a binary function, which is equal to 1 if the attribute values appear in the k-th instance and 0 if otherwise. Then, conditional probability $\hat{P}(x_i|y_j, x_\alpha)$ can be estimated as follows

$$\hat{P}(x_i|y_j, x_\alpha) = \frac{\hat{P}(x_i, y_j, x_\alpha)}{\hat{P}(y_j, x_\alpha)}. \tag{15}$$

The learning procedure of TAODE is shown as follows:

TABLE 1. Complexity summary for different BNCs where *t* is the number of training instances, *n* is the number of attributes, *v* is the maximum number of values per attribute, and *m* is the number of class labels.

BNC	Training time	Classification time
NB	$\mathcal{O}(tn)$	$\mathcal{O}(mn)$
HNB	$\mathcal{O}(tn^2 + m(nv)^2)$	$\mathcal{O}(mn^2)$
CFWNB	$\mathcal{O}(mnv + (nv)^2)$	$\mathcal{O}(mn)$
TAN	$\mathcal{O}(tn^2 + m(nv)^2 + n^2 logn)$	$\mathcal{O}(mn)$
AODE	$\mathcal{O}(tn^2)$	$\mathcal{O}(mn^2)$
WAODE-MI	$\mathcal{O}(tn^2 + mnv)$	$\mathcal{O}(mn^2)$
AODE-SR	$\mathcal{O}(tn^2)$	$\mathcal{O}(mn^2)$
TAODE	$\mathcal{O}(tn^2)$	$\mathcal{O}(mn^2)$

Algorithm 1 TAODE

Input: Training dataset $\mathcal{D}$ with N instances, testing instance $\boldsymbol{x} = \{x_1, x_2, \ldots x_n\}$.
Output: Class label for $\boldsymbol{x}$.
1. Generate a three-dimensional table of co-occurrence counts for each pair of attribute values and each class label.
2. Transform testing instance $\boldsymbol{x}$ to pseudo training dataset $\mathcal{T}$.
3. According to the attribute values in testing instance $\boldsymbol{x}$, for each class label y_j, compute $\hat{P}(x_i|y_j, x_\alpha)$ based on the estimates of $\hat{P}(y_j, x_\alpha)$ and $\hat{P}(x_i, y_j, x_\alpha)$.
4. For SPODE that takes x_α as the super-parent, compute $w_\alpha = 1/h^\alpha_{\text{SPODE}}(\mathcal{T})$.
5. Compute $P_\alpha(y, x)$ and output the class label by selecting $y^* = \arg\max_y \sum_{\alpha=1}^{m} w_\alpha P_\alpha(y, x)$.

Table 1 summarizes the time complexity of each BNC discussed. At training time, our implementation of TAODE generates a three-dimensional table of co-occurrence counts for each pair of attribute values and each class label. Then, the table is used to estimate the conditional probabilities in Eq.(4). The time complexity of forming the three dimensional probability table is $\mathcal{O}(tn^2)$. For SPODE with super-parent X_α in TAODE, to compute the weight w_α at classification time we need to compute the targeted joint entropy in Eq.(12) and targeted conditional entropy in Eq.(13). Thus, the time complexity for computing weights is $\mathcal{O}(mn^2)$. In comparison with AODE, which has a training time complexity of $\mathcal{O}(tn^2)$ and classification time complexity of $\mathcal{O}(mn^2)$, TAODE needs the same training time, and the additional computation of entropy functions does not increase the classification time complexity.

TABLE 2. Datasets.

No.	Dataset	Instance	Attribute	Class
1	Zoo	101	16	7
2	Promoters	106	57	2
3	Lymphography	148	18	4
4	Teaching-ae	151	5	3
5	Ionosphere	351	34	2
6	Dermatology	366	34	6
7	Cylinder-bands	540	39	2
8	Chess	551	39	2
9	Syncon	600	60	6
10	Vehicle	846	18	4
11	Anneal	898	38	6
12	Tic-tac-toe	958	9	2
13	Vowel	990	13	11
14	Contraceptive-mc	1473	9	3
15	Car	1,728	6	4
16	Segment	2,310	19	7
17	Hypothyroid	3,163	25	2
18	Splice-c4.5	3177	60	3
19	Kr-vs-kp	3,196	36	2
20	Dis	3,772	29	2
21	Sick	3,772	29	2
22	Abalone	4,177	8	3
23	Spambase	4,601	57	2
24	Waveform-5000	5,000	40	3
25	Wall-following	5,456	24	4
26	Optdigits	5,620	64	10
27	Thyroid	9,169	29	20
28	Sign	12,546	8	3
29	Magic	19,020	10	2
30	Letter-recog	20,000	16	26
31	Connect-4	67,557	42	3
32	Waveform	100,000	21	3

IV. EXPERIMENTS AND RESULTS

To compare the classification performance and fully clarify the difference among BNCs, we perform experiments on 32 benchmark datasets from the UCI machine learning repository [31], which are described in Table 2. The datasets can be divided into two groups, small datasets having less than 2k instances and relatively large datasets having more than 2k instances, and up to one million instances. Small and large datasets respectively account for 50% of the total 32 datasets. All the experiments have been conducted on a desktop computer with an Intel(R) Core(TM) i7-4710HQ CPU @ 2.5 GHz, 64 bits and 8G of memory. BNCs will run on the C++ software specifically designed for classification tasks, and non-Bayesian learners will run on the Weka workbranch (version 3.5.7). The following algorithms are compared:

- NB [5], Naive Bayes.
- HNB [11], Hidden Naive Bayes.
- CFWNB [12], A Correlation-based Feature Weighting Filter for Naive Bayes.
- TAN [9], Tree-augmented Naive Bayes.
- AODE [15], Averaged One-dependence Estimator.
- WAODE-MI [17], AODE which uses mutual information as the weight of each SPODE.
- AODE-SR [27], AODE with Subsumption Resolution.
- LR [32], Logistic Regression.
- LibSVM [33], Support Vector Machine.
- KNN [34], k-Nearest Neighbor.
- TAODE.

Some other issues related to the experiments are described as follows,

- All the datasets are preprocessed with an unsupervised filter to replace missing values with the modes and means from the existing data.
- For each original dataset, numeric attributes are discretized using minimum description length (MDL) discretization [35].
- Each algorithm is tested on each data set using 10 rounds 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.

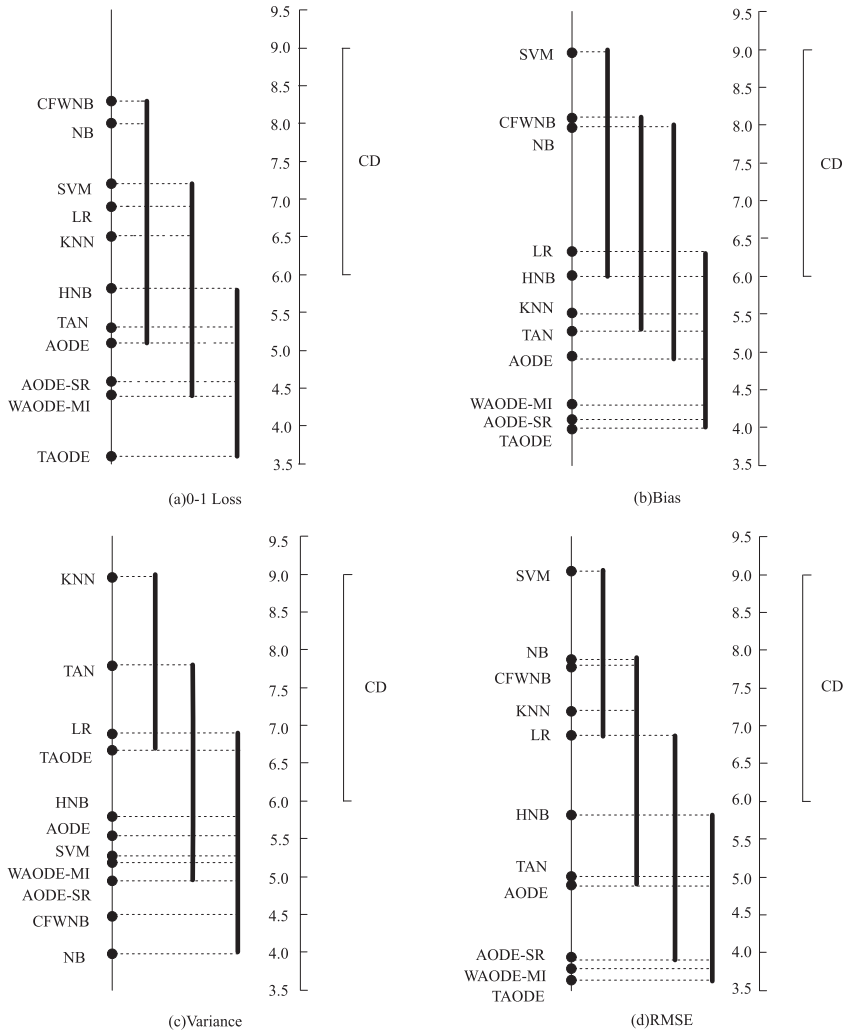


FIGURE 4. The comparison results of the Nemenyi test in terms of (a) zero-one Loss, (b) Bias, (c) Variance and (d) RMSE on 32 data sets. CD = 3.0646.

Tables 7, 8, 9 and 10 in the Appendix respectively show the experimental results in terms of zero-one loss, bias, variance and RMSE. Win-draw-loss records summarizing the relative zero-one loss, bias and variance are respectively shown in Tables 3, 4 and 5, and $Cell[i; j]$ in each table contains the number of datasets on which classifier on row i performs better, equally well or worse than the classifier on column j. Only when the outcome of a one-tailed binomial sign test is less than 0.05, the difference between algorithms is supposed to be significant.

TABLE 3. Win/Draw/Loss comparison results of zero-one loss on all datasets.

W/D/L	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN
HNB	20/3/9									
CFWNB	10/6/16	4/7/21								
TAN	23/4/5	11/11/10	21/5/6							
AODE	27/3/2	16/5/11	25/1/6	12/10/10						
WAODE-MI	26/4/2	18/5/9	25/4/3	16/9/7	7/19/6					
AODE-SR	27/3/2	17/7/8	25/3/4	14/11/7	6/26/0	7/20/5				
LR	14/4/14	12/4/16	20/2/10	7/4/21	6/4/22	9/3/20	6/4/22			
SVM	13/2/17	10/4/18	14/3/15	8/6/18	8/4/20	9/2/21	6/6/20	11/3/18		
KNN	14/1/17	15/4/13	20/2/10	11/1/20	11/1/20	12/1/19	11/1/20	16/1/15	20/1/11	
TAODE	28/3/1	18/5/9	25/5/2	18/8/6	14/17/1	9/21/2	8/21/3	23/3/6	21/3/8	21/0/11

A. RESULTS OF ZERO-ONE LOSS

NB performs better while dealing small datasets since the sparsely distributed data is more likely to approximate the independence assumption. HNB and TAN need to deeply mine the dependency relationships between predictive attributes or that between predictive attribute and the class variable, thus their advantage over NB will be significant while dealing with large datasets that can provide enough instances for estimating high-order probability distributions and computing conditional mutual information. For CFWNB, the attributes with maximum mutual relevance and minimum average mutual redundancy are considered to be highly predictive, and the attributes that are independent of others will be given priority. Thus similar to NB, CFWNB performs better while dealing small datasets. For AODE and its variants,

TABLE 4. Win/Draw/Loss comparison results of bias on all datasets.

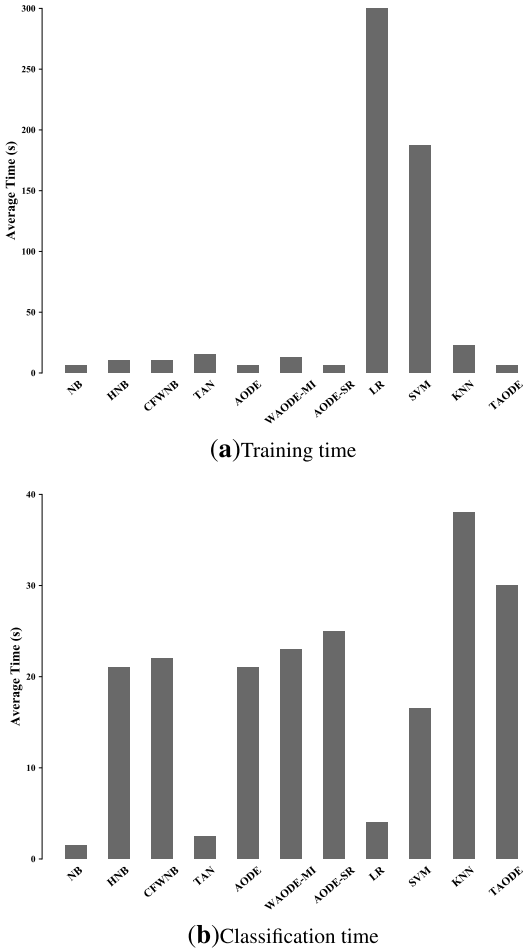
W/D/L	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN
HNB	20/2/10									
CFWNB	12/5/15	5/7/20								
TAN	24/2/6	16/7/9	23/0/9							
AODE	28/3/1	18/5/9	24/3/5	14/9/9						
WAODE-MI	28/1/3	19/5/8	25/4/3	16/11/5	7/20/5					
AODE-SR	28/3/1	20/4/8	24/3/5	16/10/6	3/29/0	4/23/5				
LR	17/3/12	15/5/12	21/4/7	10/4/18	8/5/19	8/4/20	8/5/19			
SVM	8/4/20	4/3/25	10/2/20	4/2/26	4/2/26	3/4/25	4/2/26	8/2/22		
KNN	18/2/12	17/2/13	24/1/7	14/1/17	14/2/16	13/3/16	14/1/17	15/1/16	26/2/4	
TAODE	28/2/2	18/6/8	24/3/5	16/9/7	6/25/1	6/21/5	6/23/3	21/5/6	26/3/3	16/3/13

TABLE 5. Win/Draw/Loss comparison results of variance on all datasets.

W/D/L	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN
HNB	8/3/21									
CFWNB	16/0/16	19/5/8								
TAN	3/3/26	8/3/21	8/1/23							
AODE	5/6/21	14/4/14	11/2/19	20/6/6						
WAODE-MI	7/4/21	16/3/13	12/3/17	22/5/5	6/20/6					
AODE-SR	7/6/19	14/5/13	12/2/18	23/4/5	7/21/1	7/19/6				
LR	10/2/20	12/1/19	8/1/23	15/1/16	12/3/17	9/3/20	12/3/17			
SVM	14/0/18	19/0/13	13/1/18	20/2/10	16/3/13	17/1/14	16/3/13	20/3/9		
KNN	3/2/27	6/0/26	4/1/27	8/2/22	4/2/26	2/4/26	2/3/27	8/4/20	8/3/21	
TAODE	4/7/21	12/4/16	12/2/18	22/3/7	3/18/11	6/16/10	1/17/14	16/4/12	13/1/18	26/2/4

TABLE 6. Win/Draw/Loss comparison results of RMSE on all datasets.

W/D/L	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN
HNB	20/6/6									
CFWNB	10/12/10	2/8/22								
TAN	25/4/3	9/15/8	22/5/5							
AODE	26/4/2	11/14/7	25/4/3	11/15/6						
WAODE-MI	27/3/2	13/13/6	25/5/2	12/16/4	3/27/2					
AODE-SR	27/3/2	12/14/6	26/4/2	13/15/4	1/31/0	2/29/1				
LR	12/10/10	10/5/17	15/6/11	5/7/20	5/7/20	4/5/23	4/7/21			
SVM	7/2/23	2/4/26	8/2/22	3/4/25	4/2/26	3/3/26	3/3/26	6/2/24		
KNN	10/3/19	10/2/20	13/4/15	8/3/21	7/4/21	7/4/21	7/4/21	12/2/18	20/4/8	
TAODE	27/3/2	13/13/6	25/6/1	15/12/5	4/28/0	2/29/1	2/30/0	23/5/4	26/3/3	21/4/7



(a)Training time

(b)Classification time

FIGURE 5. Time comparisons.

the independence assumption of each SPODE is much weaker and holds if that of NB holds, but not vice versa. The ensemble learning strategy of AODE greatly mitigates the negative effect caused by the independence assumption. WAODE-MI needs to compute mutual information for assigning weights, thus its advantage over AODE is significant while dealing with large datasets. AODE-SR improves upon AODE's probability estimates by detecting and deleting the generalization relationships between attribute values that are instantiated in the object being classified, that is, AODE-SR can adaptively remove redundant attribute values in testing instance at classification time and make the topology of each SPODE in AODE fit testing instance better. A user-specified minimum frequency is used to evaluate the confidence level of the generalization relationship. TAODE has the characteristics of WAODE-MI (identifying the difference among SPODEs and assigning discriminative weights) and AODE-SR (identifying the variation in conditional dependencies between attribute values in different instances).

The experimental results support the above analysis. From Table 3, HNB and TAN respectively beat NB on 20 and 23 datasets, most of them are large datasets. CFWNB beats NB on 10 datasets, most of them are small datasets. TAN performs better than NB, HNB and CFWNB, that shows the negative effect caused by the unrealistic independence assumption of NB to some extent. In contrast, AODE and its variants (including WAODE-MI and AODE-SR) all enjoy advantages over TAN. Although WAODE-MI performs similarly to AODE, its advantage over TAN is much more significant especially while dealing with large datasets. AODE is competitive with TAN (12 wins and 10 losses), whereas WAODE-MI beats TAN on 16 datasets and loses on 7, and WAODE-MI never loses to TAN when the data size$>$ 2000. AODE-SR performs even better, it beats AODE on 6 datasets and never loses, and it beats WAODE-MI on 7 datasets and loses on 5. TAODE enjoys significant advantage over AODE (14 wins and only 1 loss), and it also performs much better than WAODE-MI (9 wins and 2 losses) and AODE-SR (8 wins and 3 losses). When compared with non-Bayesian learners, TAODE wins on the majority of datasets in terms of zero-one loss. For example, TAODE beats LR on 23 datasets and loses on 6.

B. RESULTS OF BIAS AND VARIANCE

Complex topology often results in higher variance and lower bias. If non-significant conditional dependencies that carry no useful information about the class are represented as if they do, that may invariably bias the estimates of probability distributions. The topologies of HNB, CFWNB and TAN can

TABLE 7. Experimental results of 0-1 loss.

Dataset	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN	TAODE
Zoo	0.0297	0.0000	0.0396	0.0099	0.0297	0.0297	0.0297	0.0594	0.0891	0.0396	0.0198
Promoters	0.0755	0.0755	0.0849	0.1321	0.1321	0.1415	0.1321	0.1321	0.0660	0.1792	0.1226
Lymphography	0.1486	0.1689	0.1486	0.1757	0.1689	0.1554	0.1689	0.2162	0.1824	0.1892	0.1554
Teaching-ae	0.4967	0.4570	0.4570	0.5497	0.4901	0.4503	0.4768	0.4040	0.5298	0.3377	0.4636
Ionosphere	0.1054	0.0684	0.0883	0.0684	0.0741	0.1565	0.0627	0.1111	0.0655	0.1368	0.0741
Dermatology	0.0191	0.0246	0.0219	0.0328	0.0164	0.0191	0.0164	0.0301	0.1694	0.0546	0.0191
Cylinder-bands	0.2148	0.2315	0.2870	0.2833	0.1889	0.1796	0.1852	0.2130	0.2333	0.2556	0.1870
Chess	0.1125	0.0998	0.1379	0.0926	0.0998	0.0944	0.1016	0.1143	0.1615	0.0708	0.0799
Syncon	0.0283	0.0083	0.0200	0.0083	0.0100	0.0100	0.0100	0.1700	0.5283	0.0350	0.0100
Vehicle	0.3924	0.2624	0.3865	0.2943	0.2896	0.2872	0.2884	0.2045	0.7092	0.3002	0.2766
Anneal	0.0379	0.0869	0.0846	0.0111	0.0089	0.0089	0.0078	0.1303	0.1604	0.0434	0.0078
Tic-tac-toe	0.3069	0.2244	0.3100	0.2286	0.2651	0.2724	0.2651	0.0167	0.1221	0.0125	0.2630
Vowel	0.4242	0.0616	0.3101	0.1303	0.1495	0.1949	0.1505	0.1818	0.1495	0.0071	0.1323
Contraceptive-mc	0.5037	0.4779	0.4671	0.4888	0.4942	0.4922	0.4942	0.4881	0.4515	0.5567	0.4902
Car	0.1400	0.0671	0.2344	0.0567	0.0816	0.0885	0.0816	0.0689	0.0810	0.0648	0.0810
Segment	0.0788	0.0519	0.0931	0.0390	0.0342	0.0338	0.0351	0.0416	0.3463	0.0286	0.0346
Hypothyroid	0.0149	0.0253	0.0266	0.0104	0.0136	0.0104	0.0120	0.0180	0.0417	0.0291	0.0111
Splice-c4.5	0.0444	0.0403	0.0375	0.0466	0.0365	0.0365	0.0365	0.0938	0.0381	0.2575	0.0362
Kr-vs-kp	0.1214	0.0754	0.0645	0.0776	0.0842	0.0576	0.0720	0.0244	0.0610	0.0372	0.0773
Dis	0.0159	0.0162	0.0188	0.0159	0.0130	0.0143	0.0119	0.0170	0.0154	0.0170	0.0125
Sick	0.0308	0.0220	0.0247	0.0257	0.0273	0.0244	0.0252	0.0323	0.0610	0.0382	0.0249
Abalone	0.4762	0.4779	0.4752	0.4587	0.4472	0.4475	0.4475	0.4446	0.4654	0.4965	0.4465
Spambase	0.1015	0.1469	0.1487	0.0669	0.0672	0.0648	0.0680	0.0759	0.1634	0.0922	0.0602
Waveform-5000	0.2006	0.1654	0.1898	0.1844	0.1462	0.1450	0.1424	0.1340	0.1358	0.2638	0.1466
Wall-following	0.1054	0.1252	0.1584	0.0554	0.0370	0.0367	0.0370	0.2951	0.0971	0.1182	0.0361
Optdigits	0.0767	0.0411	0.0717	0.0407	0.0311	0.0302	0.0302	0.0546	0.2687	0.0139	0.0290
Thyroid	0.1111	0.2204	0.2182	0.0720	0.0701	0.0655	0.0686	0.1205	0.2368	0.1980	0.0629
Sign	0.3586	0.2800	0.4076	0.2755	0.2821	0.2768	0.2817	0.4063	0.3273	0.1340	0.2743
Magic	0.2239	0.1853	0.2300	0.1675	0.1752	0.1762	0.1751	0.2089	0.3412	0.1906	0.1725
Letter-recog	0.2525	0.1385	0.2862	0.1300	0.0883	0.0853	0.0879	0.2258	0.0243	0.0404	0.0838
Connect-4	0.2783	0.2439	0.2847	0.2354	0.2420	0.2406	0.2384	0.2425	0.2242	0.1900	0.2374
Waveform	0.0220	0.0449	0.0696	0.0202	0.0180	0.0181	0.0181	0.0279	0.0271	0.0404	0.0182

be regarded as different versions of structural augmentation of NB. This alleviates some of NB's independence assumption and therefore reduces its bias at the expense of increasing its variance. AODE needs no structure learning and different SPODEs in AODE represent different independence assumptions, the ensemble learning strategy will help AODE achieve the bias-variance trade-off. The variation in training data will affect the computation of mutual information, thus WAODE-MI will have lower variance compared to AODE. In contrast, SR removes interdependencies between attribute values and that will help reduce variance. Identification of significant conditional dependencies implicated in each instance will help achieve the bias-variance trade-off. TAODE takes each testing instance as the target and the weight for each SPODE is finely adjusted for different testing instances.

To evaluate the extent to which TAODE can accommodate the trade-off between bias and variance, we run the bias-variance decomposition experiments [36] together with the repeated cross-validation bias-variance estimation method [37]. For each fold of cross validation, the bias-variance decomposition is derived from the error on each of the testing instances and we can obtain the mean bias and variance after the validation process terminates. From Table 4 we can see that, the negative effect caused by the independence assumption and its variants makes NB, HNB and CFWNB perform poorer than TAN in terms of bias, whereas AODE and its variants (including TAODE) enjoy significant advantages over these single model BNCs and non-Bayesian learners. All the three variants of AODE achieve lower bias more often than AODE but the advantages of WAODE-MI and AODE-SR over AODE are not as significant as that of TAODE over AODE. As proposed by Brain and Webb [38], the learners with lower bias can perform better while dealing with large data. TAODE has proved to be computationally efficient low bias learners. This can be proved by the experimental results of zero-one loss that TAODE only loses to AODE on dataset **Dermatology** having only 366 instances and when the data size$>$ 500, TAODE never loses.

TABLE 8. Experimental results of bias.

Dataset	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN	TAODE
Zoo	0.0318	0.0797	0.0840	0.0303	0.0273	0.0273	0.0273	0.0955	0.2022	0.0624	0.0282
Promoters	0.0786	0.0521	0.0426	0.1329	0.4777	0.5489	0.4777	0.0390	0.2010	0.0993	0.4000
Lymphography	0.0902	0.1596	0.1647	0.1027	0.0933	0.0951	0.0933	0.1422	0.2802	0.1687	0.0931
Teaching-ae	0.4836	0.4194	0.3893	0.4566	0.4370	0.3984	0.4370	0.3239	0.4843	0.3639	0.4198
Ionosphere	0.1220	0.0725	0.0747	0.0804	0.0744	0.0751	0.0726	0.1117	0.0973	0.1368	0.0764
Dermatology	0.0079	0.0205	0.0143	0.0274	0.0055	0.0061	0.0050	0.0200	0.3482	0.0489	0.0057
Cylinder-bands	0.2000	0.2059	0.2341	0.3117	0.1589	0.1501	0.1472	0.1863	0.3630	0.2012	0.1610
Chess	0.1413	0.0752	0.0932	0.1437	0.1290	0.1286	0.1244	0.0832	0.2299	0.0807	0.1230
Syncon	0.0516	0.0081	0.0126	0.0203	0.0185	0.0180	0.0187	0.1123	0.4597	0.0281	0.0185
Vehicle	0.3330	0.1945	0.3081	0.2382	0.2415	0.2398	0.2401	0.1540	0.4033	0.2094	0.2412
Anneal	0.0354	0.0874	0.0905	0.0201	0.0214	0.0194	0.0208	0.0788	0.1654	0.0415	0.0214
Tic-tac-toe	0.2614	0.1856	0.2257	0.1746	0.2005	0.2104	0.2005	0.0236	0.2041	0.0409	0.2008
Vowel	0.3301	0.1464	0.2482	0.1942	0.1895	0.1811	0.1805	0.1688	0.2587	0.1102	0.1698
Contraceptive-mc	0.3928	0.3586	0.3858	0.3425	0.3816	0.3766	0.3811	0.3846	0.3568	0.3577	0.3735
Car	0.0937	0.0595	0.1852	0.0478	0.0556	0.0633	0.0556	0.0536	0.1141	0.0799	0.0550
Segment	0.0785	0.0472	0.0849	0.0491	0.0367	0.0357	0.0370	0.0384	0.3957	0.0297	0.0353
Hypothyroid	0.0116	0.0242	0.0245	0.0104	0.0094	0.0099	0.0095	0.0181	0.0449	0.0262	0.0099
Splice-c4.5	0.0351	0.0390	0.0345	0.0395	0.0308	0.0315	0.0308	0.0614	0.0346	0.1699	0.0308
Kr-vs-kp	0.1107	0.0702	0.0583	0.0702	0.0747	0.0518	0.0654	0.0210	0.0812	0.0531	0.0688
Dis	0.0165	0.0187	0.0208	0.0193	0.0170	0.0179	0.0162	0.0195	0.0183	0.0171	0.0178
Sick	0.0246	0.0201	0.0223	0.0207	0.0224	0.0216	0.0227	0.0277	0.0612	0.0354	0.0228
Abalone	0.4180	0.3670	0.3520	0.3126	0.3201	0.3212	0.3199	0.3613	0.3975	0.3320	0.3183
Spambase	0.0929	0.1360	0.1374	0.0570	0.0606	0.0574	0.0600	0.0560	0.1654	0.0763	0.0541
Waveform-5000	0.1762	0.1451	0.1781	0.1232	0.1235	0.1184	0.1212	0.1207	0.1136	0.1662	0.1213
Wall-following	0.0951	0.0956	0.1312	0.0491	0.0251	0.0253	0.0251	0.2512	0.1065	0.1105	0.0260
Optdigits	0.0685	0.0393	0.0598	0.0275	0.0233	0.0224	0.0229	0.0382	0.3010	0.0139	0.0224
Thyroid	0.0994	0.2049	0.2088	0.0587	0.0611	0.0561	0.0595	0.0999	0.2570	0.1622	0.0550
Sign	0.3257	0.2564	0.3843	0.2420	0.2531	0.2461	0.2489	0.3835	0.3322	0.1140	0.2446
Magic	0.2111	0.1674	0.2019	0.1252	0.1600	0.1541	0.1534	0.2018	0.3404	0.1341	0.1546
Letter-recog	0.2207	0.1212	0.2496	0.1032	0.0876	0.0823	0.0868	0.1945	0.0370	0.0414	0.0814
Connect-4	0.2660	0.2263	0.2740	0.2253	0.2264	0.2237	0.2224	0.2279	0.2282	0.1592	0.2153
Waveform	0.0219	0.0423	0.0674	0.0152	0.0156	0.0158	0.0157	0.0267	0.0262	0.0245	0.0149

From Table 5 NB performs the best in terms of variance. In contrast, the advantage of AODE over other BNCs is not significant as supposed. WAODE-MI performs similarly to AODE (6 wins and 6 losses), and AODE-SR even performs better than AODE (7 wins and only 1 loss). TAODE transforms each testing instance into a pseudo training set and by assigning weights to different SPODEs, significant and non-significant conditional dependencies implicated in testing instance will be assigned different weights correspondingly for probability estimates. That surely increases the risk of overfitting. TAODE only beats TAN in terms of variance (22 wins and 7 losses) and loses to other BNCs. Thus its zero-one loss advantage can be attributed to the cost of variance. For non-Bayesian learners, SVM performs the best in terms of variance due to its default parameters applied to all datasets.

C. RESULT OF RMSE

The root-mean-square error (RMSE) is a frequently used objective function and can measure the calibration of a classifier's class probability predictions. The comparison results shown in Table 6 are consistent with that in terms of zero-one loss. HNB and TAN enjoy significant advantages over NB especially while dealing with large datasets. CFWNB performs better than NB while dealing with small datasets. AODE and its variants perform better than single model BNCs mentioned above generally. TAODE performs the best among all the BNCs. Among these three variants of AODE, for SPODE with the same super-parents the same 1-dependence relationships are represented in the topology and the precision of estimates of conditional probabilities is restricted, thus the advantage of TAODE in terms of RMSE does not seem to be significant. When compared to these non-Bayesian learners, the advantages of AODE and its variants are very obvious.

D. FRIEDMAN TEST AND NEMENYI TEST

The Friedman [39] and the Nemenyi [40] tests are effective for comparing multiple classifiers across multiple data sets. Classifier will be ranked by comparing their classification

TABLE 9. Experimental results of variance.

Dataset	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN	TAODE
Zoo	0.0439	0.0644	0.0675	0.0606	0.0424	0.0424	0.0424	0.1032	0.1062	0.0745	0.0445
Promoters	0.0786	0.0891	0.0829	0.1729	0.0994	0.0654	0.0994	0.0794	0.2224	0.1049	0.1486
Lymphography	0.0343	0.0666	0.0568	0.1116	0.0476	0.0478	0.0476	0.1212	0.0796	0.0815	0.0498
Teaching-ae	0.1484	0.2110	0.1766	0.1914	0.1650	0.1776	0.1650	0.2564	0.1879	0.1932	0.1622
Ionosphere	0.0242	0.0198	0.0163	0.0401	0.0385	0.0368	0.0368	0.0946	0.0281	0.0361	0.0381
Dermatology	0.0216	0.0382	0.0267	0.0513	0.0199	0.0242	0.0188	0.0418	0.1335	0.0546	0.0213
Cylinder-bands	0.0656	0.0859	0.1001	0.0739	0.0961	0.1010	0.1067	0.1437	0.0511	0.1373	0.0923
Chess	0.0401	0.0439	0.0507	0.0486	0.0415	0.0364	0.0422	0.0791	0.0027	0.0797	0.0448
Syncon	0.0204	0.0145	0.0135	0.0222	0.0225	0.0230	0.0218	0.1764	0.3633	0.0316	0.0235
Vehicle	0.1120	0.1160	0.0918	0.1299	0.1277	0.1276	0.1287	0.0797	0.3345	0.1398	0.1273
Anneal	0.0168	0.0306	0.0333	0.0161	0.0197	0.0161	0.0160	0.0593	0.0264	0.0267	0.0174
Tic-tac-toe	0.0455	0.0717	0.0550	0.0824	0.0513	0.0604	0.0513	0.0245	0.0528	0.0591	0.0528
Vowel	0.2542	0.1799	0.1996	0.2445	0.2344	0.2310	0.2304	0.2239	0.1894	0.1614	0.2284
Contraceptive-mc	0.0856	0.1464	0.0990	0.1646	0.1058	0.1106	0.1077	0.1187	0.1090	0.1947	0.1238
Car	0.0520	0.0438	0.0484	0.0376	0.0438	0.0427	0.0438	0.0385	0.0370	0.0756	0.0441
Segment	0.0259	0.0215	0.0226	0.0294	0.0255	0.0255	0.0264	0.0266	0.2633	0.0259	0.0262
Hypothyroid	0.0031	0.0041	0.0015	0.0034	0.0034	0.0033	0.0030	0.0062	0.0006	0.0109	0.0030
Splice-c4.5	0.0078	0.0136	0.0068	0.0289	0.0085	0.0083	0.0085	0.1194	0.0155	0.1185	0.0085
Kr-vs-kp	0.0186	0.0194	0.0169	0.0152	0.0186	0.0119	0.0166	0.0134	0.0026	0.0559	0.0208
Dis	0.0069	0.0005	0.0065	0.0005	0.0071	0.0021	0.0041	0.0038	0.0001	0.0062	0.0040
Sick	0.0047	0.0038	0.0026	0.0051	0.0042	0.0057	0.0039	0.0084	0.0008	0.0197	0.0045
Abalone	0.0682	0.1184	0.1095	0.1693	0.1544	0.1543	0.1543	0.0746	0.0625	0.1750	0.1561
Spambase	0.0092	0.0183	0.0179	0.0158	0.0105	0.0111	0.0105	0.0243	0.0559	0.0644	0.0124
Waveform-5000	0.0259	0.0364	0.0195	0.0690	0.0410	0.0420	0.0413	0.0310	0.0320	0.1163	0.0426
Wall-following	0.0211	0.0231	0.0235	0.0288	0.0242	0.0242	0.0242	0.0481	0.0390	0.0696	0.0245
Optdigits	0.0153	0.0177	0.0207	0.0185	0.0139	0.0137	0.0138	0.0752	0.3350	0.0106	0.0140
Thyroid	0.0205	0.0231	0.0105	0.0257	0.0235	0.0239	0.0230	0.0453	0.0038	0.0895	0.0243
Sign	0.0313	0.0246	0.0197	0.0386	0.0378	0.0403	0.0384	0.0236	0.0210	0.0673	0.0406
Magic	0.0174	0.0187	0.0202	0.0490	0.0297	0.0289	0.0291	0.0064	0.0019	0.0707	0.0313
Letter-recog	0.0471	0.0561	0.0536	0.0591	0.0448	0.0455	0.0448	0.0422	0.0293	0.0448	0.0457
Connect-4	0.0156	0.0153	0.0092	0.0149	0.0222	0.0215	0.0205	0.0127	0.0107	0.0691	0.0301
Waveform	0.0009	0.0033	0.0007	0.0053	0.0025	0.0023	0.0025	0.0024	0.0026	0.0181	0.0034

performance and the Friedman statistic is defined as follows:

$$\chi_F^2 = \frac{12}{Dt(t+1)} \sum_{j=1}^{t} R_j^2 - 3D(t+1), \tag{16}$$

where $R_j^2 = \sum_i r_i^j$ and r_i^j is the rank of the jth of t algorithms on the ith of D data sets. The Friedman statistic is distributed according to χ_F^2 with $t-1$ degrees of freedom. The critical value of χ_α^2 for $\alpha = 0.05$ with $t-1 = 10$ degrees of freedom is 18.31. The Friedman statistic for zero-one loss, bias, variance and RMSE in our experiments are 67.19, 87.54, 63.43 and 106.8, respectively, $\chi_F^2 > \chi_\alpha^2$ always holds and hence there exists true difference among these algorithms.

Let d_{ij} denote the difference between the average rank of the i-th algorithm and that of the j-th algorithm. The difference between the algorithms is supposed to be significant if $d_{ij} >$ critical difference (CD) [41], which can be computed as follows,

$$CD = q_\alpha \sqrt{\frac{t(t+1)}{6D}} \tag{17}$$

where q_α for $\alpha = 0.05$ and $t = 11$ is 3.696. The experimental study is performed on 32 datasets with 11 algorithms, CD can be computed by Eq.(17) and is equal to 3.0646. The comparison results of these algorithms against each other with the Nemenyi test on zero-one loss, bias, variance and RMSE are shown in Fig.4. The left line and the parallel right line respectively indicate the algorithms and corresponding average ranks. CD is also presented in the graphs. The lower position of the compared algorithm indicates its lower rank or better performance. If the difference between algorithms is not significant, they will connected by a line.

As Fig.4(a) shows, TAODE achieves the lowest mean zero-one loss rank (3.515), followed by WAODE-MI (4.468) and AODE-SR (4.593). They enjoy significant zero-one loss advantages over AODE, TAN, HNB, CFWNB, k-NN, LR, NB and LibSVM. When bias is compared, as shown in Fig.4(b), TAODE, WAODE-MI and AODE-SR still perform the best. When variance is compared, as shown in Fig.4(c), the variance advantage of NB relative to all remaining algorithms is very clear. TAODE achieves higher

TABLE 10. Experimental results of RMSE.

Dataset	NB	HNB	CFWNB	TAN	AODE	WAODE-MI	AODE-SR	LR	SVM	KNN	TAODE
Zoo	0.0802	0.0495	0.0933	0.0647	0.0677	0.0724	0.0677	0.1261	0.1596	0.0941	0.0686
Promoters	0.2544	0.2437	0.2417	0.3264	0.2911	0.2913	0.2911	0.3636	0.2570	0.3798	0.2976
Lymphography	0.2446	0.2489	0.2419	0.2684	0.2478	0.2496	0.2475	0.3242	0.3020	0.2759	0.2501
Teaching-ae	0.4789	0.4675	0.4418	0.4825	0.4670	0.4668	0.4679	0.4743	0.5943	0.4591	0.4644
Ionosphere	0.3157	0.2521	0.2846	0.2615	0.2506	0.2489	0.2443	0.3035	0.2560	0.3686	0.2464
Dermatology	0.0631	0.0775	0.0715	0.0851	0.0692	0.0688	0.0677	0.0975	0.2376	0.1339	0.0698
Cylinder-bands	0.4291	0.4155	0.4358	0.4358	0.4080	0.4016	0.4017	0.4465	0.4830	0.5045	0.4056
Chess	0.2944	0.2717	0.3208	0.2594	0.2725	0.2603	0.2669	0.2771	0.4019	0.2608	0.2502
Syncon	0.0931	0.0435	0.0737	0.0500	0.0554	0.0537	0.0538	0.2299	0.4197	0.1075	0.0556
Vehicle	0.3934	0.3050	0.3717	0.3104	0.3097	0.3099	0.3086	0.2581	0.5955	0.3864	0.3083
Anneal	0.0981	0.1393	0.1484	0.0543	0.0546	0.0536	0.0519	0.1784	0.2312	0.1195	0.0529
Tic-tac-toe	0.4309	0.3888	0.4334	0.4023	0.3995	0.4085	0.3995	0.1289	0.3495	0.2315	0.3984
Vowel	0.2270	0.0936	0.1923	0.1271	0.1425	0.1633	0.1420	0.1722	0.1649	0.0358	0.1324
Contraceptive-mc	0.4506	0.4403	0.4389	0.4391	0.4398	0.4385	0.4398	0.4384	0.5486	0.5990	0.4394
Car	0.2252	0.1706	0.2534	0.1617	0.2005	0.1983	0.2005	0.1520	0.2013	0.1953	0.1994
Segment	0.1398	0.1052	0.1400	0.0967	0.0879	0.0870	0.0885	0.0972	0.3146	0.0902	0.0881
Hypothyroid	0.1138	0.1430	0.1541	0.0955	0.1036	0.0967	0.0986	0.1221	0.2043	0.1705	0.0974
Splice-c4.5	0.1511	0.1411	0.1388	0.1544	0.1366	0.1367	0.1366	0.2489	0.1593	0.3727	0.1366
Kr-vs-kp	0.3022	0.2501	0.2779	0.2358	0.2638	0.2343	0.2565	0.1474	0.2470	0.1946	0.2561
Dis	0.1177	0.1110	0.1201	0.1103	0.1080	0.1046	0.0996	0.1228	0.1240	0.1302	0.1047
Sick	0.1700	0.1331	0.1467	0.1434	0.1572	0.1452	0.1500	0.1643	0.2469	0.1953	0.1511
Abalone	0.4630	0.4357	0.4418	0.4250	0.4193	0.4193	0.4193	0.4168	0.5570	0.5751	0.4195
Spambase	0.2994	0.3243	0.3326	0.2403	0.2336	0.2301	0.2335	0.2435	0.4043	0.3036	0.2239
Waveform-5000	0.3348	0.2805	0.3048	0.2947	0.2659	0.2635	0.2637	0.2525	0.3009	0.4192	0.2651
Wall-following	0.2177	0.2259	0.2478	0.1586	0.1292	0.1293	0.1292	0.3122	0.2204	0.2430	0.1298
Optdigits	0.1163	0.0835	0.1080	0.0837	0.0733	0.0729	0.0726	0.1034	0.2318	0.0526	0.0727
Thyroid	0.0967	0.1316	0.1314	0.0746	0.0745	0.0715	0.0731	0.0949	0.1539	0.1405	0.0706
Sign	0.3984	0.3565	0.4104	0.3505	0.3524	0.3519	0.3520	0.4210	0.4671	0.2892	0.3487
Magic	0.3974	0.3605	0.3876	0.3461	0.3541	0.3526	0.3542	0.3839	0.5841	0.4366	0.3519
Letter-recog	0.1184	0.0868	0.1225	0.0860	0.0707	0.0695	0.0704	0.1127	0.0432	0.0551	0.0691
Connect-4	0.3587	0.3378	0.3632	0.3315	0.3370	0.3356	0.3342	0.3358	0.3866	0.3069	0.3339
Waveform	0.1176	0.1392	0.2020	0.0951	0.0860	0.0860	0.0860	0.1158	0.1345	0.1641	0.0865

mean variance rank (6.718) compared to AODE, WAODE-MI and AODE-SR. KNN has the highest variance rank (9.046) among all the algorithms. When RMSE is compared, as shown in Fig.4(d), the Nemenyi test differentiates TAODE, WAODE-MI and AODE-SR from the other learners. They three deliver significantly lower mean RMSE ranks.

E. ANALYSIS OF CLASSIFICATION AND TRAINING TIME

Fig.5 displays the mean training and classification time on 32 datasets for those algorithms. As can be seen from Fig.5(a), NB and AODE don't need to learn the topology and thus their training time is the least among all the BNCs. NB needs to store the prior probability $P(y)$ and conditional probability $P(x_i|y)$. AODE takes a bit more time to store high-order conditional probability $P(x_i|x_j, y)$, and AODE-SR and TAODE perform in a similar fashion. HNB needs to compute the conditional mutual information for each pair of attributes and use it to compute the weights. In contrast, WAODE-MI and TAN respectively need to compute mutual information for assigning weights and conditional mutual information for learning conditional dependencies between attributes, thus they both suffer the training time disadvantages over other BNCs. As can be seen from Fig.5(b), AODE is extremely computationally expensive especially at classification time compared to NB and TAN. AODE and HNB have almost the same classification time. AODE-SR needs to identify the generalization relationship and eliminates generalizations at classification time. TAODE needs to compute $h^{\alpha}_{\text{SPODE}}(\mathcal{T})$ to assign weight to each SPODE. Thus they both suffer the classification time disadvantages over other BNCs. There is no doubt that LR, SVM and KNN use more training and classification time than BNCs, actually they are more complex and programs written with Java run slowly.

V. CONCLUSION

AODE retains the simplicity and direct theoretical foundation of NB without incurring computational overhead. By identifying the difference among the SPODEs in terms of log likelihood, this paper presents an efficient and effective attribute value weighting approach, which is well balanced

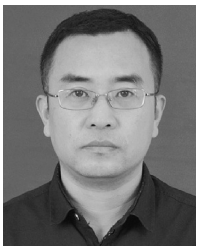
between the expressivity caused by ensemble learning strategy and the reliable probability estimation caused by independence assumptions. For different instances, discriminative weights are assigned to different SPODEs by computing the micro entropy function $h^{\alpha}_{\text{SPODE}}(\mathcal{T})$. The experimental results on widely used benchmark datasets from UCI machine learning repository show that this approach achieves bias-variance trade-off and is a competitive alternative to state-of-the-art Bayesian and non-Bayesian learners. Model weighting is only one of the four research directions for refining AODE. The exploration of using weights for attribute selection or model selection will be an interesting topic in our future work.

APPENDIX
DETAILED EXPERIMENTAL RESULTS

See Tables 7–10

REFERENCES

[1] S. Ma and H. Shi, "Tree-augmented naive Bayes ensembles," in *Proc. 3rd ICML*, Shanghai, China, vol. 3, 2004, pp. 1497–1502.
[2] J. Pearl, *Causality: Models, Reasoning Inference*, vol. 29. Cambridge, U.K.: Cambridge Univ. Press, 2009.
[3] J. Pearl, *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Amsterdam, The Netherlands: Elsevier, 2014.
[4] L. Yu, L. Jiang, D. Wang, and L. Zhang, "Attribute value weighted average of one-dependence estimators," *Entropy*, vol. 19, no. 9, p. 501, Sep. 2017.
[5] P. Langley, W. Iba, and K. Thompson, "An analysis of Bayesian classifiers," in *Proc. AAAI*, vol. 90, 1992, pp. 223–228.
[6] L. Jiang, L. Zhang, L. Yu, and D. Wang, "Class-specific attribute weighted naive Bayes," *Pattern Recognit.*, vol. 88, pp. 321–330, Apr. 2019.
[7] H. Zhang, L. Jiang, and L. Yu, "Class-specific attribute value weighting for Naive Bayes," *Inf. Sci.*, vol. 508, pp. 260–274, Jan. 2020.
[8] L. Yu, L. Jiang, D. Wang, and L. Zhang, "Toward naive Bayes with attribute value weighting," *Neural Comput. Appl.*, vol. 31, no. 10, pp. 5699–5713, Oct. 2019.
[9] N. Friedman, D. Geiger, and M. Goldszmidt, "Bayesian network classifiers," *Mach. Learn.*, vol. 29, nos. 2–3, pp. 131–163, 1997.
[10] M. Sahami, "Learning limited dependence Bayesian classifiers," in *Proc. 2th SIGKDD*, Portland, OR, USA, 1996, vol. 96, no. 1, pp. 335–338.
[11] L. Jiang, H. Zhang, and Z. Cai, "A novel Bayes model: Hidden naive Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 21, no. 10, pp. 1361–1371, Oct. 2009.
[12] L. Jiang, L. Zhang, C. Li, and J. Wu, "A correlation-based feature weighting filter for naive Bayes," *IEEE Trans. Knowl. Data Eng.*, vol. 31, no. 2, pp. 201–213, Feb. 2019.
[13] Z. Zhou, "Ensemble learning," in *Encyclopedia of Biometrics*. Boston, MA, USA: Springer, 2015, pp. 411–416.
[14] Y. Zhao, Y. Chen, K. Tu, and J. Tian, "Learning Bayesian network structures under incremental construction curricula," *Neurocomputing*, vol. 258, pp. 30–40, Oct. 2017.
[15] G. I. Webb, J. R. Boughton, and Z. Wang, "Not so naive Bayes: Aggregating one-dependence estimators," *Mach. Learn.*, vol. 58, no. 1, pp. 5–24, Jan. 2005.
[16] S. Chen, A. M. Martinez, and G. I. Webb, "Highly scalable attribute selection for averaged one-dependence estimators," in *Proc. PAKDD*. Cham, Switzerland: Springer, 2014, pp. 86–97.
[17] L. Jiang, H. Zhang, Z. Cai, and D. Wang, "Weighted average of one-dependence estimators," *J. Exp. Theor. Artif. Intell.*, vol. 24, no. 2, pp. 219–230, 2012.
[18] Y. Yang, G. Webb, J. Cerquides, K. Korb, J. Boughton, and K. M. Ting, "To select or to weigh: A comparative study of model selection and model weighing for spode ensembles," in *Proc. ECML*. Berlin, Germany: Springer, 2006, pp. 533–544.
[19] G. F. Cooper and E. Herskovits, "A Bayesian method for constructing Bayesian belief networks from databases," in *Proc. UAI*. Amsterdam, The Netherlands: Elsevier, 1991, pp. 86–94.
[20] C. Bielza and P. Larrañaga, "Discrete Bayesian network classifiers: A survey," in *Proc. ACM Comput. Surv.*, vol. 47, no. 1, pp. 1–43, 2014.
[21] X. Zheng, Z. Lin, H. Xu, C. Chen, and T. Ye, "Efficient learning ensemble SuperParent-one-dependence estimator by maximizing conditional log likelihood," *Expert Syst. Appl.*, vol. 42, no. 21, pp. 7732–7745, Nov. 2015.
[22] G. I. Webb, J. R. Boughton, F. Zheng, K. M. Ting, and H. Salem, "Learning by extrapolation from marginal to full-multivariate probability distributions: Decreasingly naive Bayesian classification," *Mach. Learn.*, vol. 86, no. 2, pp. 233–272, Feb. 2012.
[23] D. Grossman, P. Domingos, and P. Domingos, "Learning Bayesian network classifiers by maximizing conditional likelihood," in *Proc. 21th ICML*, 2004, pp. 361–368.
[24] L. Jiang, C. Li, and Z. Cai, "Decision tree with better class probability estimation," *Int. J. Pattern Recogn. Artif. Intell.*, vol. 23, no. 4, pp. 745–763, Jun. 2009.
[25] A. P. Bradley, "The use of the area under the ROC curve in the evaluation of machine learning algorithms," *Pattern Recognit.*, vol. 30, no. 7, pp. 1145–1159, Jul. 1997.
[26] D. J. Hand and R. J. Till, "A simple generalisation of the area under the roc curve for multiple class classification problems," *Mach. Learn.*, vol. 45, no. 2, pp. *171–186*, 2001.
[27] F. Zheng, G. I. Webb, P. Suraweera, and L. Zhu, "Subsumption resolution: An efficient and effective technique for semi-naive Bayesian learning," *Mach. Learn.*, vol. 87, no. 1, pp. 93–125, Apr. 2012.
[28] C. E. Shannon, "A mathematical theory of communication," *Bell Syst. Tech. J.*, vol. 27, no. 3, pp. 379–423, 1948.
[29] L. Wang, G. Wang, Z. Duan, H. Lou, and M. Sun, "Optimizing the topology of Bayesian network classifiers by applying conditional entropy to mine causal relationships between attributes," *IEEE Access*, vol. 7, pp. 134271–134279, 2019.
[30] L. Wang, S. Chen, and M. Mammadov, "Target learning: A novel framework to mine significant dependencies for unlabeled data," in *Proc. PAKDD*. Cham, Switzerland: Springer, 2018, pp. 106–117.
[31] P. M. Murphy and D. W. Aha. *UCI Repository of Machine Learning Databases*. Accessed: Jun. 11, 2019. [Online]. Available: http://www.ics.uci.edu/mlearn/MLRepository.html
[32] T. Roos, H. Wettig, P. Grünwald, P. Myllymäki, and H. Tirri, "On discriminative Bayesian network classifiers and logistic regression," *Mach. Learn.*, vol. 59, no. 3, pp. 267–296, 2005.
[33] C. Cortes and V. Vapnik, "Support-vector networks," *Mach. Learn.*, vol. 20, no. 3, pp. 273–297, 1995.
[34] N. S. Altman, "An introduction to kernel and nearest-neighbor nonparametric regression," *Amer. Statistician*, vol. 46, no. 3, pp. 175–185, Aug. 1992.
[35] U. Fayyad and K. Irani, "Multi-interval discretization of continuous-valued attributes for classification learning," in *Proc. 13th IJCAI*, Chambery, France, 1993, pp. 1022–1029.
[36] R. Kohavi and D. H. Wolpert, "Bias plus variance decomposition for zero-one loss functions," in *Proc. 13th ICML*, Bari, Italy, 1996, pp. 275–283.
[37] G. I. Webb, "Multiboosting: A technique for combining boosting and wagging," *Mach. Learn.*, vol. 40, no. 2, pp. *159–196*, 2000.
[38] D. Brain and G. Webb, "On the effect of data set size on bias and variance in classification learning," in *Proc. 4th Austral. Knowl. Acquisition Workshop*, 1999, pp. 117–128.
[39] M. Friedman, "The use of ranks to avoid the assumption of normality implicit in the analysis of variance," *J. Amer. Stat. Assoc.*, vol. 32, no. 200, pp. 675–701, Dec. 1937.
[40] P. Nemenyi, "Distribution-free multiple comparisons," Ph.D. dissertation, Dept. Statist., Princeton Univ., Princeton, NJ, USA, 1963.
[41] J. Demšar, "Statistical comparisons of classifiers over multiple data sets," *J. Mach. Learn. Res.*, vol. 7, no. 1, pp. 1–30, Jan. 2006.



LIMIN WANG received the Ph.D. degree in computer science from Jilin University, China, in 2005. He is currently a Professor with the College of Computer Science and Technology, Jilin University. He has authored or coauthored more than 60 academic articles in reputed peer-reviewed international journals and conferences. His research interests include machine learning, data mining, decision making, and Bayesian networks. He has supervised many M.S. and Ph.D. students in the above-mentioned fields. He has also been involved with reviewing and organizing different workshops, seminars, and training sessions on different technologies.

JIE CHEN received the B.S. degree from Jilin University, China, in 2018, where she is currently pursuing the master's degree with the College of Computer Science and Technology. Her research interests include Bayesian networks and data analysis.

YANG LIU received the M.S. degree from Jilin University, China, in 2018, where he is currently pursuing the Ph.D. degree with the College of Computer Science and Technology. His research interests include data mining and Bayesian networks.

MINGHUI SUN received the Ph.D. degree in computer science from the Kochi University of Technology, Japan, in 2011. He is currently an Assistant Professor with the College of Computer Science and Technology, Jilin University, China. He is interested in using HCI methods to solve challenging real world computing problems in many areas, including tactile interface, pen-based interface, and tangible interface.