Learning Bayesian Network Classifiers to Minimize Class Variable Parameters

Anonymous submission

Abstract

This study proposes and evaluates a new Bayesian network classifier (BNC) having an I-map structure with the fewest class variable parameters among all structures for which the class variable has no parent. Moreover, a new learning algorithm to learn our proposed model is presented. The proposed method is guaranteed to obtain the true classification probability asymptotically. Moreover, the method has lower computational costs than those of exact learning BNC using marginal likelihood. Comparison experiments have demonstrated the superior performance of the proposed method.

Introduction

Classifiers are divided roughly into deep learning approaches and probabilistic approaches. The former can learn huge variables and can provide more accurate classification than the latter does. However, the decisions made by the deep learning classifier is unexplainable because mathematical properties of the deep learning are unclear and because the decisions of deep learning classifiers are deterministic. By contrast, the latter has explainability of decisions because it can estimate a probability distribution of the class variable.

A popular probabilistic classifier is the naive Bayes classifier, in which the feature variables are conditionally independent given a class variable (Minsky 1961). Initially, because actual datasets were generated from more complex systems, naive Bayes was not expected to provide highly accurate classification. Therefore, the general Bayesian network (GBN) with learning by marginal likelihood (ML) as a generative model has been expected to outperform naive Bayes because GBN is more expressive than naive Bayes. However, Friedman, Geiger, and Goldszmidt (1997) demonstrated that naive Bayes sometimes outperforms the GBN using a greedy search to find the smallest minimum description length (MDL) score, which had been originally intended to approximate ML. They explained the inferior performance of the MDL by decomposing it into a log likelihood (LL) term that reflects the model fitting to training data, and a penalty term that reflects the model complexity. Moreover, they decomposed the LL term into a conditional log likelihood (CLL) of the class variable given the feature variables, which is related directly to the classification, and into a joint LL of the feature variables, which is not related directly to the classification. Furthermore, they proposed conditional MDL (CMDL), a modified MDL replacing the LL with the CLL.

Consequently, they claimed that the Bayesian network (BN) minimizing CMDL, as a discriminative model, shows better accuracy than that maximizing ML. Unfortunately, CLL has no closed-form equation for estimating the optimal parameters. This finding implies that optimizing CLL requires a gradient descent algorithm (e.g., extended logistic regression algorithm (Greiner and Zhou 2002)). Nevertheless, the optimization algorithm involves the reiteration of each structure candidate, which greatly increases the computational costs.

To resolve this difficulty, Friedman, Geiger, and Goldszmidt (1997) proposed an augmented naive Bayes classifier (ANB) in which the class variable links directly to all feature variables. Links among feature variables are allowed. Actually, ANB ensures that all feature variables can contribute to classification. Later, restricted ANBs of various types were proposed, such as tree-augmented naive Bayes (TAN) (Friedman, Geiger, and Goldszmidt 1997) and forest-augmented naive Bayes (FAN) (Lucas 2004).

Maximization of CLL entails heavy computation. Therefore, various approximation methods have been proposed to maximize it. Carvalho, Adão, and Mateus (2013) proposed *approximate CLL* (aCLL), which is decomposable and which is computationally efficient. Moreover, Grossman and Domingos (2004) proposed a learning structure method using a greedy hill-climbing algorithm (Heckerman, Geiger, and Chickering 1995) to maximize CLL. Furthermore, Mihaljević, Bielza, and Larrañaga (2018) proposed a method to reduce the space for the greedy search of BN Classifiers (BNCs) with the CLL score. These reports described that the BNC maximizing the approximated CLL performed better than that maximizing the approximated ML.

Nevertheless, they did not explain why CLL outperformed ML. For large datasets, because ML has an asymptotic consistency, the classification accuracies presented by maximizing ML are expected to be comparable to those presented by maximizing CLL. Differences between the performances of the two scores in these studies might depend on their respective learning algorithms: they were approximate learning algorithms, not exact ones. Recent studies have explored efficient algorithms for the exact learning of GBN to maximize ML (Koivisto and Sood 2004; Singth and Moore

2005; Silander and Myllymäki 2006; De Campos and Ji 2011; Malone et al. 2011; Yuan and Malone 2013; Cussens 2012; Barlett and Cussens 2013; Suzuki 2017).

Sugahara, Uto, and Ueno (2018) compared the classification performances of BNC with exact learning using ML as a generative model and those with approximate learning using CLL as a discriminative model. Results show that maximizing ML leads to better classification accuracy than maximizing CLL provides for large datasets. However, the results also indicate that classification accuracies obtained by exact learning BNC using ML are much worse than those obtained using other methods when the sample is small and the class variable has numerous parents in the exactly learned networks. When a class variable has numerous parents, estimation of the conditional probability parameters of the class variable becomes unstable because the parent configurations become numerous and because the sample for learning the parameters becomes sparse.

To resolve this difficulty, Sugahara, Uto, and Ueno (2018) proposed an exact learning ANB, which maximizes ML and which ensures that the class variable has no parents. In earlier studies, the ANB constraint was used to learn the BNC as a discriminative model. In contrast, they use the ANB constraint to learn the BNC as a generative model. Their method asymptotically learns the optimal ANB, which is an independence map (I-map) of the true probability distribution with the fewest parameters among all possible ANB structures. Moreover, Sugahara, Uto, and Ueno (2018) proved that exactly learned ANB is asymptotically *classification equivalent* to the true structure. The ANB guarantees asymptotic estimation of the identical posterior of the class variable to that of the true structure. However, the ANB maximizing ML has the following three shortcomings. (1) ANB might include irrelevant feature variables with the class variable. These variables are known to often lower classification accuracy because they only introduce noise in the classification. (2) ANB maximizing ML guarantees an asymptotic classification equivalence to the true structure when the true structure does not follow ANB. (3) ANB structure maximizing ML is not guaranteed to converge an I-map minimizing the number of parameters for estimating the probabilities of the class variable but to that minimizing the number of all the parameters.

This study proposes a learning I-map BNC with the fewest class variable parameters among all the structures in which the class variable has no parents (we designate them as NPCDAGs (no parent class DAGs)). The proposed learning BNC has the following three benefits. (1) The proposed learning BNC asymptotically does not include irrelevant feature variables for the class variable because NPCDAGs do not force the addition of edges between the class variable and feature variables. (2) The proposed learning BNC achieves an asymptotic classification equivalence to the true structure, even when the true structure does not follow ANB. (3) The proposed learning BNC asymptotically produces an I-map NPCDAG with the lowest number of class variable parameters (NCP).

To learn the proposed BNC, we prove that maximizing the ML asymptotically produces an I-map BNC with the lowest NCP only when given a variable order. Based on this theorem, we formulate learning of the proposed BNC as a shortest-path-finding problem for an order graph of BNs. Popular methods to solve shortest path finding problems are breadth-first search and depth-first search. Their computational time increases exponentially as the number of variables increases. We can decrease their computational costs using branch and bound method. However, the traditional heuristic functions used for cutting-edge branch and bound method can not apply our method to learn the proposed BNC.

We prove that the NCP of a naive Bayes classifier is the lower bound NCP of our model. As an algorithm for learning our model, we propose a depth-first branch and bound algorithm using the NCP of a naive Bayes classifier as a heuristic function. The proposed algorithm provides shorter runtime than that provided by the traditional learning BNC algorithm. Moreover, the proposed algorithm can be stopped at any time. We can obtain the current best solution. Comparison experiments have demonstrated the superior performance of the proposed method.

Background

Bayesian networks

A BN is a graphical model that represents conditional independence among random variables as a directed acyclic graph (DAG). The BN decomposes the joint probability distribution exactly into a product of the conditional probability for each variable.

Letting $\mathbf{V} = \{X_0, X_1, \cdots, X_n\}$ be a set of discrete variables, where $X_i, (i = 0, \cdots, n)$ can take values in the set of states $\{1, \cdots, r_i\}$, then we write $X_i = k$ when X_i takes the state k. According to the BN structure G, the joint probability distribution is represented as $P(X_0, X_1, \cdots, X_n \mid G) = \prod_{i=0}^{n} P(X_i \mid \mathbf{Pa}_{X_i}^G, G)$, where $\mathbf{Pa}_{X_i}^G$ is the parent variable set of X_i in G. When structure G might be readily apparent from the context, we use $\mathbf{Pa}_i$ to denote the parents.

Letting σ be a variable order in $\mathbf{V}$, e.g. $\sigma = (X_4, X_1, X_2, X_3)$, the BN structure G is said to be consistent with order σ when all parents of the variable precede the variable in the order.

Let θ_{ijk} be a conditional probability parameter of $X_i = k$ when the j-th instance of the parents of X_i is observed (we can say $\mathbf{Pa}_i = j$). Then we define $\Theta_{ij} = \bigcup_{k=1}^{r_i} \{\theta_{ijk}\}$, $\Theta_i = \bigcup_{j=1}^{q_i} \{\Theta_{ij}\}$, and $\Theta = \bigcup_{i=0}^{n} \Theta_i$, where $q_i = \prod_{v:X_v \in \mathbf{Pa}_i} r_v$. A BN is defined using a pair $B = (G, \Theta)$.

The BN structure G represents conditional independence assertions in the probability distribution by d-separation. First, we must define *collider* to define the d-separation. Letting *path* denote a sequence of adjacent variables, then the collider is defined as

Definition 1 *Assuming a structure $G = (\mathbf{V}, \mathbf{E})$, a variable $Z \in \mathbf{V}$ on a path ρ is a collider if and only if there exist two distinct incoming edges into Z from non-adjacent variables.*

The d-separation is defined as presented below.

Definition 2 *Assuming a structure $G = (\mathbf{V}, \mathbf{E})$, $X, Y \in \mathbf{V}$, and $\mathbf{U} \subseteq \mathbf{V} \setminus \{X, Y\}$, the two variables X and Y are d-*

separated, given $\mathbf{U}$ *in* G*, if and only if every path* ρ *between* X *and* Y *satisfies either of the following two conditions: (1)* $\mathbf{U}$ *includes a non-collider on* ρ*. (2) There is a collider* Z *on* ρ*;* $\mathbf{U}$ *does not include* Z *and its descendants. We denote the* d*-separation between* X *and* Y *given* $\mathbf{U}$ *in the structure* G *as* $Dsep_G(X, Y \mid \mathbf{U})$.

$I(X, Y \mid \mathbf{U})$ denote that X and Y are conditionally independent given $\mathbf{U}$ in the true joint probability distribution P^*. A BN structure G is an *independence map (I-map)* if all the d-separations in G entail conditional independence in P^*.

Definition 3 *Assuming the true joint probability distribution* P^* *of the random variables in a set* $\mathbf{V}$ *and a structure* $G = (\mathbf{V}, \mathbf{E})$*, then* G *is an I-map if the following proposition holds:* $\forall X, Y \in \mathbf{V}, \forall \mathbf{U} \subseteq \mathbf{V} \setminus \{X, Y\}, (Dsep_G(X, Y \mid \mathbf{U}) \Rightarrow I(X, Y \mid \mathbf{U}))$.

We introduce the following notation, which is necessary for our discussion of learning BNs. Let $D = \{\mathbf{x}^1, \cdots, \mathbf{x}^d, \cdots, \mathbf{x}^N\}$ be a complete dataset consisting of N i.i.d. instances, where each instance $\mathbf{x}^d$ is a data vector $(x_0^d, x_1^d, \cdots, x_n^d)$. The likelihood of BN $B = (G, \Theta)$, given D, is $P(D \mid B) = \prod_{i=0}^{n} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \theta_{ijk}^{N_{ijk}}$. The maximum likelihood estimators of θ_{ijk} are given as $\hat{\theta}_{ijk} = N_{ijk}/N_{ij}$. In that equation, N_{ijk} represents the number of samples of $X_i = k$ when $\mathbf{Pa}_i = j$, $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$. The most popular parameter estimator of BNs is the *expected a posteriori* (EAP), which is the expectation of θ_{ijk} with respect to the posterior of Θ_{ij} assuming the Dirichlet prior of Θ_{ij}. The EAP estimator is represented as $\hat{\theta}_{ijk} = (N'_{ijk} + N_{ijk})/(N'_{ij} + N_{ij})$, where N'_{ijk} denotes the hyperparameters of the Dirichlet prior distributions. In addition, (N'_{ijk} is a pseudo-sample corresponding to N_{ijk}), with $N'_{ij} = \sum_{k=1}^{r_i} N'_{ijk}$. The BN structure G must be estimated from the observed datasets because it is generally unknown. We maximize the score with an *asymptotic consistency* which guarantees estimation of an I-map with the fewest parameters. The ML score has an asymptotic consistency (Chickering 2002).

When we assume the Dirichlet prior for Θ_{ij}, ML is represented as $P(D \mid G) = \prod_{i=0}^{n} \prod_{j=1}^{q_i} \Gamma(N'_{ij})/\Gamma(N'_{ij} + N_{ij}) \prod_{k=1}^{r_i} \Gamma(N'_{ijk} + N_{ijk})/\Gamma(N'_{ijk})$. When $N'_{ijk} = N'/(r_i q_i)$, this score is called the *Bayesian Dirichlet equivalent uniform* (BDeu) (1995), where N' is the equivalent sample size (ESS) determined by users.

Bayesian network classifiers

A BNC can be interpreted as a BN for which X_0 is the class variable and for which $X_1, \cdots, X_n$ are feature variables. Given an instance $\mathbf{x} = (x_1, \cdots, x_n)$ for feature variables $X_1, \ldots, X_n$, the BNC B predicts the class variable by maximizing the following posterior probability of the class variable X_0.

$$P(X_0 = c \mid x_1, \cdots, x_n, B) \tag{1}$$
$$= \frac{\prod_{j=1}^{q_0} \prod_{k=1}^{r_0} (\theta_{0jk})^{1_{0jk}} \prod_{i:X_0 \in \mathbf{Pa}_i} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{ijk})^{1_{ijk}}}{\sum_{c'=1}^{r_0} \prod_{j=1}^{q_0} \prod_{k=1}^{r_0} (\theta_{0jk})^{1_{0jk}} \prod_{i:X_0 \in \mathbf{Pa}_i} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} (\theta_{ijk})^{1_{ijk}}},$$

In those equations, $1_{ijk} = 1$ if $X_i = k$ and $\mathbf{Pa}_i = j$ in the case of $\mathbf{x}$, and $1_{ijk} = 0$ otherwise. From Equation (1), we can infer class c given only the values of the parents of X_0, the children of X_0, and the parents of the children of X_0, which comprise the *Markov blanket* of X_0.

Sugahara, Uto, and Ueno (2018) proved that, under the following Assumptions 1 and 2, the ANB guarantees asymptotic estimation of the true probability of the class variable as described below.

Definition 4 *(Acid, De Campos, and Castellano 2005) Letting* $\mathcal{G}$ *be a set of all the BN structures and letting* D *be any finite dataset, then* $\forall G_1, G_2 \in \mathcal{G}$*, we say that* G_1 *and* G_2 *are classification-equivalent if* $P(X_0 \mid \mathbf{x}, G_1, D) = P(X_0 \mid \mathbf{x}, G_2, D)$ *for any feature variable's value* $\mathbf{x}$.

Assumption 1 *A true structure* $G^* = (\mathbf{V}, \mathbf{E}^*)$ *with the following property exists:* $\forall X, Y \in \mathbf{V}, \forall \mathbf{U} \subseteq \mathbf{V} \setminus \{X, Y\}, (Dsep_{G^*}(X, Y \mid \mathbf{U}) \Leftrightarrow I(X, Y \mid \mathbf{U}))$.

Assumption 2 *For* $\forall X \in \mathbf{V}$*,* X *and* X_0 *are adjacent to* G^*.

Theorem 1 *(Sugahara, Uto, and Ueno 2018) Under Assumptions 1 and 2, for a sufficiently large sample size, the exact learning ANB using BDeu achieves the classification-equivalent structure to* G^*.

Learning BNCs with the Minimum Number of Class Variable Parameters

From Theorem 1, the ANB maximizing BDeu achieves the true posterior of the class variable asymptotically. From Equation (1), the posterior of the class variable depends only on a set of parameters Θ_0 and $\bigcup_{i:X_0 \in \mathbf{Pa}_i} \Theta_i$, which we call designate as class variable parameters. Letting G be a structure, then the NCP in G is defined as $NCP(G) = \sum_{i=0}^{n} NCP_i(\mathbf{Pa}_i)$. Therein, $NCP_i(\mathbf{Pa}_i) = (r_i - 1)q_i$ if $i = 0 \vee X_0 \in \mathbf{Pa}_i$, and $NCP_i(\mathbf{Pa}_i) = 0$ otherwise. However, maximizing BDeu entails no guarantee of minimizing the NCP.

This study proposes a new learning BNC method that guarantees asymptotic estimation of an I-map with the lowest NCP. Our search space is a set of structures in which the class variable has no parents, which we designate as NPCDAGs. The following benefits are gained by introducing the limited search space. First, this space is guaranteed to cover all possible posteriors of the class variable (Mihaljević, Bielza, and Larrañaga 2018). Second, teaching BNCs to have no parents of the class variable is expected to provide more accurate classification than teaching GBNs does. The reasons are the following. The prior of the class variable of GBN dominates the posterior when the parents become numerous. This dominance hinders the posterior from fully reflecting the likelihood given the obtained data. In addition, the prior determined solely by the limited number of parent variables often leads to extremely unstable results. In contrast, the proposed method teaches the posterior to reflect the likelihood of all class variable parameters. This reflected likelihood enhances the process to make the most of all related variable data to learn the posterior. In addition, the proposed method employs a marginal probability estimate of the class variable, which is stably obtained, as a prior.

We derive the following theorem to present the advantage of the proposed learning BNC.

Theorem 2 *Under Assumption 1, I-maps NPCDAG with the lowest NCP asymptotically converge to classification equivalent structures to G^*.*

Supplementary materials include a proof of Theorem 2. This theorem states that our proposed BNC has the same classification performance, asymptotically, as that of the true model. ANB maximizing BDeu does not guarantee an asymptotic classification equivalence to the true structure when the true structure does not follow ANB. As inferred from Theorem 2, the proposed learning BNC achieves an asymptotic classification equivalence to the true structure even when the true structure does not follow ANB.

However, the traditional algorithms maximizing BDeu do not guarantee that one can obtain an I-map with the lowest NCP. To introduce the proposed method, the following theorem is proved.

Theorem 3 *When the sample is sufficiently large, the highest BDeu scoring structure consistent with an order σ is an I-map with the lowest NCP among all structures consistent with σ.*

Supplementary materials provide a proof of Theorem 3. According to Theorem 3, the set consisting of the BDeu-largest structure for each variable order includes the NCP-smallest structure. This fact suggests the following algorithm. (1) Given each variable order, obtain the highest BDeu structure. (2) Obtain a structure minimizing NCP among the structures obtained in (1).

Before presenting details of the procedure necessary for our method, we introduce the following notation. First, we let $\mathbf{Pre}_X^\sigma$ denote a set of preceding variables to X in a variable order σ, and let G_σ^* denote the highest BDeu scoring structure consistent with σ. Additionally, we define the *best parents* of X_i in a candidate set $\mathbf{U}$ as the parent set which maximizes the local score in $\mathbf{U}$: $g_i^*(\mathbf{U}) = \arg\max_{\mathbf{W}\subseteq\mathbf{U}} Score_i(\mathbf{W})$. Moreover, we describe a structure G as a vector $G = (\mathbf{Pa}_0, \mathbf{Pa}_1, \cdots, \mathbf{Pa}_n)$ of parent sets: $\mathbf{Pa}_i$ is the subset of $\mathbf{V}$ from which there are edges to X_i. We let $\boldsymbol{\sigma}_0(\mathbf{U})$ denote a set of variable orders for $\mathbf{U}$ in which the first element is X_0. Also, we let $G^*(\mathbf{U})$ denote a structure which minimizes NCP among all structures composed of $\mathbf{U}$, consistent with orders in $\boldsymbol{\sigma}_0(\mathbf{U})$. When a variable has no child in a structure, it is a *sink* in the structure. We use $X_s^*(\mathbf{U})$ to denote a sink in $G^*(\mathbf{U})$.

Learning the proposed BNC is the shortest path to finding a problem for NPC reverse order graph (NROG), which is a directed graph consisting of nodes corresponding to elements of $2^{\mathbf{V}} \setminus 2^{\mathbf{V}\setminus\{X_0\}}$. For a variable $X_i \in \mathbf{V}$ and a variable set $\mathbf{U} \subseteq \mathbf{V}$, NROG has an edge from $\mathbf{U}$ to $\mathbf{U}\setminus\{X_i\}$. An example of NROG for $\mathbf{V} = \{X_0, X_1, X_2, X_3\}$ is presented in Figure 1. An edge from $\mathbf{U}$ to $\mathbf{U}\setminus\{X_i\}$ represents that a sink in $G^*(\mathbf{U})$ is X_i and that the parents of X_i are $g_i^*(\mathbf{U}\setminus\{X_i\})$. Moreover, the edge from $\mathbf{U}$ to $\mathbf{U}\setminus\{X_i\}$ has a cost $NCP_i(g_i^*(\mathbf{U}\setminus\{X_i\}))$. Each path from $\mathbf{V}$ to X_0 in NROG corresponds to each variable order in $\sigma_0(\mathbf{V})$. For $\sigma_0 \in \sigma_0(\mathbf{V})$, one can obtain $G_{\sigma_0}^*(\mathbf{V})$ by following the path corresponding to σ_0. The cost $c(p)$ of path p corresponding σ_0 is defined as

$$c(p) = \sum_{i=1}^{n} NCP_i(g_i^*(\mathbf{Pre}_{X_i}^{\sigma_0})) + r_0 - 1 = NCP(G_{\sigma_0}^*(\mathbf{V})).$$

By finding the shortest path p^* in which $c(p^*) \leq c(p)$ for any path p, one can obtain $G^*(\mathbf{V})$.

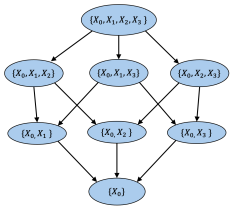

Figure 1: NPC reverse order graph (NROG) of four variables.

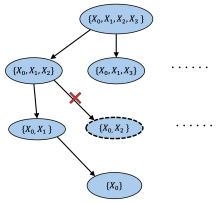

Figure 2: Running example of the depth-first branch and bound algorithm.

Popular methods to solve the shortest path finding problems are breadth-first search and depth-first search. Their computational times increase exponentially as the number of variables increases. Their computational cost can be decreased using branch and bound method. For a variable set $\mathbf{U} \subseteq \mathbf{V}$, we define $g(\mathbf{U})$ as the cost of a path from $\mathbf{V}$ to $\mathbf{U}$, and define $h(\mathbf{U})$ as the lower bound of the cost of the path from $\mathbf{U}$ to $\{X_0\}$. We use $f(\mathbf{U}) = g(\mathbf{U}) + h(\mathbf{U})$ for the cutting edges of NROG. When $f(\mathbf{U})$ is higher than a cost of the current best solution, we cut the node $\mathbf{U}$ because any path which passes $\mathbf{U}$ is not the best path.

Figure 2 depicts an example of depth-first search using branch and bound method for $\mathbf{V} = \{X_0, X_1, X_2, X_3\}$. After expanding $\{X_0, X_1, X_2, X_3\}$ and searching $\{X_0, X_1, X_2\}$, we expand $\{X_0, X_1, X_2\}$ and search $\{X_0, X_1\}$. Then we expand $\{X_0, X_1\}$ and search $\{X_0\}$. The resulting structure is an I-map with the lowest NCP among all structures, consistent with the variable order (X_0, X_1, X_2, X_3). The current best solution is updated to the NCP of this structure. Next, we expand $\{X_0, X_2\}$ if $f(\{X_0, X_2\})$ is lower than the current best solution, or cut $\{X_0, X_2\}$ otherwise. Then, we expand $\{X_0, X_1, X_3\}$ if $f(\{X_0, X_1, X_3\})$ is lower than the current best solution, or otherwise cut $\{X_0, X_1, X_3\}$, and so on.

However, traditional heuristic functions used as $h(\mathbf{U})$ are not applicable for our method to learn the proposed BNC. We prove the following theorem and propose a new heuristic function for our learning algorithm.

Theorem 4 *For any variable set $\mathbf{V}$, let $G^*(\mathbf{V})$ be an I-map with the lowest NCP, and let $G^{NB}(\mathbf{V})$ be the naive Bayes classifiers consisting of a set of feature variables $\mathbf{V}_c$, which are children of the class variable in $G^*(\mathbf{V})$. Then, $NCP(G^{NB}(\mathbf{V}_c)) \leq NCP(G^*(\mathbf{V}))$ holds.*

Supplementary materials present a proof of Theorem 4. This theorem states that one can predict the lower bound of NCP of $G^*(\mathbf{V})$ given $\mathbf{V}_c$. We propose the heuristic function

$h^*(\mathbf{U}) = \sum_{X_i \in (\mathbf{U} \cup \mathbf{V_c})} NCP_i(X_0)$. The proposed heuristic function has consistency. A heuristic function with the consistency guarantees that one can find the shortest path.

Definition 5 *For any node* $\mathbf{U}$ *and* $\mathbf{R}$ *in which there is an edge from* $\mathbf{U}$ *to* $\mathbf{R}$ *in an NROG, the heuristic function* $h(\mathbf{U})$ *has a consistency if and only if* $h(\mathbf{U}) \leq h(\mathbf{R}) + c(\mathbf{U}, \mathbf{R})$ *holds, and where* $c(\mathbf{U}, \mathbf{R})$ *is the cost of edge from* $\mathbf{U}$ *to* $\mathbf{R}$.

Theorem 5 h^* *has a consistency.*

Supplementary materials include the proof of Theorem 5.

However, we do not obtain $\mathbf{V}_c$ before learning structures. The proposed method predicts the $\mathbf{V}_c$ using feature selection with Bayes factor. Sugahara, Uto, and Ueno (2018) proposed a method for learning children of the class variable using the Bayes factor. The Bayes factor is a ratio between BDeu of independence model g_1 and that of dependence model g_2, i.e., $BF(X_0, X_i) = BDeu(g_1)/BDeu(g_2)$. Sugahara, Uto, and Ueno (2018) determines the independence between X_0 and X_i when the Bayes factor is lower than a threshold. Bayes factor with BDeu guarantees prediction of the true children of the class variable when the sample size is sufficiently large. We employ the feature selection method proposed by Sugahara, Uto, and Ueno (2018) to obtain $\mathbf{V}_c$ used in the proposed heuristic function.

Experiments

This section describes experiments conducted to demonstrate the benefits of the proposed method.

Benchmark datasets

First, we compare the classification accuracies of the following ten methods using the benchmark datasets in Table 1. (1) *Naive Bayes* (2) *GBN-CMDL* (Grossman and Domingos 2004): Greedy learning GBN method using the hill-climbing search by minimizing CMDL while estimating parameters by maximizing LL (3) *BNC2P* (Grossman and Domingos 2004): Greedy learning method with at most two parents per variable using the hill-climbing search by maximizing CLL while estimating parameters by maximizing LL (4) *TAN-aCLL* (Carvalho, Adão, and Mateus 2013): Exact learning TAN method by maximizing aCLL (5) *MC-DAGGES* (Mihaljević, Bielza, and Larrañaga 2018): Greedy learning method in the space of the Markov equivalent classes of MC-DAGs using the greedy equivalence search (Chickering 2002) by maximizing CLL while estimating parameters by maximizing LL (6) *GBN-BDeu*: Exact learning GBN method by maximizing BDeu (7) *ANB-BDeu*: Exact learning ANB method by maximizing BDeu (8) *fsANB-BDeu*: Exact learning ANB method by maximizing BDeu with feature selection by Bayes factor (Sugahara, Uto, and Ueno 2018) (9) *Proposed(BFS)*: Learning the proposed BNC using the breadth-first search to the shortest path finding problems of NROG (10) *Proposed(DFB&B)*: Learning the proposed BNC using the proposed depth-first branch and bound algorithm to the shortest path finding problems of NROG

We ascertain the ESS $N' \in \{1, 10, 100, 1,000\}$ of BDeu scores in *GBN-BDeu*, *ANB-BDeu*, *fsANB-BDeu*, *Proposed(BFS)*, and *Proposed(DFB&B)* using ten-fold cross validation to obtain the highest classification accuracy. The proposed methods are implemented in C++. The other methods are implemented in Java. To ensure double-blind review, the code will be made publicly available after the manuscript is reviewed and accepted for publication. As described throughout this paper, our experiments are conducted on a computer with a 3.2 GHz 16-core processor and 128 GB of memory. This experiment uses 24 real datasets from the *UCI repository* (Lichman 2013). Continuous variables are discretized using a standard discretization algorithm proposed by (Fayyad and Irani 1993). In addition, data with missing values are removed from the datasets. We use EAP estimators with $N'_{ijk} = 1/(r_i q_i)$ as conditional probability parameters of the respective classifiers.

Table 1 presents the classification accuracies of the respective classifiers. The datasets presented in Table 1 are listed in ascending order of sample per pattern (SPP), which is the sample size divided by the number of all possible patterns of the variables' values. Here, the classification accuracies represent the average percentage of correct classifications from ten-fold cross-validation. To confirm the significant differences of *Proposed(BFS)* and *Proposed(DFB&B)* from the other methods, we apply Hommel's tests (Hommel 1988), which are used as a standard in machine learning studies (Demšar 2006). The p-values are presented at the bottom of Table 1. Table 2 presents the NCPs of structures learned using the respective methods. Table 3 presents the average runtimes of the respective methods. Moreover, "NTCE" in Table 3 presents the numbers of times *Proposed(DFB&B)* cuts the edges of NROGs.

The results presented in Table 1 show that *Proposed(BFS)* outperforms other methods at the $p < 0.1$ significance level, except for *fsANB-BDeu*. Moreover, *Proposed(DFB&B)* outperforms *Naive Bayes*, *GBN-CMDL*, *MC-DAGGES*, and *GBN-BDeu* at the $p < 0.1$ significance level. For large SPP such as datasets Nos. 20 and 24, the accuracies of *Naive Bayes*, *BNC2P*, and *TAN-aCLL* tend to be worse than those of *Proposed(BFS)* and *Proposed(DFB&B)* because of the upper bound of the maximum number of parents. The small upper bound of the maximum number of parents tends to engender a poor representational power of the structure (Ling and Zhang 2003). For large SPPs such as Nos. 3 and 6, *GBN-CMDL* and *MC-DAGGES* have lower accuracy than *Proposed(BFS)* does because the exact learning methods estimate the network structures more precisely than the greedy learning methods do. *Proposed(BFS)* and *Proposed(DFB&B)* provide much higher accuracies than *GBN-BDeu* and *ANB-BDeu* do for small SPP, such as dataset No. 8. Table 2 shows that *Proposed(BFS)* and *Proposed(DFB&B)* provide smaller NCPs than *GBN-BDeu* and *ANB-BDeu* do in the dataset because *Proposed(BFS)* and *Proposed(DFB&B)* directly minimizes NCP, but *GBN-BDeu* and *ANB-BDeu* minimize the numbers of all parameters. Consequently, *Proposed(BFS)* and *Proposed(DFB&B)* can avoid overfitting to the datasets. That feature improves the classification accuracies for small SPP. The classification accuracies of *Proposed(BFS)* and *Proposed(DFB&B)* are almost identical to those of *GBN-BDeu*, *ANB-BDeu*, and *fsANB-BDeu* for large SPP. Especially, our findings indi-

No.	Dataset	Variables	Sample size	SPP	Naive-Bayes	GBN-CMDL	BNC2P	TAN-aCLL	MC-DAG GES	GBN-BDeu	ANB-BDeu	fsANB-BDeu	Proposed (BFS)	Proposed (DFB&B)
1	Image Segmentation	19	2310	9.41×10^{-15}	0.9286	0.7994	0.9481	0.9290	0.9286	0.9390	0.9468	0.9468	0.9550	0.9558
2	Pendigits	17	10992	1.13×10^{-13}	0.8805	0.8194	0.9541	0.9630	0.8898	0.9641	0.9636	0.9636	0.9601	0.9609
3	Letter	17	20000	9.32×10^{-13}	0.7384	0.4729	0.8444	0.8142	0.7810	0.8350	0.8454	0.8435	0.8608	0.8616
4	Lymphography	19	148	1.63×10^{-7}	0.8446	0.7230	0.7973	0.8311	0.8176	0.7500	0.7770	0.7838	0.8041	0.7905
5	EEG	15	14980	2.17×10^{-7}	0.6874	0.7592	0.7304	0.7197	0.7166	0.7308	0.7644	0.7644	0.7304	0.7285
6	Breast Cancer Wisconsin	10	683	3.42×10^{-7}	0.9751	0.8843	0.9590	0.9488	0.9722	0.9751	0.9751	0.9751	0.9751	0.9751
7	Zoo	17	101	7.34×10^{-5}	0.9406	0.7921	0.9406	0.9307	0.9406	0.9307	0.9505	0.9604	0.9505	0.9307
8	Hepatitis	20	80	7.63×10^{-5}	0.8500	0.8000	0.8875	0.8750	0.8500	0.6125	0.5750	0.8375	0.7875	0.8000
9	Wine	14	178	1.19×10^{-4}	0.9831	0.9494	0.9888	0.9775	0.9831	0.9775	0.9663	0.9663	0.9775	0.9775
10	Australian	15	690	2.23×10^{-4}	0.8464	0.8261	0.8348	0.8522	0.8638	0.8507	0.8420	0.8478	0.8551	0.8507
11	Vehicle	19	846	8.07×10^{-4}	0.4350	0.6123	0.5922	0.5816	0.5378	0.5768	0.6253	0.6135	0.6019	0.5827
12	Breast Cancer	10	277	8.33×10^{-4}	0.7401	0.6173	0.6895	0.7184	0.6282	0.7184	0.7040	0.7473	0.7401	0.7401
13	Heart	14	270	1.22×10^{-3}	0.8444	0.8333	0.7963	0.8407	0.8111	0.8074	0.8407	0.8370	0.8074	0.8074
14	HTRU2	9	17898	1.56×10^{-3}	0.9689	0.9668	0.9796	0.9764	0.9759	0.9787	0.9779	0.9779	0.9783	0.9784
15	Congressional Voting Records	17	232	1.77×10^{-3}	0.9095	0.9698	0.9741	0.9181	0.9052	0.9655	0.9483	0.9307	0.9655	0.9698
16	Solar Flare	11	1389	3.72×10^{-3}	0.7811	0.8243	0.8186	0.8229	0.7927	0.8431	0.8229	0.8373	0.8431	0.8431
17	Glass	10	214	6.63×10^{-3}	0.5561	0.5607	0.6028	0.6308	0.5467	0.5701	0.6449	0.5911	0.6262	0.6075
18	Contraceptive Method Choice	10	1473	2.66×10^{-2}	0.4671	0.4542	0.4630	0.4705	0.4650	0.4542	0.4481	0.4610	0.4623	0.4467
19	Hayes–Roth	5	132	2.29×10^{-1}	0.8182	0.6136	0.6515	0.6742	0.5909	0.6212	0.7879	0.7652	0.8333	0.8333
20	Balance Scale	5	625	3.33×10^{-1}	0.9152	0.3333	0.8672	0.8656	0.7072	0.9152	0.9152	0.9152	0.9152	0.9152
21	Lenses	5	24	3.33×10^{-1}	0.7500	0.8333	0.6667	0.7083	0.7500	0.8333	0.7500	0.8750	0.8750	0.8750
22	Iris	5	150	6.17×10^{-1}	0.9400	0.9467	0.9200	0.9400	0.9267	0.9467	0.9400	0.9400	0.9467	0.9467
23	LED7	8	3200	2.50×10^{0}	0.7294	0.7372	0.7384	0.7350	0.7325	0.7294	0.7294	0.7294	0.7316	0.7325
24	Banknote authentication	5	1372	2.72×10^{0}	0.9249	0.9413	0.9388	0.9293	0.9388	0.9402	0.9410	0.9410	0.9410	0.9410
	average				0.8106	0.7529	0.8160	0.8189	0.7938	0.8111	0.8201	0.8354	0.8385	0.8354
	p-value (*Proposed(BFS) vs. the other methods*)				0.0128	0.0024	0.0071	0.0466	0.0051	0.0012	0.0801	0.10 <	-	-
	p-value (*Proposed(DFB&B) vs. the other methods*)				0.0847	0.0075	0.10 <	0.10 <	0.0278	0.0166	0.10 <	0.10 <	-	-

Table 1: Classification accuracies achieved using the respective methods.

No.	Naive-Bayes	GBN-CMDL	BNC2P	TAN-aCLL	MC-DAG GES	GBN-BDeu	ANB-BDeu	fsANB-BDeu	Proposed (BFS)	Proposed (DFB&B)
1	1091.0	34356.3	4374.7	2358.0	1091.0	2320.8	25640.8	25619.0	4323.6	5550.7
2	1499.0	68425.8	10467.6	13435.0	930231.0	15252.9	15986.0	15986.0	9175	9887
3	3665.0	109779123.0	32919.6	21761.0	113400.6	19121.1	31235.4	30179.8	12354.2	12354.2
4	167.0	292.5	431.0	384.6	587.0	206510.5	727675.8	147.4	104.2	145.8
5	137.0	656785.6	494.2	1060.6	34781.4	2834.0	7515.0	7515.0	1849.2	1849.4
6	163.0	1000.0	1126.9	1459.0	7469.2	153.1	163.0	163.0	161.2	161.2
7	146.0	70.4	126.6	295.8	146.0	415.7	1032.9	524.7	508	290
8	39.0	8.1	53.6	75.0	39.0	1977.3	6824.8	470.8	9.6	13.4
9	74.0	23.0	159.7	152.3	74.0	87.2	8183.6	8183.6	28.4	28.4
10	71.0	415.0	363.2	184.8	67.0	19.2	280.6	211.8	64	60
11	75.0	75.4	69.6	143.0	109.0	79.7	322.2	312.2	1377	1018.2
12	65.0	1122.0	352.9	144.8	14979.0	3.6	103.6	28.4	35.2	30.2
13	41.0	51.2	91.8	82.2	111.8	22.6	52.2	49.8	19.2	19.2
14	99.0	28672.0	221.3	645.0	3524.0	246.1	2119.0	2119.0	197.8	176.2
15	33.0	4.0	17.4	63.0	31.2	17.6	96.2	85.0	9.8	9
16	206.0	1381.4	624.6	839.6	8342.0	19.4	904.4	308.6	8	8
17	59.0	69.2	83.5	107.0	80.6	30.2	106.4	78.8	480.2	509.6
18	53.0	780.8	123.5	121.4	234.2	11.0	138.8	137.0	32.6	27.2
19	35.0	128.0	93.5	107.0	198.8	128.0	35.0	29.0	29	29
20	50.0	1250.0	179.2	194.0	458.0	48.4	50.0	50.0	50	50
21	17.0	8.3	26.7	29.3	17.0	8.2	17.3	8.3	8.3	8.3
22	26.0	12.6	55.6	62.0	102.8	18.0	36.8	36.8	18.8	18.8
23	79.0	124.0	136.5	139.0	119.0	78.1	79.0	79.0	94	98
24	29.0	126.0	101.9	71.0	355.0	247.6	315.4	315.4	15.4	15.4
average	330.0	4607262.7	2195.6	1829.8	46522.9	10402.1	34538.1	3859.9	1289.7	1348.2

Table 2: Number of class variable parameters (NCP) of the learned structures obtained using the respective methods.

	Runtime										NTCE
No.	Naive-Bayes	GBN-CMDL	BNC2P	TAN-aCLL	MC-DAG GES	GBN-BDeu	ANB-BDeu	fsANB-BDeu	Proposed (BFS)	Proposed (DFB&B)	Proposed (DFB&B)
1	0.0	2573.8	78.1	540.9	5.9	860.5	434.1	2713.0	5411.7	296.7	149209.5
2	0.0	8351.6	496.7	852.0	197.9	1227.2	605.7	6461.7	700.2	147.4	37821.8
3	0.0	60640.0	2420.7	2238.4	511.2	2231.4	1107.2	3671.8	504.2	103.2	78524.0
4	0.0	78.1	3.6	33.9	0.5	67.4	34.1	62.7	555.7	97.3	189638.4
5	0.0	1604.4	84.4	18.1	33.1	37.7	20.5	295.1	25.7	12.0	13069.7
6	0.0	5.0	1.1	2.2	0.5	0.3	0.2	2.0	0.3	0.1	98.0
7	0.0	40.7	1.4	8.0	0.3	5.7	2.8	6.5	377.2	18.2	6915.0
8	0.0	32.4	1.5	30.6	0.2	65.7	33.0	290.0	10044.8	250.8	386621.3
9	0.0	11.0	1.4	3.1	0.3	0.9	0.5	4.2	14.8	5.6	5945.4
10	0.0	54.9	3.6	4.9	0.7	7.2	3.7	10.3	21.3	7.7	8652.0
11	0.0	351.2	12.0	37.9	5.1	102.3	55.3	177.2	6527.6	206.5	159026.0
12	0.0	2.5	0.5	2.2	0.2	0.2	0.1	0.2	0.3	0.2	176.1
13	0.0	12.3	1.3	2.6	0.4	1.2	0.7	4.5	10.9	4.5	5104.1
14	0.0	98.2	12.7	2.2	6.1	0.3	0.2	3.2	0.2	0.3	74.5
15	0.0	29.1	1.4	5.6	0.3	8.3	4.1	23.9	427.1	34.8	22001.9
16	0.0	534.1	6.5	6.8	1.6	0.3	0.2	0.4	0.9	0.5	326.2
17	0.0	3.8	0.7	4.7	0.2	0.1	0.0	0.4	0.5	0.3	178.9
18	0.0	15.9	2.2	2.8	0.8	0.2	0.1	0.9	0.4	0.2	134.1
19	0.0	0.1	0.1	2.8	0.0	0.0	0.0	0.0	0.0	0.0	3.3
20	0.0	0.3	0.2	2.8	0.1	0.0	0.0	0.1	0.0	0.0	4.0
21	0.0	0.1	0.1	2.7	0.0	0.0	0.0	0.1	0.0	0.0	3.9
22	0.0	0.1	0.1	2.7	0.0	0.0	0.0	0.1	0.0	0.0	3.4
23	0.0	40.7	8.6	7.3	1.1	0.1	0.0	1.0	0.1	0.1	27.6
24	0.0	0.4	0.3	2.1	0.1	0.0	0.0	0.1	0.0	0.0	1.9
average	0.0	3103.4	130.8	159.1	32.0	192.4	96.0	572.0	1026.0	49.4	44315.0

Table 3: Average runtimes (s) of the respective methods and the numbers of times *Proposed(DFB&B)* cuts edges.

cate that neither the difference between *Proposed(BFS)* and *ANB-BDeu* nor the difference between *Proposed(BFS)* and *fsANB-BDeu* is significant. The reason might be that datasets in Table 1 satisfy Assumptions 1 and 2. As shown in the next subsection, we demonstrate that the proposed method outperforms *GBN-BDeu*, *ANB-BDeu*, and *fsANB-BDeu* when Assumptions 1 and 2 are violated. The classification accuracies of *Proposed(BFS)* are approximately equal to those of *Proposed(DFB&B)* for each dataset. However, *Proposed(BFS)* provides higher average classification accuracy than that of *Proposed(DFB&B)*. Because *Proposed(BFS)* uses no branch and bound algorithm to cut edges of NROGs, *Proposed(BFS)* estimates the proposed BNC structures more precisely than *Proposed(DFB&B)* does. From Table 3, the average runtime of the *Proposed(DFB&B)* is shorter than those of the other methods, except for *MCDAGGES*. Especially, *Proposed(DFB&B)* provides much shorter runtime than *Proposed(BFS)* does for large NTCE of *Proposed(DFB&B)* such as datasets Nos. 1 and 8. The results demonstrate the efficiency of the proposed branch and bound algorithm of *Proposed(DFB&B)*.

Finally, the classification accuracies of the ten methods for eleven large datasets from the UCI repository are compared. Table 4 presents the classification accuracies of the respective classifiers for the large datasets. In Table 4, "time over (TO)" signifies that learning was not completed within a time limit of 6 hr and "out of memory (MO)" signifies a failure to learn the structure because of memory insufficiency. The results presented in Table 4 show that *Proposed(DFB&B)* outperforms the other methods at the $p < 0.1$ significance level. *TAN-aCLL* fails to learn structures because of MO. Four exact learning methods (*GBN-BDeu*, *ANB-BDeu*, *fsANB-BDeu*, and *Proposed(BFS)*) fail to learn structures because of TO. The computational time and space of these exact learning methods increase exponentially with the number of variables. However, *Proposed(DFB&B)* can be stopped at any time. The current best solution is obtainable because the depth-first search sequentially updates the current best solution.

Simulation datasets

To demonstrate the advantage of the proposed method compared to *GBN-BDeu*, *ANB-BDeu*, and *fsANB-BDeu*, this subsection presents additional experiments using simulation data. The experiments compare NCPs of structures learned using the four methods and compare the Kullback–Leibler divergences (KLDs) between the true posteriors of the class variable and those using the four methods from simulation

No.	Dataset	Variables	Sample size	Naive-Bayes	GBN-CMDL	BNC2P	TAN-aCLL	MC-DAG GES	GBN-BDeu	ANB-BDeu	fsANB-BDeu	Proposed (BFS)	Proposed (DFB&B)
1	wdbc	31	569	0.9139	0.9209	0.9209	MO	0.9156	TO	TO	TO	TO	0.9350
2	ionosphere	35	351	0.7550	0.8860	0.7493	MO	0.7493	TO	TO	TO	TO	0.8832
3	kr-vs-kp	37	3196	0.6640	0.9252	0.8339	MO	0.7672	TO	TO	TO	TO	0.9252
4	biodeg	42	1055	0.7877	0.8237	0.8284	MO	0.8066	TO	TO	TO	TO	0.7877
5	Flowmeters_D	44	180	0.8333	0.8000	0.8833	MO	0.8500	TO	TO	TO	TO	0.8833
6	Parkinson	48	240	0.7625	0.5000	0.4750	MO	0.7708	TO	TO	TO	TO	0.7708
7	PAMAP2	53	174915	0.6864	0.6306	0.6502	MO	0.7052	TO	TO	TO	TO	0.8634
8	spam	58	4601	0.8794	0.9270	0.9168	MO	0.8813	TO	TO	TO	TO	0.9331
9	molecular	61	3190	0.9433	0.9241	0.9364	MO	0.9266	TO	TO	TO	TO	0.9464
10	Nuclear	75	1047	0.9303	1.0000	1.0000	MO	0.9370	TO	TO	TO	TO	0.9914
11	MI	116	544	0.9154	0.9081	0.9136	MO	0.9154	TO	TO	TO	TO	0.9375
	average	-	-	0.8247	0.8405	0.8280	-	0.8387	-	-	-	-	0.8961
p-value(*Proposed(DFB&B)* vs. the other methods)				0.0051	0.0930	0.0366	-	0.0069	-	-	-	-	-

Table 4: Classification accuracies of the respective methods used for large datasets.

		ANB-BDeu				GBN-BDeu				fsANB-BDeu				Proposed(BFS)			
Network	Sample size	avgKLD	NCP	Imin(NP)	MB	avgKLD	NCP	Imin(GBN)	MB	avgKLD	NCP	Imin(NP)	MB	avgKLD	NCP	Imin(NP)	MB
Cancer (Structure (1))	100	5.11×10^{-2}	9	×	4	2.70×10^{-2}	5	×	2	5.87×10^{-2}	7	×	3	2.70×10^{-2}	5	×	2
	1,000	4.11×10^{-2}	9	×	4	2.67×10^{-2}	5	×	2	3.97×10^{-2}	7	×	3	2.67×10^{-2}	5	×	2
	10,000	1.27×10^{-3}	11	○	4	1.27×10^{-3}	8	○	4	1.27×10^{-3}	11	○	4	1.27×10^{-3}	11	○	4
	100,000	8.46×10^{-5}	11	○	4	8.46×10^{-5}	8	○	4	8.46×10^{-5}	11	○	4	8.46×10^{-5}	11	○	4
Asia (Structure (2))	100	8.81×10^{-2}	21	×	7	5.21×10^{-2}	5	×	2	1.04×10^{-1}	13	×	5	4.40×10^{-2}	3	×	1
	1,000	3.70×10^{-2}	21	×	7	3.40×10^{-2}	9	×	3	4.89×10^{-2}	7	×	3	6.38×10^{-2}	7	×	2
	10,000	2.58×10^{-2}	21	×	7	3.60×10^{-3}	10	×	5	2.88×10^{-2}	15	×	5	2.46×10^{-2}	11	×	4
	100,000	1.94×10^{-3}	25	×	7	2.72×10^{-4}	10	○	5	1.32×10^{-2}	17	×	5	2.72×10^{-4}	13	○	5
Markov net (Structure (3))	100	2.20×10^{-1}	29	×	3	2.08×10^{-1}	3	×	1	2.20×10^{-1}	29	×	3	8.18×10^{-2}	5	×	1
	1,000	6.47×10^{-2}	29	×	3	1.38×10^{-2}	17	×	2	6.47×10^{-2}	29	×	3	6.63×10^{-2}	5	×	1
	10,000	2.96×10^{-3}	29	×	3	2.96×10^{-3}	27	×	3	2.96×10^{-3}	29	×	3	4.43×10^{-4}	17	○	2
	100,000	1.20×10^{-3}	29	×	3	1.20×10^{-3}	29	×	3	1.20×10^{-3}	29	×	3	7.94×10^{-5}	17	○	2

Table 5: NCPs of structures learned by *ANB-BDeu*, *GBN-BDeu*, *fsANB-BDeu*, and *Proposed(BFS)*, and the average KLDs between the true posteriors of the class variable and those of the four methods.

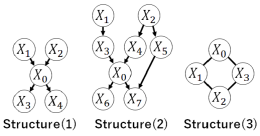

Figure 3: Structures: (1) the Cancer network, (2) the Asia network, and (3) a Markov network with a cycle.

datasets. This experiment uses the following three networks: the Cancer network (Scutari 2010) in the structure (1) of Figure 3, which satisfies Assumptions 1 and 2; the Asia network (Scutari 2010) in the structure (2) of Figure 3, which satisfies Assumption 1 but which violates Assumption 2; and the Markov network in the structure (3) of Figure 3, which violates Assumptions 1 and 2.

From the three networks, we generate datasets randomly with sizes $N = 100, 1,000, 10,000$, and $100,000$. Based on the generated data, after learning BNC structures using the four methods, we evaluate the KLDs and the NCPs. Table 5 presents the NCPs and the average KLD over all feature variables' values. "Imin(GBN)" and "Imin(NP)" in Table 5 respectively signify whether the learned structure is an I-map with the lowest NCP in all GBN structures and in all NPCDAGs. In Table 5, "MB" signifies the size of the class variable's Markov blanket.

Results demonstrate that, for the Cancer network, the average KLDs between the true posteriors of the class variable and the ones learned by all four methods are identical when $N \geq 10,000$ because the four methods learn an I-map with the lowest NCP. Similarly, for the Asia network, the average KLDs between the true posteriors of the class variable and those learned by *Proposed(BFS)* are identical to those by exact learning *GBN-BDeu* when $N \geq 10,000$ because *Proposed(BFS)* and the *GBN-BDeu* learn an I-map with the lowest NCP. However, *ANB-BDeu* and *fsANB-BDeu* provide larger NCPs and larger average KLD than *GBN-BDeu* and *Proposed(BFS)* do when $N \geq 10,000$. The reason is that *ANB-BDeu* and *fsANB-BDeu* do not guarantee an asymptotic classification equivalence to the true structure when Assumption 2 does not hold.

Moreover, in the Markov network presented in Figure 3, the average KLDs between the true posteriors of the class variable and those learned by *Proposed(BFS)* are smaller than those learned by *GBN-BDeu* when $N \geq 10,000$ because *Proposed(BFS)* learns I-maps with the lowest NCP, but the *GBN-BDeu* does not. The reason is that the *GBN-BDeu* does not asymptotically guarantee estimation of an I-map minimizing NCP, although it guarantees minimization of the number of all parameters. Furthermore, *GBN-BDeu* has a larger size of the class variable Markov blanket than *Proposed(BFS)* does when $N \geq 10,000$.

Conclusions

We proposed a new BNC having an I-map NPCDAG with the lowest NCP. The proposed BNC asymptotically does not include irrelevant feature variables for the class variable because NPCDAGs do not force the addition of edges between the class variable and feature variables. The proposed learning BNC achieves asymptotic classification equivalence to the true structure, even when the true structure does not follow ANB. Moreover, the proposed BNC is guaranteed to minimize the NCP irrespective of whether the true model follows a BN, or not. Furthermore, this study proposed and evaluated an algorithm to learn the proposed BNC using depth-first branch and bound algorithm. Specifically, we (1) prove that the NCP of Naive Bayes is lower bound NCP of the proposed BNC and (2) propose a new heuristic function using the lower bound. The experiment results demonstrated the reflectivity of the proposed method. As future work, we expect to apply model averaging (Liao et al. 2019) to our proposed method.

References

Acid, S.; De Campos, L. M.; and Castellano, J. G. 2005. Learning Bayesian Network Classifiers: Searching in a Space of Partially Directed Acyclic Graphs. *Machine Learning*, 59: 213–235.

Barlett, M.; and Cussens, J. 2013. Advances in Bayesian Network Learning Using Integer Programming. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 182–191.

Carvalho, A. M.; Adão, P.; and Mateus, P. 2013. Efficient Approximation of the Conditional Relative Entropy with Applications to Discriminative Learning of Bayesian Network Classifiers. *Entropy*, 15: 2716–2735.

Chickering, D. M. 2002. Optimal Structure Identification With Greedy Search. *JMLR*, 3: 507–554.

Cussens, J. 2012. Bayesian network learning with cutting planes. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 153–160.

De Campos, C. P.; and Ji, Q. 2011. Efficient Structure Learning of Bayesian Networks Using Constraints. *JMLR*, 12: 663–689.

Demšar, J. 2006. Statistical Comparisons of Classifiers over Multiple Data Sets. *JMLR*, 7: 1–30.

Fayyad, U. M.; and Irani, K. B. 1993. Multi-Interval Discretization of Continuous-Valued Attributes for Classification Learning. In *International Joint Conference on Artificial Intelligence*, 1022–1027.

Friedman, N.; Geiger, D.; and Goldszmidt, M. 1997. Bayesian Network Classifiers. *Machine Learning*, 29: 131–163.

Greiner, R.; and Zhou, W. 2002. Structural Extension to Logistic Regression: Discriminative Parameter Learning of Belief Net Classifiers. In *Proceedings of the National Conference on Artificial Intelligence*, 167–173.

Grossman, D.; and Domingos, P. 2004. Learning Bayesian Network classifiers by maximizing conditional likelihood. In *Proceedings of the International Conference on Machine Learning*, 361–368.

Heckerman, D.; Geiger, D.; and Chickering, D. M. 1995. Learning Bayesian Networks: The Combination of Knowledge and Statistical Data. *Machine Learning*, 20: 197–243.

Hommel, G. 1988. A stagewise rejective multiple test procedure based on a modified bonferroni test. *Biometrika*, 383–386.

Koivisto, M.; and Sood, K. 2004. Exact Bayesian Structure Discovery in Bayesian Networks. *JMLR*, 5: 549–573.

Liao, Z. A.; Sharma, C.; Cussens, J.; and van Beek, P. 2019. Finding All Bayesian Network Structures within a Factor of Optimal. In *Proceedings of the AAAI Conference on Artificial Intelligence*.

Lichman, M. 2013. UCI Machine Learning Repository.

Ling, C. X.; and Zhang, H. 2003. The Representational Power of Discrete Bayesian Networks. *JMLR*, 3: 709–721.

Lucas, P. J. F. 2004. Restricted Bayesian Network Structure Learning. 146: 217–234.

Malone, B.; Yuan, C.; Hansen, E. A.; and Bridges, S. 2011. Improving the Scalability of Optimal Bayesian Network Learning with External-Memory Frontier Breadth-First Branch and Bound Search. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 479–488.

Mihaljević, B.; Bielza, C.; and Larrañaga, P. 2018. Learning Bayesian network classifiers with completed partially directed acyclic graphs. In *Proceedings of the International Conference on Probabilistic Graphical Models*, 272–283.

Minsky, M. 1961. Steps toward Artificial Intelligence. In *Proceedings of the IRE*, 8–30.

Pedregosa, F.; Varoquaux, G.; Gramfort, A.; Michel, V.; Thirion, B.; Grisel, O.; Blondel, M.; Prettenhofer, P.; Weiss, R.; Dubourg, V.; Vanderplas, J.; Passos, A.; Cournapeau, D.; Brucher, M.; Perrot, M.; and Édouard Duchesnay. 2011. Scikit-learn: Machine Learning in Python. *JMLR*, 12(85): 2825–2830.

Scutari, M. 2010. Learning Bayesian Networks with the bnlearn R Package. *JSSA*, 35: 1–22.

Silander, T.; and Myllymäki, P. 2006. A Simple Approach for Finding the Globally Optimal Bayesian Network Structure. In *Proceedings of the Conference on Uncertainty in Artificial Intelligence*, 445–452.

Singth, A.; and Moore, A. 2005. Finding optimal Bayesian networks by dynamic programming. Technical report, Technical Report, Carnegie Mellon University.

Sugahara, S.; Uto, M.; and Ueno, M. 2018. Exact learning augmented naive Bayes classifier. In *Proceedings of the International Conference on Probabilistic Graphical Models*, 439–450.

Suzuki, J. 2017. A theoretical analysis of the BDeu scores in Bayesian network structure learning. *Behaviormetrika*, 44: 97–116.

Yuan, C.; and Malone, B. 2013. Learning Optimal Bayesian Networks: A Shortest Path Perspective. *JAIR*, 48: 23–65.