Journal of Artificial Intelligence Research 59 (2017) 103-132 Submitted 10/16; published 06/17

Learning Discrete Bayesian Networks from Continuous Data

Yi-Chun Chen yichunc0419@alumni.stanford.edu
Institute of Computational and Mathematical Engineering, Stanford University, Stanford, CA 94035 USA

Tim A. Wheeler wheelert@stanford.edu
Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94035 USA

Mykel J. Kochenderfer mykel@stanford.edu
Department of Aeronautics and Astronautics, Stanford University, Stanford, CA 94035 USA

Abstract

Learning Bayesian networks from raw data can help provide insights into the relationships between variables. While real data often contains a mixture of discrete and continuous-valued variables, many Bayesian network structure learning algorithms assume all random variables are discrete. Thus, continuous variables are often discretized when learning a Bayesian network. However, the choice of discretization policy has significant impact on the accuracy, speed, and interpretability of the resulting models. This paper introduces a principled Bayesian discretization method for continuous variables in Bayesian networks with quadratic complexity instead of the cubic complexity of other standard techniques. Empirical demonstrations show that the proposed method is superior to the established minimum description length algorithm. In addition, this paper shows how to incorporate existing methods into the structure learning process to discretize all continuous variables and simultaneously learn Bayesian network structures.

1. Introduction

Bayesian networks (Pearl, 1988; Koller & Friedman, 2009) are often used to model uncertainty and causality, with applications ranging from decision-making systems (Kochenderfer, Holland, & Chryssanthacopoulos, 2012) to medical diagnosis (Lustgarten, Visweswaran, Gopalakrishnan, & Cooper, 2011). Learning Bayesian networks from raw data can help users discover knowledge without field-specific experts, such as causality and probabilistic relation between variables. It is common to assume that the random variables in a Bayesian network are discrete, since many Bayesian network learning algorithms are unable to efficiently handle continuous variables. In addition, many of the commonly used Bayesian network software packages, such as Netica (Norsys, 2009), SMILearn (Druzdzel, 1999), and bnlearn (Scutari, 2010), are designed for discrete variables. However, many applications require the use of continuous data, such as positions and velocities in dynamic systems (Kochenderfer, Edwards, Espindle, Kuchar, & Griffith, 2010).

There are three common approaches to extending Bayesian network learning algorithms to continuous variables. The first is to model the conditional probability density of each continuous variable using specific families of parametric distributions, and then to redesign Bayesian network learning algorithms based on the parameterizations. One example of parametric continuous distributions in Bayesian networks is the Gaussian graphical model (Weiss & Freeman, 2001).

The second approach is to use nonparametric distributions, such as particle representations and Gaussian processes (Ickstadt, Bornkamp, Grzegorczyk, Wieczorek, Sheriff, Grecco, & Zamir, 2010). Unlike parametric methods, nonparametric methods can often fit any underlying probability distribution given sufficient data. Parametric models have a fixed number of parameters, whereas the number of parameters in a nonparametric model grow with the amount of training data.

The third approach is discretization. Automated discretization methods have been studied in machine learning and statistics for many years (Dougherty, Kohavi, & Sahami, 1995; Kerber, 1992; Holte, 1993; Fayyad & Irani, 1993), primarily for classification problems. These methods learn the best discretization policy for a continuous attribute by considering its interaction with a class variable. It is common to discretize all continuous variables before learning a Bayesian network structure to avoid having to consider variable interactions, as the interactions and dependencies between variables in Bayesian networks introduce complexity. Prior work exists for discretizing continuous variables in naive Bayesian networks and tree-augmented networks (Friedman, Geiger, & Goldszmidt, 1997), but only a few discretization methods for general Bayesian networks have been proposed (Friedman & Goldszmidt, 1996; Kozlov & Koller, 1997; Monti & Cooper, 1998; Steck & Jaakkola, 2004).

A common discretization method for Bayesian networks is to split continuous data into uniform-width intervals or to use field-specific expertise. Several Bayesian methods exist to discretize according to the marginal distribution (Scott, 1979; Freedman & Diaconis, 1981; Knuth, 2013; Scargle, Norris, Jackson, & Chiang, 2013), but this ignores dependencies between variables and leads to suboptimal performance, as demonstrated in Appendix A. A more principled method is the minimum description length (MDL) principle discretization (Friedman & Goldszmidt, 1996). The MDL principle proposed by Rissanen (1978) states that the best model for a dataset is the one that minimizes the amount of information needed to describe it (Grünwald, 2007). MDL methods trade off goodness-of-fit against model complexity to reduce generalization error. In the context of Bayesian networks, Friedman and Goldszmidt (1996) applied the MDL principle to determine the optimal number of discretization intervals for continuous variables and the optimal positions of their discretization edges. Their approach selects a discretization policy that minimizes the sum of the description length of the discretized Bayesian network and the information necessary for recovering the continuous values from the discretized data.

The optimal discretization policy of a single continuous variable under MDL can be found using dynamic programming in cubic runtime with the number of data instances. For Bayesian networks with multiple continuous variables, MDL discretization can be iteratively applied to each continuous variable. Only one variable is treated as continuous at a time, while all other continuous variables are treated as discretized based on an initial discretization policy or the discretization result from a previous iteration.

MDL discretization requires that the network structure be known in advance, but an iterative approach allows it to be incorporated into the structure learning process. Simultaneous structure learning and discretization alternates between traditional discrete structure learning and optimal discretization. Starting with some preliminary discretization policy, one first applies a structure learning algorithm to identify the locally optimal graph structure. One then refines the discretization policy based on the learned network. The cycle is repeated until convergence.

Results in this work suggest that the MDL method suffers from low sensitivity to discretization edge locations and returns too few discretization intervals for continuous variables. This is caused by MDL's use of mutual information to measure the quality of discretization edges. Mutual information, which is composed of empirical probabilities computed using event count ratios, varies less significantly with the positions of discretization edges than the method we suggest in this article.

MODL (Boullé, 2006) is a Bayesian method for discretizing a continuous feature according to a class variable, which selects the model with maximum probability given the data. The MODL method uses dynamic programming to find the optimal discretization policy for a continuous variable given a discrete class variable, and has an $O\big(n^3 + r \cdot n^2\big)$ runtime, where r is the number of class variable instantiations. Lustgarten et al. (2011) suggested several formulations for the prior over models. The asymptotic equivalence between MDL and MODL on the single-variable, single-class problem was examined by Vitányi and Li (2000).

This paper describes a new Bayesian discretization method for continuous variables in Bayesian networks, extending prior work on single-variable discretization methods from Boullé (2006) and Lustgarten et al. (2011). The proposed method optimizes the discretization policy relative to the network and takes parents, children, and spouse variables into account. The optimal single-variable discretization method is derived in Section 3, using a prior which reduces the discretization runtime to $O\big(r \cdot n^2\big)$ without sacrificing optimality. Section 4 covers Bayesian networks with multiple continuous variables and Section 5 covers discretization while simultaneously learning network structure. The paper concludes with a comparison against the existing minimum-description length (Friedman & Goldszmidt, 1996) method on real-world and synthetic datasets in Section 6.

2. Preliminaries

This section covers the notation used throughout the paper to describe discretization policies. This section also provides a brief overview of Bayesian networks.

2.1 Discretization Policies

Let $\mathscr{X}$ be a continuous variable and let x be a specific instance of $\mathscr{X}$. A discretization policy $\Lambda_{\mathscr{X}} = \langle e_1 < e_2 < \ldots < e_{k-1} \rangle$ for $\mathscr{X}$ is a mapping from $\mathbb{R}$ to $\{1, 2, \ldots, k\}$ such that

$$\Lambda_{\mathscr{X}}(x) = \begin{cases} 1, & \text{if } x < e_1 \\ i, & \text{if } e_{i-1} \leq x < e_i \\ k, & \text{otherwise.} \end{cases} \tag{1}$$

The discretization policy discretizes $\mathscr{X}$ into k intervals. Let the samples of $\mathscr{X}$ in a given dataset D be sorted in ascending order, $x_{1:n} = \{x_1 \leq x_2 \leq \ldots \leq x_n\}$, and let the unique values be $u_{1:m} = \{u_1 < u_2 < \ldots < u_m\}$. The index of the last occurrence of u_i in $x_{1:n}$ is denoted s_i.

The discretization edges $e_{1:k-1}$ mark the boundaries between discretization intervals. In this paper, as with MODL, they are restricted to the midpoints between unique ascending instances of $\mathscr{X}$. Thus, each edge e_i equals $(u_j + u_{j+1})/2$ for some j. Two useful integer representations of $\Lambda_{\mathscr{X}}$ can be written

$$\Lambda_{\mathscr{X}} = \langle \lambda_1 < \lambda_2 < \ldots < \lambda_k \rangle \equiv \langle \gamma_1, \gamma_2, \ldots, \gamma_k \rangle, \tag{2}$$

where $\lambda_1 = s_1$, $\lambda_i \in s_{1:m}$, $\gamma_1 = \lambda_1$, and $\gamma_i = \lambda_i - \lambda_{i-1}$. The $\lambda_{1:k}$ representation is the number of instances before every discretization edge whereas the $\gamma_{1:k}$ representation is the number of instances within each discretization interval. Both $\lambda_{1:k}$ and $\gamma_{1:k}$ representations will be used in the following content. The former makes the dynamic programming algorithm more succinct, and the latter makes equations more compact when calculating data instantiations. Table 1 and 2 demonstrate variable assignments for an example dataset.

Table 1: An example dataset used to demonstrate the variables defined in Section 2.1. Rows are assignments to continuous variable $\mathscr{X}$ and discrete variables A and B.

$\mathscr{X}$	A	B
1.1	1	3
1.2	1	3
1.2	1	4
2.9	2	3
2.9	2	4
2.9	1	4
3.5	1	4
4.3	1	4
4.3	2	3
5.0	2	3

Table 2: Assignments for variables defined in Section 2.1 according to the dataset in Table 1. Note that $\Lambda_{\mathscr{X}}^{\text{ex}}$ is an example of a valid discretization policy. Discretization policies can be represented by their edges $\langle\rangle_e$, their sorted unique dataset indices $\langle\rangle_\lambda$, or by the interval between sorted unique dataset indices $\langle\rangle_\gamma$. That is to say, in the edge representation, $e_1 = 1.55$ and $e_2 = 4.65$. In the $\lambda_{1:k}$ representation, $\lambda_1 = 3$, $\lambda_2 = 9$, and $\lambda_3 = 10$. In the $\gamma_{1:k}$ representation, $\gamma_1 = 3$, $\gamma_2 = 6$, and $\gamma_3 = 10$. Furthermore, one can check that $\lambda_1 = \gamma_1 = 3$, $\lambda_2 - \lambda_1 = \gamma_2$, and $\lambda_3 - \lambda_2 = \gamma_2$.

Variable	Value
n	10
m	6
$x_{1:10}$	$\{1.1, 1.2, 1.2, 2.9, 2.9, 2.9, 3.5, 4.3, 4.3, 5.0\}$
$u_{1:6}$	$\{1.1, 1.2, 2.9, 3.5, 4.3, 5.0\}$
$s_{1:6}$	$\{1, 3, 6, 7, 9, 10\}$
$\Lambda_{\mathscr{X}}^{\text{ex}}$	$\langle 1.55, 4.65\rangle_e$, $\langle 3, 9, 10\rangle_\lambda$, $\langle 3, 6, 1\rangle_\gamma$

2.2 Bayesian Networks

A Bayesian Network B over N random variables $X_{1:N}$ is defined by a directed acyclic graph G containing one node for each variable X_i, and each node is associated with a conditional probability distribution $P(X_i \mid \text{pa}_{X_i})$ for the node given its parents in G. The edges in a Bayesian network represent probabilistic dependencies among nodes and encode the Markov property: each node X_i is independent of its non-descendants given its parents pa_{X_i} in G. The children of node X_i are denoted ch_{X_i}.

When discussing the discretization of a particular continuous variable $\mathscr{X}$, let P_i be the ith parent of $\mathscr{X}$, let C_i be the ith child of $\mathscr{X}$, and let $\boldsymbol{S}_i$ be the set of spouses of $\mathscr{X}$ associated with the ith child. The parents, children, and spouses of a variable form its Markov blanket.

3. Single Variable Discretization

This section covers the discretization of a single continuous variable $\mathscr{X}$ in a Bayesian network where all other variables are discrete. An optimal discretization policy for a dataset D maximizes $P(\Lambda) \cdot P(D \mid \Lambda)$ for some prior $P(\Lambda)$ and likelihood $P(D \mid \Lambda)$.

3.1 Priors and Objective Function

Let $D_{\boldsymbol{Y}}$ be the subset of the training data corresponding to variable subset $\boldsymbol{Y}$. Four principles for the optimal discretization policy enable the formulation of $P(\Lambda)$ and $P(D_{-\mathscr{X}} \mid \Lambda)$, where $D_{-\mathscr{X}}$ is the subset of the dataset for all variables but $\mathscr{X}$, and the probability of the data associated with the target variable, $P(D_{\mathscr{X}})$, is already captured in $P(\Lambda)$. The four principles, which can be considered an extension of the priors in MODL (Boullé, 2006) and Lustgarten et al. (2011) to Bayesian networks, are:

1. The prior probability of a discretization edge between two consecutive unique values u_i and u_{i+1} is proportional to their difference:

$$1 - \exp\left(-L \cdot \frac{u_{i+1} - u_i}{u_m - u_1}\right), \tag{3}$$

   where larger values of L encourage more discretization intervals. This prior encourages edges between well-separated values over edges between closely packed samples. The experiments in this paper set L to the largest number of intervals among discrete variables in $\mathscr{X}$'s Markov blanket, which encourages Bayesian networks with consistent interval counts. Higher values of L can be used when the continuous variable requires more discretization intervals than its Markov blanket.

2. For a given discretization interval, every distribution over the parents of $\mathscr{X}$ is equiprobable.

3. For each pair $\langle C_i, \boldsymbol{S}_i \rangle$ and a given discretization interval, every distribution over C_i given an instance of $\boldsymbol{S}_i$ is equiprobable.

4. The distributions over $\mathscr{X}$ given each child, parent, and spouse instantiation are independent.

From the first principle one obtains the prior over the discretization policy:

$$P(\Lambda) \propto \prod_{i=1}^{k-1} \left[1 - \exp\left(-L \cdot \frac{x_{\lambda_i+1} - x_{\lambda_i}}{x_n - x_1}\right)\right] \prod_{i=1}^{k} \left[\exp\left(-L \cdot \frac{x_{\lambda_i} - x_{\lambda_{i-1}+1}}{x_n - x_1}\right)\right]. \tag{4}$$

The likelihood term $P(D_{-\mathscr{X}} \mid \Lambda)$ for a Bayesian network graph structure factors according to $\mathscr{X}$'s Markov blanket:

$$P(D_{-\mathscr{X}} \mid \Lambda) \propto P\big(D_{\mathrm{pa}_{\mathscr{X}}} \mid \Lambda\big) \cdot \prod_i P(D_{C_i} \mid \Lambda, D_{\boldsymbol{S}_i}). \tag{5}$$

For example, in Figure 1, $P(D_{-\mathscr{X}} \mid \Lambda)$ factors according to

$$P(D_{P_1,P_2,P_3} \mid \Lambda) \cdot P(D_{C_1} \mid \Lambda, D_{\boldsymbol{S}_1}) \cdot P(D_{C_2} \mid \Lambda, D_{\boldsymbol{S}_2}). \tag{6}$$

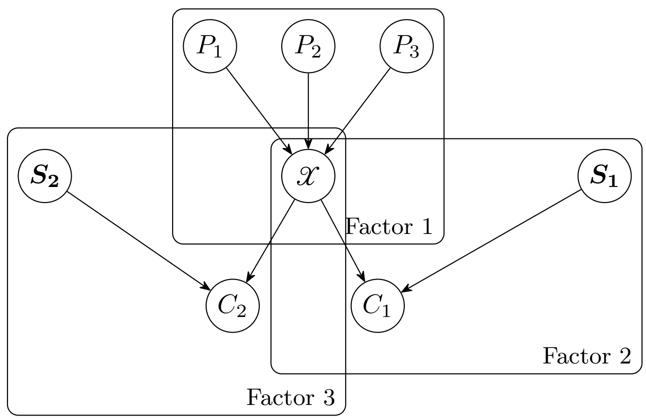

Figure 1: Factorization of $P(D_{-\mathscr{X}} \mid \Lambda)$.

The concept behind the factorization is also the motivation behind forward sampling in a Bayesian network. The parents of $\mathscr{X}$ are independent of the children given $\mathscr{X}$. The parents are not necessarily individually independent, and thus the parental term $P(D_{\mathrm{pa}_{\mathscr{X}}} \mid \Lambda)$ cannot be factored further. The children of $\mathscr{X}$ are similarly independent given $\mathscr{X}$ and the corresponding spouses, leading to their factored product $\prod_i P(D_{C_i} \mid \Lambda, D_{\boldsymbol{S}_i})$. Each component in the decomposition can be evaluated given a discretization policy and a dataset.

3.1.1 Evaluation of $P\big(D_{\mathrm{pa}_{\mathscr{X}}} \mid \Lambda\big)$

Let J_{P} be the number of instantiations of the parents of $\mathscr{X}$, and let $n_{i,j}^{(\mathrm{P})}$ be the number of instances of $\mathscr{X}$ within the ith discretization interval of Λ given the jth parental instantiation. Note that $\gamma_i = \sum_j n_{i,j}^{(\mathrm{P})}$. It follows that

$$P\big(D_{\mathrm{pa}_{\mathscr{X}}} \mid \Lambda\big) = \overbrace{\prod_{i=1}^{k}}^{\text{Principle 4}} \overbrace{\frac{1}{\binom{\gamma_i+J_{\mathrm{P}}-1}{J_{\mathrm{P}}-1}} \frac{1}{\frac{\gamma_i!}{n_{i,1}^{(\mathrm{P})}!\, n_{i,2}^{(\mathrm{P})}! \cdots n_{i,J_{\mathrm{P}}}^{(\mathrm{P})}!}}}^{\text{Principle 2}}. \tag{7}$$

Both factors on the right hand side come from the second principle: all distributions of values of the parents of $\mathscr{X}$ in a given interval are equiprobable. The fourth principle is that the distributions for each interval are independent, so the factors can be multiplied together. Note that the combinatorics of factors $\binom{\gamma_i+J_{\mathrm{P}}-1}{J_{\mathrm{P}}-1}$ and $\gamma_i!/n_{i,1}^{(\mathrm{P})}! \cdots n_{i,J_{\mathrm{P}}}^{(\mathrm{P})}!$ are explained in the MODL paper (Boullé, 2006).

3.1.2 Evaluation of $P(D_{C_j} \mid \Lambda, D_{\boldsymbol{S}_j})$

Let J_{C_j} be the number of instantiations of the jth child of $\mathscr{X}$, let $J_{\boldsymbol{S}}^{(j)}$ be the number of instantiations of the jth spouse set $\boldsymbol{S}_j$, and let $n_{i,m,\ell}^{(j)}$ be the number of instances of $\mathscr{X}$ in the ith discretization interval of Λ given the mth instantiation of C_j and the ℓth instantiation of $\boldsymbol{S}_j$. Let $n_{i,\ell}^{(j)} = \sum_{m=1}^{J_j} n_{i,m,\ell}^{(j)}$. Note that $\gamma_i = \sum_\ell n_{i,\ell}^{(j)}$ for all j. It follows that

$$P(D_{C_j} \mid \Lambda, D_{\boldsymbol{S}_j}) = \overbrace{\prod_{i=1}^{k}\prod_{\ell=1}^{J_{\boldsymbol{S}}^{(j)}}}^{\text{Principle 4}} \overbrace{\frac{1}{\binom{n_{i,\ell}^{(j)}+J_{C_j}-1}{J_{C_j}-1}} \frac{1}{\frac{n_{i,\ell}^{(j)}!}{n_{i,1,\ell}^{(j)}!\, n_{i,2,\ell}^{(j)}! \cdots n_{i,J_{C_j},\ell}^{(j)}!}}}^{\text{Principle 3}}. \tag{8}$$

Both factors on the right hand side come from the third principle: all distributions of values of C_j in a given interval and with a given value of S_j are equiprobable. According to the fourth prior, these distributions are independent from each other, and one can thus take their product. If $\boldsymbol{S}_j = \emptyset$, then Equation 8 is equivalent to

$$P(D_{C_j} \mid \Lambda, \boldsymbol{S}_j = \emptyset) = \overbrace{\prod_{i=1}^{k}}^{\text{Principle 4}} \overbrace{\frac{1}{\binom{\gamma_i+J_{C_j}-1}{J_{C_j}-1}} \frac{1}{\frac{\gamma_i!}{n_{i,1,\emptyset}^{(j)}!\, n_{i,2,\emptyset}^{(j)}! \cdots n_{i,J_{C_j},\emptyset}^{(j)}!}}}^{\text{Principle 3}}, \tag{9}$$

where $n_{i,m,\emptyset}^{(j)}$ is the number of instances of $\mathscr{X}$ in the ith discretization interval of Λ given the mth instantiation of C_j.

The objective function can be formulated given equations 4, 5, 7, 8 and 9. The log-inverse of $P(\Lambda) \cdot P(D_{-\mathscr{X}} \mid \Lambda)$ is minimized for computational convenience:

$$\begin{aligned}
&\sum_{i=1}^{k-1} -\ln\left(1 - \exp\left(-L \cdot \frac{x_{\lambda_i+1} - x_{\lambda_i}}{x_n - x_1}\right)\right) + \sum_{i=1}^{k} L \cdot \frac{x_{\lambda_i} - x_{\lambda_{i-1}+1}}{x_n - x_1} + \\
&\sum_{i=1}^{k}\left[\ln\binom{\gamma_i + J_{\mathrm{P}} - 1}{J_{\mathrm{P}} - 1} + \ln\left(\frac{\gamma_i!}{n_{i,1}^{(\mathrm{P})}!\, n_{i,2}^{(\mathrm{P})}! \cdots n_{i,J_{\mathrm{P}}}^{(\mathrm{P})}!}\right)\right] + \\
&\sum_{j=1}^{n_c}\sum_{i=1}^{k}\sum_{\ell=1}^{J_{\boldsymbol{S}}^{(j)}}\left[\ln\binom{n_{i,\ell}^{(j)} + J_{C_j} - 1}{J_{C_j} - 1} + \ln\left(\frac{n_{i,\ell}^{(j)}!}{n_{i,1,\ell}^{(j)}!\, n_{i,2,\ell}^{(j)}! \cdots n_{i,J_{C_j},\ell}^{(j)}!}\right)\right]
\end{aligned} \tag{10}$$

3.2 Single-Variable Discretization Algorithm

The procedure used to minimize the objective function involves dynamic programming. Note that because the objective function is cumulative over intervals, if a partition $\Lambda = \langle \gamma_1, \gamma_2, \ldots, \gamma_k \rangle$ of $\mathscr{X}$ is an optimal discretization policy, then any subinterval is optimal for the corresponding subproblem. It follows that dynamic programming can be used to solve the optimization problem exactly.

Precomputation reduces runtime. Hence, $h(u,v)$ is computed first for each interval $\gamma_q = \gamma_q^{(u,v)}$ starting from x_u to x_v for all u, v satisfying $u \leq v$:

$$\begin{aligned} h(u,v) = \ln \binom{\gamma_q + J_{\mathrm{P}} - 1}{J_{\mathrm{P}} - 1} + \ln \left(\frac{\gamma_q!}{n_{q,1}^{(\mathrm{P})}! n_{q,2}^{(\mathrm{P})}! \cdots n_{q,J_p}^{(\mathrm{P})}!} \right) \\ + \sum_{j=1}^{n_c} \sum_{\ell=1}^{J_S^{(j)}} \left[\ln \binom{n_{i,\ell}^{(j)} + J_{C_j} - 1}{J_{C_j} - 1} + \ln \left(\frac{n_{i,\ell}^{(j)}!}{n_{i,1,\ell}^{(j)}! \, n_{i,2,\ell}^{(j)}! \, \cdots \, n_{i,J_{C_j},\ell}^{(j)}!} \right) \right] \end{aligned} \tag{11}$$

The calculation of $h(u,v)$ for all $u \leq v$ has a $O\big(n_c \cdot L^{n_s} \cdot n^2 + L^{n_p} \cdot n^2\big)$ runtime, where n_c and n_p are the number of child and parent variables respectively, L is the largest cardinality of variables in X's Markov blanket, and $n_s = \max_j |\operatorname{pa}_{C_j}|$. The n^2 factor in the runtime comes from the fact that both x_u and x_v have to run over all possible values of $\mathscr{X}$, which has n instantiations.

The optimization problem over Equation 10 can now be solved. The dynamic programming procedure is shown in Algorithm 1. It takes three inputs: $\mathscr{X}$, the continuous variable; D, the joint data instances over all variables sorted in ascending order according to D_X; and G, the network structure. The runtime of Algorithm 1 is also $O\big(n_c \cdot L^{n_s} \cdot n^2 + L^{n_p} \cdot n^2\big)$ because the runtime of the dynamic programming procedure is less than the runtime for computing $h(u,v)$. As will be discussed in Section 6, the MDL discretization method has a runtime of $O\big(n^3 + (n_c \cdot L^{n_s} + L^{n_p}) \cdot n^2\big)$, which includes an extra $O\big(n^3\big)$ term. The Bayesian discretization method is quadratic in the sample count n, whereas the MDL discretization method is cubic.

Algorithm 1 is guaranteed to be optimal. For faster methods with suboptimal results, see Boullé (2006).

4. Multi-variable Discretization

The single-variable discretization method can be extended to Bayesian networks with multiple continuous variables by iteratively discretizing individual variables. The discretization process for a single variable requires that all other variables be discrete. The iterative approach uses an initial discretization policy in order to start the process which assigns k equal-width intervals to each continuous variable, where k is the largest number of intervals of initially discrete variables in the network.

After the initial discretization, the one-variable discretization method is iteratively applied over each continuous variable in reverse topological order, from the leaves to the root. Reverse topological order has the advantage of relying on fewer initial discretizations of the continuous variables during the first pass. For example, in the network of Figure 2, if S_2 is the only discrete variable, then the discretization of P_1 involves both P_2 and X, whereas the discretization of C_1 only involves X.

The algorithm is terminated when the number of discretization intervals and their associated edges converge for all variables, and a maximum number of complete passes is enforced to prevent infinite iterations when convergence does not occur. The algorithm typically converges within a few passes when tested on real-world data.

Algorithm 1 Discretization of one continuous variable in a Bayesian network

function DiscretizeOne(D, G, $\mathscr{X}$)
  $H \leftarrow$ an $n \times n$ matrix such that $H[u, v] = h(u, v)$; can be precomputed
  $L \leftarrow$ the largest cardinality over all discrete variables in the Markov blanket of $\mathscr{X}$
  $S[i] \leftarrow$ the optimal objective value computed over samples 1 to s_i
  $\Lambda[i] \leftarrow$ the discretization policy for the subproblem over samples 1 to s_i
  $W[i] \leftarrow -\ln\Big[1 - \exp\Big(-L \cdot \frac{x_{s_i+1} - x_{s_i}}{u_m - u_1}\Big)\Big]$ for $i \in [1, m-1]$ and $W[m] \leftarrow 0$
  for $v \leftarrow 1$ to m **do**
    if $v = 1$ **then**
      $S[v] \leftarrow H(1, s_v) + L[v]$
      $\Lambda[v] \leftarrow \{(u_v + u_{v+1})/2\}$
    else
      $\hat{S}, \hat{u} \leftarrow \infty, 0$
      DiscEdge $\leftarrow \infty$
      for $u \leftarrow 1$ to v **do**
        if $u = v$ **then**
          $\tilde{S} \leftarrow W[v] + H(1, s_v) + L \cdot \frac{u_v - u_1}{u_m - u_1}$
        else
          $\tilde{S} \leftarrow W[v] + H(s_u + 1, s_v) + L \cdot \frac{u_v - u_{u+1}}{u_m - u_1} + S[u]$
        if $\tilde{S} < \hat{S}$ **then**
          $\hat{S}, \hat{u} \leftarrow \tilde{S}, u$
          DiscEdge $\leftarrow (x_{s_u} + x_{s_u+1})/2$
      $S[v] \leftarrow \hat{S}$
      $\Lambda[v] \leftarrow \Lambda[\hat{u}] \cup \{\text{DiscEdge}\}$
  return $\Lambda[m]$

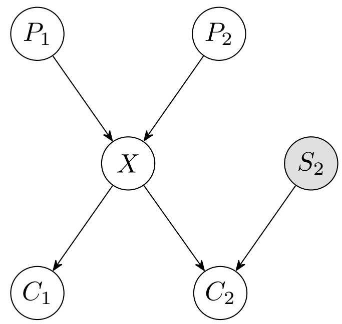

Figure 2: An example network.

The pseudocode for the multi-variable discretization procedure is shown in Algorithm 2. It requires four inputs: D, a dataset of samples from the joint distribution; G, the fixed network structure; $\boldsymbol{\mathscr{X}}$, the set of all continuous variables in reverse topological order, and $\hat{n}_{\text{cycle}}$, an upper bound on the number of complete passes.

Algorithm 2 Discretization of multiple continuous variables

function DiscretizeAll(D, G, $\boldsymbol{\mathscr{X}}$, $\hat{n}_{\text{cycle}}$)
    $\Lambda_{\boldsymbol{\mathscr{X}}} \leftarrow$ the discretization policies for each $\mathscr{X}$ in $\boldsymbol{\mathscr{X}}$
    $D^* \leftarrow$ the dataset D discretized according to $\Lambda_{\boldsymbol{\mathscr{X}}}$
    $k \leftarrow$ Max($\{|X|$ s.t. X does not have corresponding $\mathscr{X}$ in $\boldsymbol{\mathscr{X}}\}$)
    for $(\mathscr{X}, X)$ such that $\mathscr{X} \in \boldsymbol{\mathscr{X}}$ and X is the discretized version of $\mathscr{X}$ **do**
        $\Lambda_{\mathscr{X}} \leftarrow$ equal-width discretization of $\mathscr{X}$ with k intervals
        $D^*_X \leftarrow \Lambda_{\mathscr{X}}(D_{\mathscr{X}})$
    $n_{\text{cycle}} \leftarrow 0$
    while $\Lambda_{\boldsymbol{\mathscr{X}}}$ has not converged **and** $n_{\text{cycle}} \leq \hat{n}_{\text{cycle}}$ **do**
        increment n_{cycle}
        for $(\mathscr{X}, X)$ such that $\mathscr{X} \in \boldsymbol{\mathscr{X}}$ **do**
            $D^*_X \leftarrow D_{\mathscr{X}}$
            $\Lambda_{\mathscr{X}} \leftarrow$ DiscretizeOne($D^*, G, \mathscr{X}$)
            $D^*_X \leftarrow \Lambda_{\mathscr{X}}(D_{\mathscr{X}})$
    return $\Lambda_{\boldsymbol{\mathscr{X}}}$

5. Combining Discretization with Structure Learning

It is often necessary to infer the structure of a Bayesian network from data. Three common approaches to Bayesian network structure learning are constraint-based, score-based, and Bayesian model averaging (Koller & Friedman, 2009). This work uses the K2 structure learning algorithm (Cooper & Herskovits, 1992), a frequently used score-based structure learning method. Score-based structure learning methods over discrete variables commonly evaluate candidate structures according to their likelihood against a dataset D. In practice, one maximizes the log-likelihood, also known as the Bayesian score, $\ln P(G \mid D)$ (Cooper & Herskovits, 1992).

For a Bayesian network over N discrete and discretized variables $X_{1:N}$, r_i represents the number of instantiations of X_i, and q_i represents the number of instantiations of the parents of X_i. If X_i has no parents, then $q_i = 1$. The Bayesian score is:

$$\ln P(G \mid D) = \ln P(G) + \sum_{i=1}^{N} \sum_{j=1}^{q_i} \ln \left(\frac{\Gamma\left(\alpha_{ij}^{(0)}\right)}{\Gamma\left(\alpha_{ij}^{(0)} + \beta_{ij}^{(0)}\right)} \right) + \sum_{k=1}^{r_i} \ln \left(\frac{\Gamma\left(\alpha_{ij}^{(k)} + \beta_{ij}^{(k)}\right)}{\Gamma\left(\alpha_{ij}^{(k)}\right)} \right), \tag{12}$$

where Γ is the gamma function, $\alpha_{ij}^{(k)}$ is a Dirichlet parameter and $\beta_{ij}^{(k)}$ is the observed sample count for the kth instantiation of X_i and the jth instantiation of pa_{X_i}. In the equation above,

$$\alpha_{ij}^{(0)} = \sum_{k=1}^{r_i} \alpha_{ij}^{(k)} \qquad\qquad \beta_{ij}^{(0)} = \sum_{k=1}^{r_i} \beta_{ij}^{(k)}. \tag{13}$$

The space of acyclic graphs is superexponential in the number of nodes; it is common to rely on heuristic search strategies (Koller & Friedman, 2009). The K2 algorithm assumes a given topological ordering of variables and greedily adds parents to nodes to maximally increase the Bayesian score. A fixed ordering ensures acyclicity but does not guarantee a globally optimal network structure. K2 is typically run multiple times with different topological orderings, and the network with the highest likelihood is retained.

Traditional Bayesian structure learning algorithms require discretized data, whereas the proposed discretization algorithm requires a known network structure. The discretization methods can be combined with the K2 structure learning algorithm (Cooper & Herskovits, 1992) to simultaneously perform Bayesian structure learning and discretization of continuous variables.

The proposed algorithm alternates between K2 structure learning and discretization. The dataset is initially discretized as was described in Section 4, and K2 is run to obtain an initial network structure. The affected continuous variables are rediscretized every time an edge is added by K2. The resulting discretization policies are used to update the discretized dataset, and the next step of the K2 algorithm is executed. This cycle is repeated until the K2 algorithm converges.

This procedure is given in Algorithm 3, and takes five inputs: D, a dataset of samples from the joint distribution; $\boldsymbol{\mathcal{X}}$, the set of all continuous variables; `order`, a permutation of the variables in D; $\hat{n}_{\text{parent}}$, an upper bound on the number of parents per node; and $\hat{n}_{\text{cycle}}$, an upper bound on the number of complete passes. The upper bound on the number of parents is common practice, and is used to prevent computing conditional distributions with excessively large parameter sets. The function g in Algorithm 3 computes a component of the Bayesian score (Equation 12):

$$g\left(X_i, \mathrm{pa}_{X_i}\right) = \sum_{j=1}^{q_i} \ln \left(\frac{\Gamma\left(\alpha_{ij}^{(0)}\right)}{\Gamma\left(\alpha_{ij}^{(0)} + \beta_{ij}^{(0)}\right)} \right) + \sum_{k=1}^{r_i} \ln \left(\frac{\Gamma\left(\alpha_{ij}^{(k)} + \beta_{ij}^{(k)}\right)}{\Gamma\left(\alpha_{ij}^{(k)}\right)} \right). \tag{14}$$

It is common practice to run K2 multiple times with different variable permutations and to then choose the structure with the highest score. As such, Algorithm 3 is run multiple times with different variable permutations and the discretized Bayesian network with the highest score is retained.

Algorithm 3 Learning a discrete-valued Bayesian network

function LearnDVBN(D, $\boldsymbol{\mathscr{X}}$, `order`, $\hat{n}_{\text{parent}}$, $\hat{n}_{\text{cycle}}$)
    $N \leftarrow$ the number of variables in the Bayesian network
    $k \leftarrow \textsc{Max}\{\|v\|, v \notin C\}$
    $\Lambda_{\boldsymbol{\mathscr{X}}} \leftarrow$ the discretization policies for each $\mathscr{X}$ in $\boldsymbol{\mathscr{X}}$
    $D^* \leftarrow$ the dataset D discretized according to $\Lambda_{\boldsymbol{\mathscr{X}}}$
    $G \leftarrow$ the initially edgeless graph structure
    for $(\mathscr{X}, X)$ such that $\mathscr{X} \in \boldsymbol{\mathscr{X}}$ and X is the discretized version of $\mathscr{X}$ **do**
        $\Lambda_{\mathscr{X}} \leftarrow$ equal-width discretization of $\mathscr{X}$ with k intervals
        $D^*_X \leftarrow \Lambda_{\mathscr{X}}(D_{\mathscr{X}})$
    for $i \leftarrow 1$ to N **do**
        $P_{\text{old}} \leftarrow g(X_i, \text{pa}_{X_i})$ (Equation 14)
        OKToProceed $\leftarrow$ **true**
        while OKToProceed **and** $\| \text{pa}_{X_i} \| < \hat{n}_{\text{parent}}$ **do**
            $Y \leftarrow$ an element from the set `order`$[1 : i] \backslash (\text{pa}_X)$
            $P_{\text{new}} \leftarrow \big(g(X_i, \text{pa}_{X_i}) \cup (Y)\big)$
            if $P_{\text{new}} > P_{\text{old}}$ **then**
                $P_{\text{old}} \leftarrow P_{\text{new}}$
                $\text{pa}_{X_i} \leftarrow \big(\text{pa}_{X_i}\big) \cup (Y)$
                $\boldsymbol{\mathscr{X}'} \leftarrow \boldsymbol{\mathscr{X}}$ sorted in reverse-topological order given G
                $\Lambda \leftarrow$ DiscretizeAll(D, G, $\boldsymbol{\mathscr{X}'}$, $\hat{n}_{\text{cycle}}$) (see Algorithm 2)
                $D^* \leftarrow \Lambda(D_{\boldsymbol{\mathscr{X}}})$
            else
                OKToProceed $\leftarrow$ **false**
    return G, Λ

6. Experiments

This section describes experiments conducted to evaluate the Bayesian discretization method. All experiments were run on datasets from the publicly available University of California, Irvine machine learning repository (Lichman, 2013). Variables are labeled alphabetically in the order given on the dataset information webpage. In the figures that follow, shaded nodes correspond to initially discrete variables and the subscripts indicate the number of discrete instantiations.

Two experiments were conducted on each dataset. The first experiment compares the performance of the Bayesian and MDL discretization methods on a known Bayesian network structure. The structure was obtained by discretizing each continuous variable into k uniform-width intervals, where k is the median number of instantiations of the discrete variables, and using the structure with the highest Bayesian score from 1000 runs of the K2 algorithm with random topological orderings. The second experiment compares the same methods applied when simultaneously discretizing and learning network structure.

The discretizations are compared using the mean cross validated log-likelihood of the data D given the Bayesian network B and discretization policies $\Lambda_{\boldsymbol{\mathscr{X}}}$. Note that all log-likelihoods shown have been normalized by the number of data samples. The log-likelihood has two components,

$$\ln p(D \mid B, \Lambda_{\boldsymbol{\mathscr{X}}}) = \ln P(D^* \mid B) + \ln p(D \mid \Lambda_{\boldsymbol{\mathscr{X}}}, D^*), \tag{15}$$

where D is the original test dataset and D^* is the test dataset discretized according to $\Lambda_{\boldsymbol{\mathscr{X}}}$.

The log-likelihood of the discretized dataset given the Bayesian network is

$$P(D^* \mid B) = \prod_{i=1}^{N} \prod_{j=1}^{q_i} \prod_{k=1}^{r_i} \left(\frac{\alpha_{ij}^{(k)} + \beta_{ij}^{(k)}}{\alpha_{ij}^{(0)} + \beta_{ij}^{(0)}} \right)^{\beta_{ij}^{*(k)}}, \tag{16}$$

where $\boldsymbol{\alpha}$ and $\boldsymbol{\beta}$ are the Dirichlet prior counts and observed counts in the training set for B, and $\boldsymbol{\beta}^*$ is the set of observed counts in the test set D^*. All experiments used a uniform Dirichlet prior of $\alpha_{ijk} = 1$ for all i, j, and k.

The log-likelihood of the original dataset given the discrete dataset is

$$\ln p(D \mid \Lambda_{\boldsymbol{\mathscr{X}}}, D^*) = \sum_{i=1}^{N} 1_{\{\mathscr{X}_i \in \boldsymbol{\mathscr{X}}\}} \cdot \sum_{k=1}^{r_i} \sum_{j=1}^{q_i} \beta_{ij}^{*(k)} \ln \left(\frac{1}{e_k^{\Lambda_{\mathscr{X}_i}} - e_{k-1}^{\Lambda_{\mathscr{X}_i}}} \right). \tag{17}$$

Note that $e_0^{\Lambda_{\mathscr{X}_i}} = \textsc{Min}(D_{\mathscr{X}_i})$ and $e_{r_i}^{\Lambda_{\mathscr{X}_i}} = \textsc{Max}(D_{\mathscr{X}_i})$. The mean cross-validated log-likelihood is the mean log-likelihood on the witheld dataset among cross-validation folds, and acts as an estimate of generalization error. Ten folds were used in each experiment.

The method for computing the MDL discretization policy is similar to the method for the Bayesian method. For a Bayesian network with a single continuous variable $\mathscr{X}$, the

MDL objective function is

$$\begin{aligned} &\frac{1}{2}\ln(n)\left\{|\mathrm{pa}_{\mathscr{X}}|(|X|-1)+\sum_{j,\mathscr{X}\in\mathrm{pa}_{X_j}}|\mathrm{pa}_{X_j}|(|X_j|-1)\right\}+\ln(|X|) \\ &+(m-1)H\left(\frac{|X|-1}{m-1}\right)-n\cdot\left[I(X,\mathrm{pa}_{\mathscr{X}})+\sum_{j,\mathscr{X}\in\mathrm{pa}_{X_j}}I(X_j,\mathrm{pa}_{X_j})\right], \end{aligned} \tag{18}$$

where $I(A, B)$ is the mutual information between two discrete variable sets A and B, $H(p) = -p\ln(p) - (1-p)\ln(1-p)$, and X is the discretized version of $\mathscr{X}$.

The global minimum of Equation 18 can be found using dynamic programming. For a given Bayesian network, the first three terms only depend on the number of discretization intervals $|X|$ and the fourth term is cumulative over the intervals. Hence, if $\Lambda = \langle\gamma_1, \gamma_2, \ldots, \gamma_k\rangle$ is an MDL optimal discretization policy with k intervals, then $\Lambda' = \langle\gamma_1, \gamma_2, \ldots, \gamma_{k-1}\rangle$ is an MDL optimal discretization policy for the corresponding subproblem with $k-1$ intervals. This procedure takes runtime $O\big(n^3 + (n_c \cdot L^{n_s} + L^{n_p}) \cdot n^2\big)$. All variables are defined in Section 3.2. Therefore, the runtime of the proposed method in this paper is less than the MDL discretization method by $O\big(n^3\big)$, since the former has a more efficient form of dynamical programming. There also exist faster methods of MDL discretization, but they lead to suboptimal results (Friedman & Goldszmidt, 1996).

Note that the preliminary discretization and the discretization order of variables can be arbitrary for the MDL discretization method, as stated in the original work (Friedman & Goldszmidt, 1996). In order to make a fair comparison between the Bayesian and MDL method, the preliminary discretization and the discretization order follows the same procedure as in Section 4.

6.1 Dataset 1: Auto MPG

The Auto MPG dataset contains variables related to the fuel consumption of automobiles in urban driving. The dataset has 392 samples over eight variables, not including six instances with missing data. Three variables are discrete: B, G, and H, with 5, 13, and 3 instantiations respectively.

6.1.1 Discretization with Fixed Structure

The Bayesian and MDL discretization methods were tested on the Auto MPG data using the network shown in Figure 3. This structure was obtained by initially discretizing each continuous variable into five uniform-width intervals, where five is the median cardinality of the discrete variables, and then taking the structure with highest likelihood from 1000 runs of K2.

Table 3 lists the discretization edges and mean log-likelihoods under 10-fold cross validation of the discrete Bayesian network resulting from the two discretization methods. The MDL method does not produce any discretization edges, assigning one continuous interval to each continuous variable, and produces the result with the lower likelihood. The reason

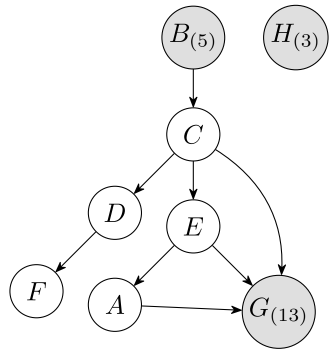

Figure 3: Bayesian network structure obtained from running K2 on the initially uniformly discretized Auto MPG dataset.

why MDL produces fewer discretization edges is discussed in Section 6.5. Two examples that MDL produces comparable results to the Bayesian method are given in Appendix B.

Table 3: Discretization result of the Auto MPG dataset with fixed structure from Figure 3. The first five rows list the discretization edges and the last row lists the mean cross-validated log-likelihood; positive values are better.

Variable	Bayesian	MDL
A	15.25, 17.65, 20.90, 25.65, 28.90	-
C	70.5, 93.5, 109.0, 159.5, 259.0, 284.5	-
D	71.5, 99.0, 127.0	-
E	2115, 2481, 2960, 3658	-
F	12.35, 13.75, 16.05, 22.85	-
Log-Likelihood	-26.04	-30.40

Figure 4 shows the marginal probability density for variables A and C under the Bayesian discretization policy overlaid with the original Auto MPG data. The resulting probability density is a good match to the original data.

6.1.2 Discretization while Learning Structure

In this experiment, the network structure was not fixed in advance and was learned simultaneously with the discretization policies. Figure 5 shows a learned Bayesian network structure and the corresponding numbers of intervals after discretization for each continuous variable. This result was obtained by running Algorithm 3 fifty times using the Bayesian method and choosing the structure with the highest K2 score (Equation 12).

Figure 6 compares the Bayesian discretizaton policy for variables A and C in the learned network with the original Auto MPG data. The color of a discretized region indicates the marginal probability density of a sample from $P(A, C)$ being drawn from that region. Although there are fewer discretization edges for A and C in the learned network, the marginal distribution is still captured. The discretization policy will vary as the network structure changes, and it still produces high-quality discretizations.

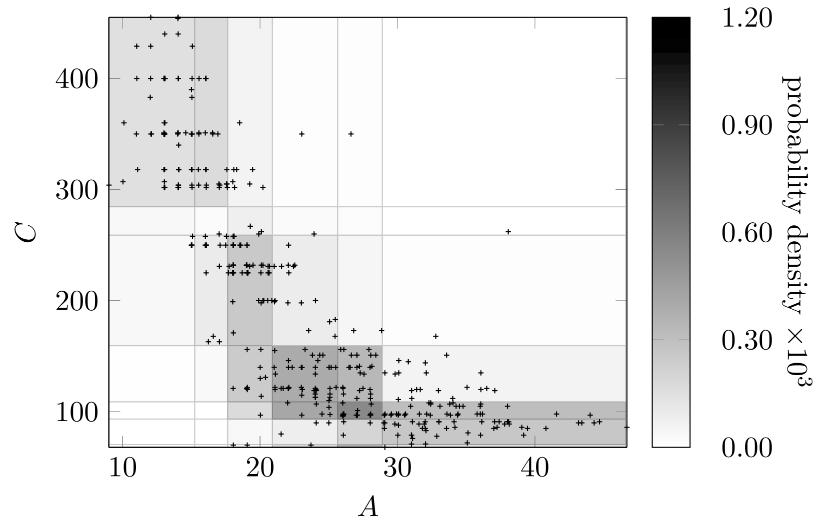

Figure 4: Comparison of the Bayesian discretization policy for variables A and C to the original Auto MPG data learned with a fixed network. The marginal probability density closely matches the data.

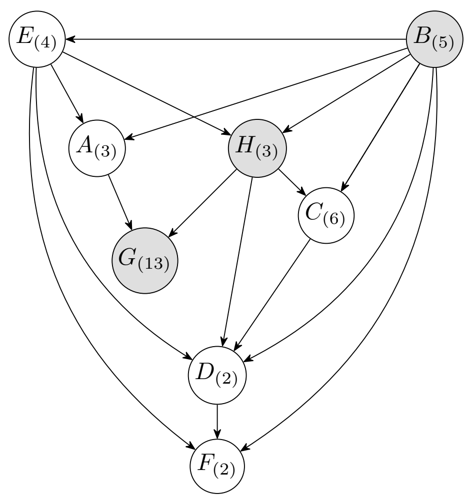

Figure 5: The discrete-valued Bayesian network learned from the Auto MPG dataset using the Bayesian method.

6.2 Dataset 2: Wine

The Wine dataset contains variables related to the chemical analysis of wines from three different Italian cultivars. The dataset has 178 samples over fourteen variables. Variable A is the only discrete variable and has three instantiations.

6.2.1 Discretization with Fixed Structure

The Bayesian and MDL discretization methods were tested on the Wine data using the network shown in Figure 7. This structure was obtained by initially discretizing each continuous variable into three uniform-width intervals, where three is the median cardinality

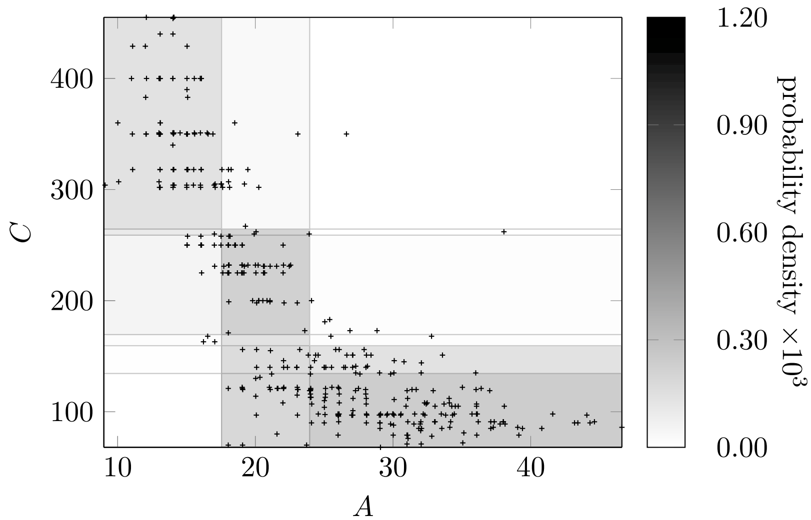

Figure 6: Comparison of the Bayesian discretization policy for variables A and C to the original Auto MPG data learned simultaneously with the network structure. Although the number of discretization edges is less than those in Figure 4, the probability distribution still closely matches the original data.

of the discrete variables, and then taking the structure with highest likelihood from 1000 runs of K2.

Table 4 lists the discretization edges and mean log-likelihoods under 10-fold cross validation of the Bayesian network resulting from each discretization method. The Bayesian method outperforms the MDL method in likelihood by a significant margin. The MDL method creates significantly fewer discretization edges.

Some discretization edges appear in both discretization methods, such as 1.42 and 2.35 for variable C and 0.785 for variable L. MDL can find some important discretization edges, but it is not sensitive to find more edges. Actually, as shown later in Section 6.5, while the MDL and Bayesian methods both produce the optimal discretization as the training data grows, the Bayesian method can produce these discretizations with fewer data samples.

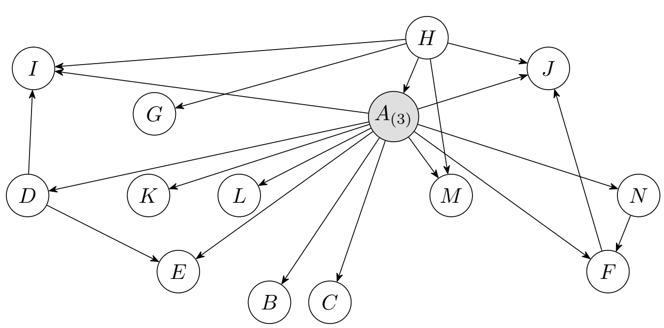

Figure 7: Bayesian network structure obtained from running K2 on the initially uniformly discretized Wine dataset.

Figure 8 compares the Bayesian and MDL discretizaton policies for variables E and K with the original Wine data. The discretization edges 17.9 for E and 34.6 for K appear

Table 4: Results from discretization of the Wine MPG dataset with fixed structure. Bold discretization edges were identified by both methods.

Variable	Bayesian	MDL
B	12.745, 13.54	12.78
C	**1.42**, **2.235**	**1.42**, **2.235**
D	2.03, 2.605, 3.07	-
E	**17.9**, 23.25	**17.9**
F	88.5, 135.0	-
G	2.105, 2.58, 3.01	-
H	0.975, 1.885, 2.31, 3.355	-
I	**0.395**	**0.395**
J	1.185, 1.655	-
K	**3.46**, 4.85, 7.4	**3.46**, 7.55
L	**0.785**, 1.005, 1.295	**0.785**
M	2.475	2.115, 2.505
N	476.0, 716.0, 900.5	-
Log-Likelihood	-19.94	-23.60

in both plots. The MDL method does not use enough intervals for discretization. Relative sensitivities of each method to the input data is discussed in Section 6.5.

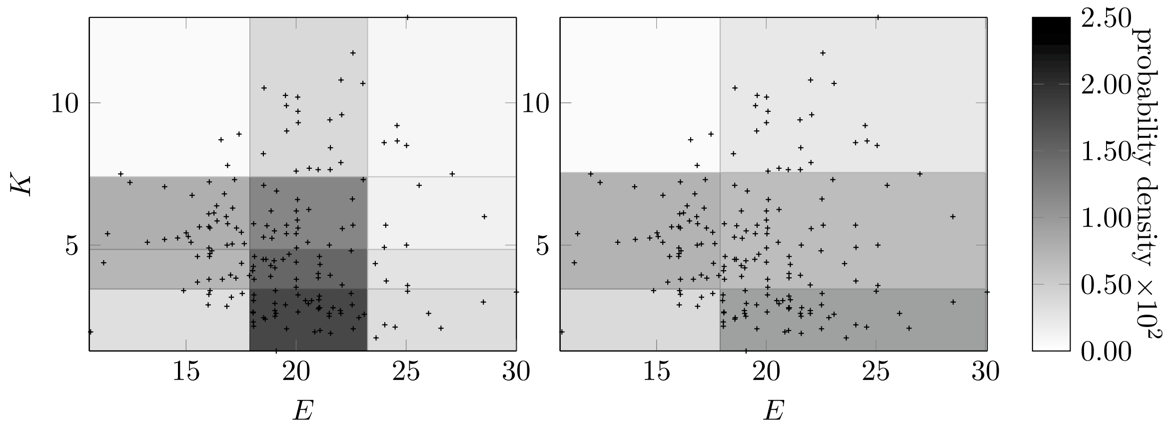

Figure 8: Comparison of the discretization policies for variables E and K obtained using the Bayesian and MDL methods.

6.2.2 Discretization while Learning Structure

Figure 9 is the discrete-valued Bayesian network learned from the Wine dataset, obtained by running Algorithm 3 fifty times. A comparison of Figures 9 and 7 show that the Bayesian network learned during the discretization process has more edges than the network learned on the initially discretized data. When a network is learned along with discretization, the algorithm has more freedom to adjust the structure and the discretization policy simultaneously to identify useful correlations and produce a denser structure.

Figure 10 shows the discretization policy for variables E and K obtained with the Bayesian method. The discretization edge $E = 17.9$ also appears in both discretization policies from the fixed network (Figure 8). This suggests that discretization edges can be robust against network structure. Furthermore, the discretization edge at $E = 23.5$ in the fixed-structure case is missing in the learned structure. This is caused by E having twice as many parents in the learned network structure. The more parents a variable has, the fewer discretization intervals it can to support, as the number of sufficient statistics required to define the resulting distribution increases exponentially with the number of parents.

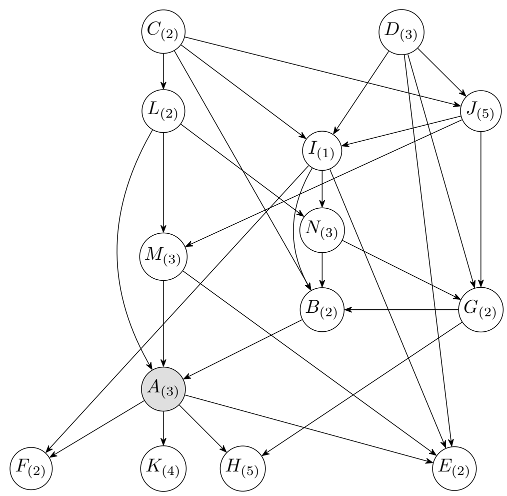

Figure 9: The discrete-valued Bayesian network learned from the Wine dataset using the Bayesian method.

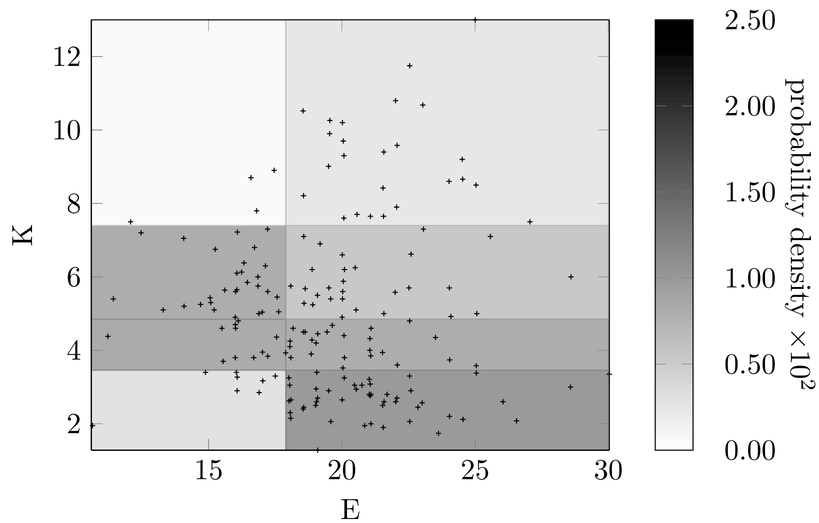

Figure 10: The discretization policy for variables E and K on the learned network. The discretization edge $E = 17.9$ also appears in the policies for the fixed network.

6.3 Dataset 3: Housing

The Housing dataset contains variables related to the values of houses in Boston suburbs. The dataset has 506 samples over fourteen variables. Only variables D and I are discrete-

Table 5: Discretization policy summary of the Housing dataset based on the fixed network structure shown. The first twelve rows show the numbers of discretization intervals and the last row is the mean cross-validated log-likelihood.

Variable	Bayesian	MDL
A	3	1
B	4	1
C	8	1
E	14	1
F	4	1
G	3	1
H	6	1
J	8	1
K	5	7
L	4	1
M	6	1
N	6	1
Log-Likelihood	−31.40	−43.20

valued, with 2 and 9 instantiations, respectively. Despite their being continuous, several variables in the Housing dataset possess many repeated values. The following experiments were conducted with a maximum of three parents per variable to prevent running out of memory when running MDL.

6.3.1 Discretization with Fixed Structure

The Bayesian network structure in Figure 11 was obtained by initially discretizing each continuous variable into five uniform-width intervals and then running the K2 algorithm 1000 times and choosing the network with the highest likelihood. Table 5 shows the numbers of interval after discretization for each continuous variable and the log-likelihood of the dataset based on each discretization method. The MDL method does not produce any discretization edges for most variables. The relative weighting of each method's objective function will be discussed in Section 6.5.

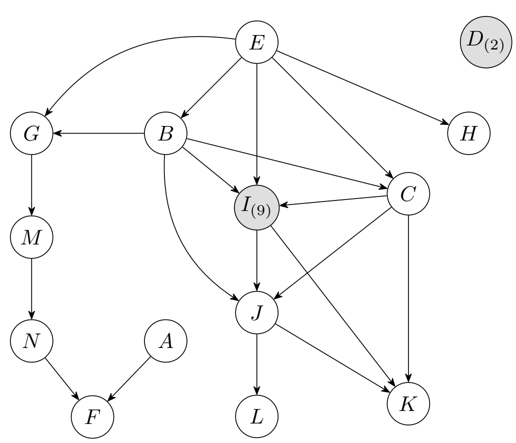

Figure 11: Bayesian network structure obtained from running K2 on the initially uniformly discretized Housing dataset.

Figures 12 and 13 show the discretization policy learned using the Bayesian approach on the fixed network shown in Figure 11. The scatter points in Figure 12 were jittered to show the quantity of repeated values for variables C and E. Each repeated point forms a single discrete region, thereby encouraging discretization. In contrast, the samples for H are well spread out, resulting in fewer discretization regions and larger discretization intervals.

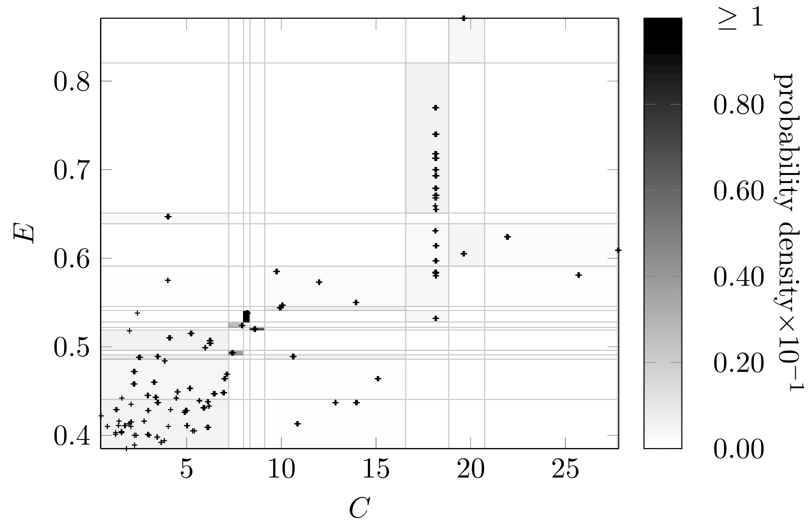

Figure 12: The discretization policies for variables C and E learned using the Bayesian approach on the fixed network. The scatter points were jittered to reveal the repeated values in the dataset. Values perfectly repeated in C and E lead to high correlation and concentrated densities in the marginal distribution.

6.3.2 Discretization while Learning Structure

Figure 14 shows a learned Bayesian network structure and the corresponding numbers of intervals after discretization for each continuous variable. Variable D is neighborless in both the learned and fixed networks. The continuous variables C, E, J and K, which are all connected through C, have many discretization intervals. Typically, when discretizing a variable, the expected number of discretization intervals is close to the highest cardinality among variables in its Markov blanket. This naturally leads to clusters of variables with many discretization intervals.

Figures 15 and 16 show the discretization result for variables in the network shown in Figure 14. Again, variable C and E have many discretization edges due to repeated values. Although the number of intervals after discretization on H is less the number in Figure 13, it still captures the distribution of the raw data along with the discretization edges for E.

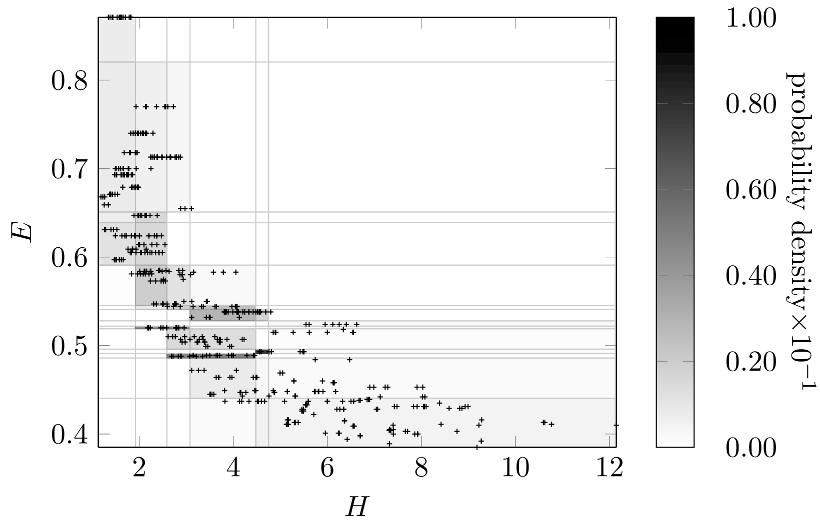

Figure 13: The discretization policies for variables H and E learned using the optimal Bayesian approach on the fixed network shown in Figure 11. The values for variable H are not as repeated, thus producing fewer discretization intervals than E.

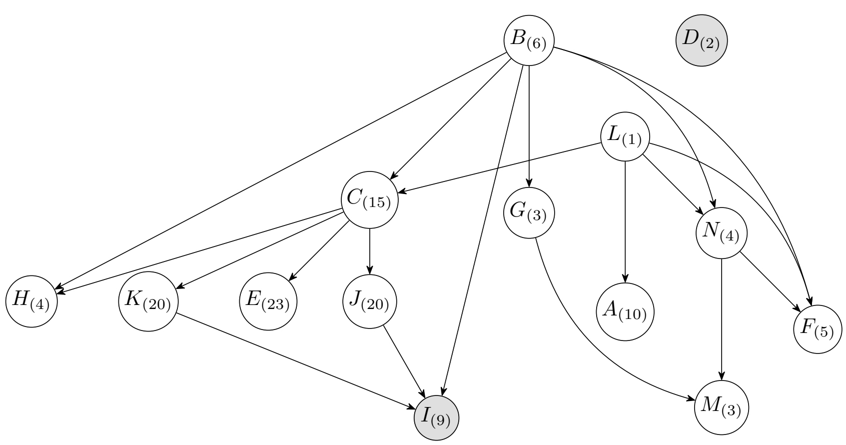

Figure 14: The discrete-valued Bayesian network learned from the Housing dataset using the optimal Bayesian method.

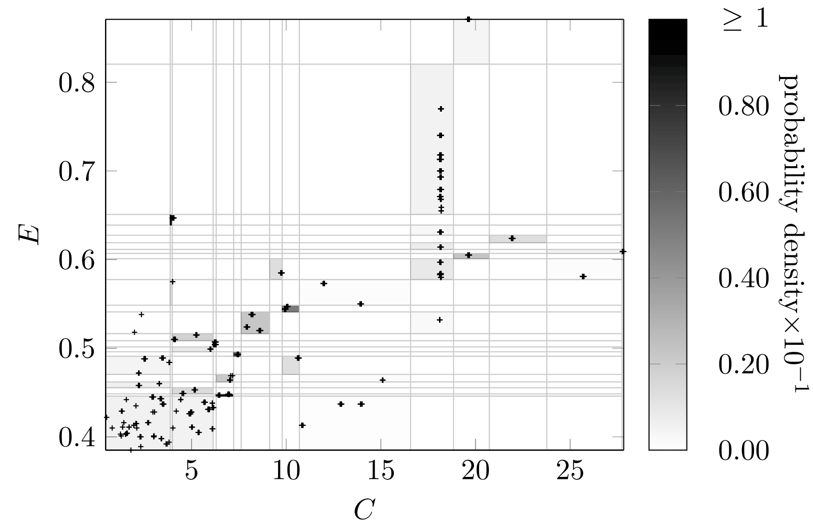

Figure 15: The discretization policy for variables C and E on the learned network in Figure 14. The large number of discretization edges are due to repeated values in C and E. The two regions at $(C, E) = (3.9, 0.65)$ and $(6.8, 0.44)$ are color-saturated due to the high concentration of repeated values.

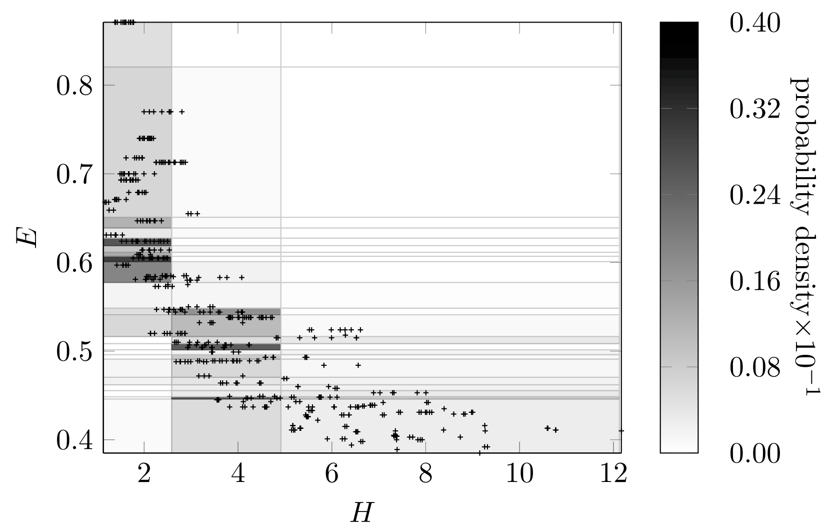

Figure 16: The discretization policy for variables H and E on the learned network in Figure 14. There are fewer discretization intervals for H than for the fixed network structure in Figure 13, but the discretization result still closely models the raw data.

6.4 Results on Additional Datasets

Six additional datasets (Lichman, 2013) are tested to compare the Bayesian and MDL discretization methods. The results are summarized in Table 6. Each row corresponds to a dataset and provides the number of samples n, the number of variables N, the fixed network structure, and the log-likelihood for each method. Fixed network structures are obtained by the same principle as with the previous three datasets and are represented by listing the parents for each variable. For example, in the first row, the representation $\{\emptyset, A, BE, AB, B, AH, BDG, B\}$ indicates that $\mathrm{pa}_A = \emptyset$, $\mathrm{pa}_B = A$, $\mathrm{pa}_c = \{B, E\}$, etc. The Bayesian discretization method has higher mean cross-validated log likelihood in all cases.

Table 6: Discretization results for several datasets, summarizing the number of samples, number of variables, network structure, and mean cross-validated log-likelihood under each discretization method. The Bayesian method consistently has higher likelihood.

Name	n	N	Network	Bayesian	MDL
Seeds	210	8	$\{\emptyset, A, BE, AB, B, AH, BDG, B\}$	−2.35	−5.71
Glasses	214	10	$\{F, EFH, FGH, AFH, AF, \emptyset, AF, F, \emptyset, \emptyset\}$	−1.62	−7.75
Heart	270	13	$\{H, M, J, EF, \emptyset, \emptyset, J, IK, C, \emptyset, IJ, A, I\}$	−25.81	−26.91
Liver Disorder	345	7	$\{\emptyset, EF, \emptyset, CE, C, CD, D\}$	−23.73	−26.57
Indian Liver Patient	583	10	$\{DEFI, E, \emptyset, \emptyset, DG, \emptyset, F, CGIJ, DEJ, \emptyset, \emptyset\}$	−34.89	−40.53
Banknote Authority	1372	5	$\{BD, D, AB, \emptyset, AD\}$	−8.99	−12.13

Finding optimal discretization policies according to Equation 10 via dynamic programming requires computation times on the order of days on a personal laptop on datasets with more than 100 variables. The runtime for the MDL approach is significantly worse. Approximation approaches are reasonable options for such large datasets, for example a greedy heuristic (Friedman & Goldszmidt, 1996). Approximations can also be applied during dynamic programming when solving Equation 10 (Boullé, 2006).

6.5 Discussion

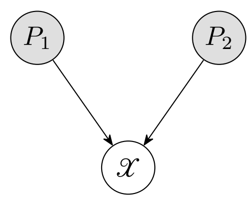

Figure 17: A simple Bayesian network used to demonstrate the sensitivity of each method.

To qualitatively assess the sensitivity of each method to the number and position of discretization intervals, consider the Bayesian network in Figure 17, where $\mathscr{X}$ is continuous and P_1 and P_2 are discrete. If $\mathrm{pa}_{\mathscr{X}} = \{P_1, P_2\}$ and $D_{\mathscr{X}} = \{1, 2, 3, \dots n\}$, then the corresponding objective functions for the discretization methods are:

$$
\begin{aligned}
f_{\mathrm{MDL}} &= \overbrace{Z_1 \cdot k + \ln(k) + \ln\binom{n+k-1}{k-1}}^{\text{Penalty Term}} + \overbrace{n \cdot I(X^*, \mathrm{pa}_X)}^{\text{Edge Position Term}} \\
f_{\mathrm{Bayesian}} &= \underbrace{Z_2 \cdot k + \sum_{i=1}^{k} \ln\binom{\gamma_i + J_{\mathrm{P}} - 1}{J_{\mathrm{P}} - 1}}_{\text{Penalty Term}} + \underbrace{\sum_{i=1}^{k} \ln\left(\frac{\gamma_i!}{n^{\mathrm{P}}_{i,1}! n^{\mathrm{P}}_{i,2}! \cdots n^{\mathrm{P}}_{i,J_{\mathrm{P}}}!}\right)}_{\text{Edge Position Term}}
\end{aligned}
\tag{19}
$$

Here, Z_1 and Z_2 are constant over discretizations, k is the number of discretization intervals, and $I(A, B) = \sum_{a,b} \hat{P}(a, b) \ln \frac{\hat{P}(a,b)}{\hat{P}(a)\hat{P}(b)}$ is the mutual information based on estimated probabilities. Note that the third term of f_{MDL} was approximated using $H(p)$ (see Equation 18) in the original work by Friedman and Goldszmidt (1996), but it is written here without that approximation.

The penalty terms tend to increase with the number of discretization intervals; the edge position terms vary with the position and number of the discretization edges. A policy with a larger number of discretization edges is only optimal if the edge position term varies enough with respect to the penalty term in order to produce a local minimum of sufficiently low value.

The value of the edge position term for MDL is primarily determined by the mutual information, which varies less severely than the corresponding terms in the Bayesian method. The MDL term uses empirical probability distributions based off of ratios of counts, whereas the Bayesian method uses factorial terms, and thus the MDL method varies less. The MDL method is therefore less sensitive to the discretization edges. This sensitivity gives rise to the relative performance of the two methods in the experiments conducted above.

To support this argument, we test a synthetic dataset with 800 samples using the network structure in Figure 17. Data samples are generated by the distribution described in Table 7: both P_1 and P_2 are fair coins. The continuous variable $\mathscr{X}$ is conditionally dependent on P_1 and P_2. Note that the symbol $\sim U[a, b]$ in Table 7 represents a uniform distribution from a to b. The optimal discretization policy on $\mathscr{X}$ has six edges, $\{2.0, 3.0, 4.0, 4.5, 5.0, 6.0\}$, since $\mathscr{X}$ takes numeric values from 1.0 to 7.5 and there are seven intervals corresponding to different joint distributions of P_1 and P_2.

Table 7: Probability distributions for the variables in Figure 17. P_1 and P_2 are discrete whereas $\mathscr{X}$ is continuous.

Variable	Distribution
P_1	$P(p_1^{(1)}) = 0.5$ $P(p_1^{(2)}) = 0.5$
P_2	$P(p_2^{(1)}) = 0.5$ $P(p_2^{(2)}) = 0.5$
$\mathscr{X}$	$p(x \mid p_1^{(1)}, p_2^{(1)}) \sim U[2, 6]$ $p(x \mid p_1^{(1)}, p_2^{(2)}) \sim U[1, 3]$ $p(x \mid p_1^{(2)}, p_2^{(1)}) \sim U[4, 5]$ $p(x \mid p_1^{(2)}, p_2^{(2)}) \sim U[4.5, 7.5]$

Figure 18 shows how the test likelihood and the number of discretization edges for $\mathscr{X}$ vary with respect to the number of training samples. A dataset of 800 samples is drawn from the synthetic distribution. The size of the subset used for training is varied, and the remaining samples are used as the test set. When the training set is small, it is difficult to recognize the underlying pattern and discretize the continuous variable $\mathscr{X}$ accordingly. In this case, both the likelihood of the training data and number of discretization edges are small. However, the Bayesian method is more sensitive to variation in the data and produces a distribution both with higher likelihood and more discretization edges. As the

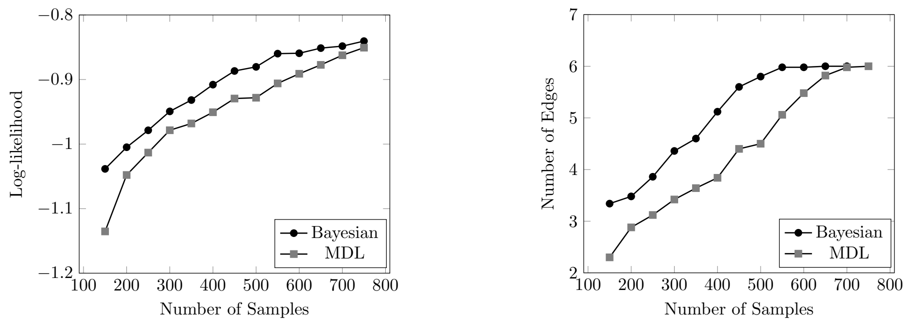

Figure 18: The average test log-likelihood and the average number of discretization edges for $\mathscr{X}$ as the number of training samples is varied. The complete dataset size is fixed at 800 samples, drawn randomly from the synthetic distribution. The size of the subset used for training is varied, and the remaining samples are used as the test set.

size of the training set grows, both methods converge towards the optimal discretization policy. The Bayesian method finds the six discretization edges faster than the MDL method and consistently outperforms in likelihood. The curves in Figure 18 demonstrate that the Bayesian method is indeed more sensitive to the dataset than the MDL method. The results in Figure 18 are averaged over one hundred trials.

7. Conclusion

This paper introduced a principled discretization method for continuous variables in Bayesian networks with quadratic complexity instead of the cubic complexity of other standard techniques. Empirical demonstrations show that the proposed method is superior to the state of the art. In addition, this paper shows how to incorporate existing methods into the structure learning process to discretize all continuous variables and simultaneously learn Bayesian network structures. The proposed method was incorporated and its superior performance was empirically demonstrated. Future work will investigate edge positions at locations other than the midpoints between samples and will extend the approach to cluster categorical variables with very many levels. All software is publicly available at `github.com/sisl/LearnDiscreteBayesNets.jl`.

Appendix A.

This section illustrates the advantage of structure-aware discretization methods over marginal discretization methods (Scott, 1979; Freedman & Diaconis, 1981; Knuth, 2013; Scargle et al., 2013). Marginal discretization methods ignore intervariable dependencies and can thus lead to suboptimal discretizations for the joint distribution. Consider a synthetic dataset with two variables, discrete A and continuous B, with 500 data samples sampled from the Bayesian network $A \to B$ described in Table 8. Following the experiment procedure in Section 6, both the MDL method and the proposed Bayesian approach find the true discretization edge for B near 1.0 and have log likelihood -0.023. They recover the discretization shown in Figure 19. The Bayesian Blocks method (Scargle et al., 2013), as with any marginal method, cannot distinguish between the different conditional distribu-

Variable	Distribution
A	$P(a^1) = 0.5$ $P(a^2) = 0.5$
B	$p(b \mid A = a^1) \sim U[0, 1]$ $p(b \mid A = a^2) \sim U[1, 2]$

Table 8: Probability distributions for $A \rightarrow B$, in which A is discrete and B is continuous. The marginal distribution for B is $U[0, 2]$.

tions and thus does not locate a discretization edge. The resulting log likelihood is worse, at -0.69. This example shows that structure-aware discretization methods can leverage additional information and thus produce better joint representations.

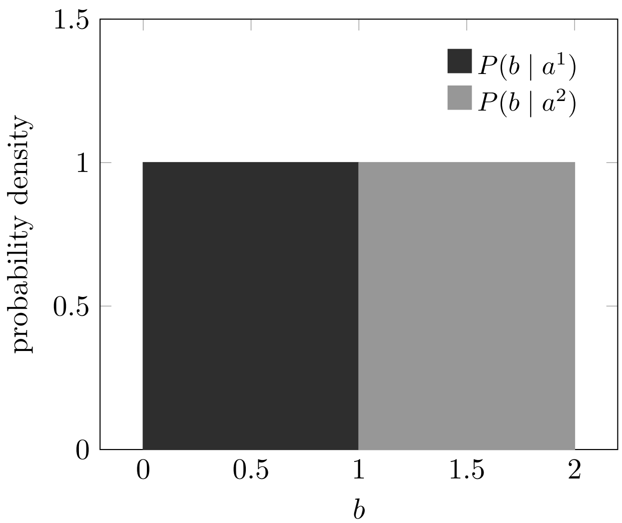

Figure 19: The resulting conditional probability distribution for B. Marginal methods see one uniform distribution.

Appendix B.

This section gives two examples of Bayesian networks where MDL produces comparable results to the Bayesian method. The first example used the Auto MPG dataset. Instead of testing on the network obtained using initially uniform discretization and structure learning, the tested network is obtained by Algorithm 3, which can be considered better-structured than the network in Figure 3. The second example is on the Iris dataset (Lichman, 2013). In the work of Friedman and Goldszmidt (1996), the Iris dataset was tested on the naive Bayes network structure and the prediction accuracy was shown. The same test is reproduced below to compare the Bayesian method to the MDL discretization method.

B.1 Auto MPG with Comparable MDL Performance

The Bayesian network structure in Figure 20 and the discretization results in Table 9 demonstrate that the MDL method does not always produce fewer discretization intervals than the Bayesian approach and that the MDL method can produce discretization policies with comparable likelihood. The Bayesian network in Figure 20 was obtained by running Algorithm 3 fifty times on the Auto MPG dataset with a maximum of two parents per variable. The network is structured more favorably with regards to discretization than the network in Figure 3, as the discretization policies were simultaneously learned with the network structure, resulting in higher data likelihood. Many edges appear in both discretization results and are bolded in Table 9.

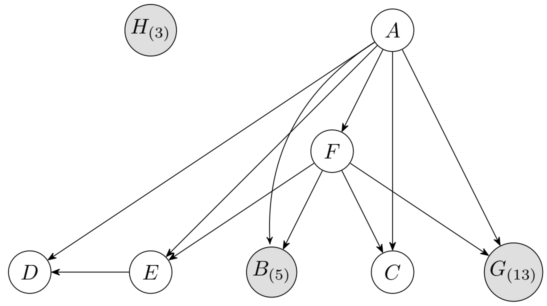

Figure 20: A Bayesian network structure for the Auto MPG dataset that leads to comparable discretization policies for the MDL and Bayesian methods.

Table 9: The discretization policies for the Auto MPG dataset with the fixed structure shown in Figure 20. The methods have the same discretization for variable A, and the log-likelihood for the Bayesian method is only slightly better than that for the MDL method.

Variable	Bayesian	MDL
A	**17.65**, **22.75**	**17.65**, **22.75**
C	70.5, **159.5**, **259.0**, 284.5	**159.5**, **259.0**
D	**99.0**, 127.0	84.5, **99.0**, 123.5
E	2764.5, 3030.0, 3657.5	-
F	14.05	-
Log-Likelihood	-25.94	-26.83

B.2 Reproducing MDL Results from the Literature

The Iris dataset contains variables related to the morphologic variation of three Iris flower species. The dataset has 150 samples over five variables. Only variables E is discrete-valued, and corresponds to the species category. The following experiment was conducted on the naive Bayes structure: variable E is the parent of variable A, B, C, and D. Table 10 shows the discretization edges, the prediction accuracy of the learned classifiers, and the mean log-likelihood. The discretization policy shown is for running the algorithms on the full dataset; the likelihood and prediction accuracies were found with 10-fold cross validation. The MDL discretization method and the Bayesian method have the same discretization result: all discretization edges coincide. In addition, both methods have 94% accuracy, which matches the number in the literature (Friedman & Goldszmidt, 1996). The Bayesian method likelihood is slightly higher across folds than the MDL method likelihood.

Table 10: The discretization policies for the Iris dataset with the naive Bayes structure. The two methods produce the same discretization policy on each continuous variable and have nearly identical 10-fold prediction accuracy and log-likelihood.

Variable	Bayesian	MDL
A	5.45, 6.15	5.45, 6.15
B	2.95, 3.35	2.95, 3.35
C	2.45, 4.75	2.45, 4.75
D	0.8, 1.75	0.8, 1.75
Prediction Accuracy	0.94	0.94
Log-Likelihood	−2.25	−2.50

References

Boullé, M. (2006). MODL: A Bayes optimal discretization method for continuous attributes. *Machine Learning*, *65*, 131–165.

Cooper, G. F., & Herskovits, E. (1992). A Bayesian method for the induction of probabilistic networks from data. *Machine Learning*, *9*(4), 309–347.

Dougherty, J., Kohavi, R., & Sahami, M. (1995). Supervised and unsupervised discretization of continuous features. In *International Conference on Machine Learning (ICML)*. Morgan Kaufman.

Druzdzel, M. J. (1999). SMILE: Structural modeling, inference, and learning engine and GeNIe: a development environment for graphical decision-theoretic models. In *National Conference on Artificial Intelligence (AAAI)*, pp. 902–903.

Fayyad, U., & Irani, K. (1993). Multi-interval discretization of continuous-valued attributes for classification learning. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pp. 1022–1027.

Freedman, D., & Diaconis, P. (1981). On the histogram as a density estimator: L2 theory. *Wahrscheinlichkeitstheorie und verwandte Gebiete*, *57*(4), 453–476.

Friedman, N., Geiger, D., & Goldszmidt, M. (1997). Bayesian network classifiers. *Machine Learning*, *29*, 131–163.

Friedman, N., & Goldszmidt, M. (1996). Discretizing continuous attributes while learning Bayesian networks. In *International Conference on Machine Learning (ICML)*. Morgan Kaufman.

Grünwald, P. D. (2007). *Minimum Description Length Principle*. MIT Press.

Holte, R. (1993). Very simple classification rules perform well on most commonly used datasets. *Machine Learning*, *11*, 63–90.

Ickstadt, K., Bornkamp, B., Grzegorczyk, M., Wieczorek, J., Sheriff, M. R., Grecco, H. E., & Zamir, E. (2010). Nonparametric Bayesian network. *Bayesian Statistics*, *9*, 283–316.

Kerber, R. (1992). ChiMerge: Discretization of numeric attributes. In *National Conference on Artificial Intelligence (AAAI)*, pp. 123–128. AAAI Press.

Knuth, K. H. (2013). Optimal data-based binning for histograms. *arXiv preprint*, *physics*(0605197), 1–30.

Kochenderfer, M. J., Edwards, M. W. M., Espindle, L. P., Kuchar, J. K., & Griffith, D. J. (2010). Airspace encounter models for estimating collision risk. *AIAA Journal of Guidance, Control, and Dynamics*, *33*(2), 487–499.

Kochenderfer, M. J., Holland, J. E., & Chryssanthacopoulos, J. P. (2012). Next-generation airborne collision avoidance system. *Lincoln Laboratory Journal*, *19*(1), 17–33.

Koller, D., & Friedman, N. (2009). *Probabilistic Graphical Models: Principles and Techniques*. MIT Press.

Kozlov, A., & Koller, D. (1997). Nonuniform dynamic discretization in hybrid networks. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 314–325.

Lichman, M. (2013). UCI machine learning repository. http://archive.ics.uci.edu/ml.

Lustgarten, J. L., Visweswaran, S., Gopalakrishnan, V., & Cooper, G. F. (2011). Application of an efficient Bayesian discretization method to biomedical data. *BMC Bioinformatics*, *12*(1), 309.

Monti, S., & Cooper, G. (1998). A multivariate discretization method for learning Bayesian networks from mixed data. In *Conference on Uncertainty in Artificial Intelligence (UAI)*, pp. 404–413.

Norsys (1992–2009). Netica Bayesian network software package. http://www.norsys.com.

Pearl, J. (1988). *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufman.

Rissanen, J. (1978). Modeling by shortest data description. *Automatica*, *14*(5), 465–471.

Scargle, J. D., Norris, J. P., Jackson, B., & Chiang, J. (2013). Studies in astronomical time series analysis. VI. Bayesian block representations. *The Astrophysical Journal*, *764*(2), 167.

Scott, D. W. (1979). On optimal and data-based histograms. *Journal of Statistical Software*, *66*(3), 605–610.

Scutari, M. (2010). Learning Bayesian networks with the bnlearn R package. *Journal of Statistical Software*, *35*(3), 1–22.

Steck, H., & Jaakkola, T. S. (2004). Predictive discretization during model selection. In *Pattern Recognition*, Vol. 3175 of *Lecture Notes in Computer Science*, pp. 1–8. Springer.

Vitányi, P., & Li, M. (2000). Minimum description length induction, Bayesianism, and Kolmogorov complexity. *IEEE Transactions on Information Theory*, *46*(2), 446–464.

Weiss, Y., & Freeman, W. T. (2001). Correctness of belief propagation in Gaussian graphical models of arbitrary topology. *Neural Computation*, *13*(10), 2173–2200.