Uni- and Multi-Dimensional Clustering Via Bayesian Networks

Omid Keivani and Jose M. Peña

Abstract This chapter discusses model based clustering via Bayesian networks. Both uni-dimensional and multi-dimensional clustering methods are discussed. The main idea for uni-dimensional clustering via Bayesian networks is to use the Bayesian structural clustering algorithm, which is a greedy algorithm that makes use of the EM algorithm. On the other hand, for multi-dimensional clustering we investigate latent tree models which according to our knowledge, are the only model based approach to multi-dimensional clustering. There are generally two approaches for learning latent tree models: Greedy search and feature selection. The former is able to cover a wider range of models, but the latter is more time efficient. However, latent tree models are unable to capture dependency between partitions through attributes. So we propose two approaches to overcome this shortcoming. Our first approach extends the idea of Bayesian structural clustering for uni-dimensional clustering, while the second one is a combination of feature selection methods and the main idea of multi-dimensional classification with Bayesian networks. We test our second approach on both real and synthetic data. The results show the goodness of our approach in finding meaningful and novel partitions.

1 Introduction

Clustering, also known as unsupervised learning, is a task aiming to group objects together based on their similarities. Different notions of similarity will lead to significantly different clustering algorithms. There are generally two popular notions for cluster similarity: distances among objects and cluster distributions. The first approach tries to find homogeneous clusters such that each cluster is as different as possible from others. These methods are called distance based. The second approach tries to model the probability distribution that gave rise to each cluster. This approach is called model based.

O. Keivani (✉) • J.M. Peña
ADIT, IDA, Linköping University, Linköping, Sweden
e-mail: omid.keivani@liu.se; jose.m.pena@liu.se

© Springer International Publishing Switzerland 2016
M.E. Celebi, K. Aydin (eds.), *Unsupervised Learning Algorithms*,
DOI 10.1007/978-3-319-24211-8_7

Each approach has its own merits and drawbacks. For example, distance based methods are simpler and more time efficient. On the other hand, model based approaches are more flexible and will find the structure of the mechanism that produced the data instead of just cluster assignments or representatives for clusters (cluster centers). For further discussion about model based approaches and their advantages, see [1]. In this chapter, we will only consider model based approaches via Bayesian networks (BNs).

Datasets are usually represented by a $n \times m$ matrix, where n is the number of objects in dataset and m is the number of attributes. Therefore, the aim of a model based clustering method would be to find a model, which describes the existing structures within the dataset best. Model based clustering typically consists in modeling the probability distribution of the cluster variable, i.e., $P(C)$, and the probability distribution of the attributes given the cluster variable, i.e., $P(A_1, A_2, \ldots, A_m|C)$. The probability distribution of the attributes can be obtained by combining the two previous distributions into a so-called finite mixture model, i.e.

$$P(A_1, A_2, \ldots, A_m) = \sum_c P(C)P(A_1, A_2, \ldots, A_m|C). \tag{1}$$

In clustering, we are however more interested in the probability distribution of the cluster variable given the attributes, i.e.

$$P(C|A_1, A_2, \ldots, A_m) \propto P(C)P(A_1, A_2, \ldots, A_m|C). \tag{2}$$

However, when the number of attributes is large, it may be cumbersome to work with $P(A_1, \ldots, A_m|C)$. Instead, probabilistic graphical models (PGMs) such as BNs can be used to factorize this distribution, making it possible to work with it efficiently. Specifically,

$$P(C|A_1, A_2, \ldots, A_m) \propto P(C) \prod_i P(A_i|\mathrm{Pa}_G(A_i), C), \tag{3}$$

where $\mathrm{Pa}_G(A_i)$ are the attributes that are parents of A_i in the directed and acyclic graph G (DAG), which is known as the structure of the BN. Figure 1 shows a BN with four variables where, $\mathrm{Pa}(C) = \{A, B\}$, $\mathrm{Pa}(D) = \{C, B\}$, and $\mathrm{Pa}(A) = \mathrm{Pa}(B) = \emptyset$. The second component of the BN is a set of parameters [the conditional probability distributions in the r.h.s of (3)] typically estimated through maximum likelihood (ML) from data. Clearly, knowing the correct structure of the BN is crucial for finding the correct clustering. There are generally two approaches to this task: The structure is provided by an expert beforehand, or the structure is learnt from some data at hand. In this chapter, we will review the latter approach, which includes works such as [2–5].

Most of the model based approaches to clustering assume that there exists only one cluster variable, i.e., uni-dimensional clustering. However, data may be

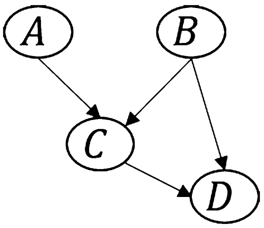

Fig. 1 A simple BN with four random variables

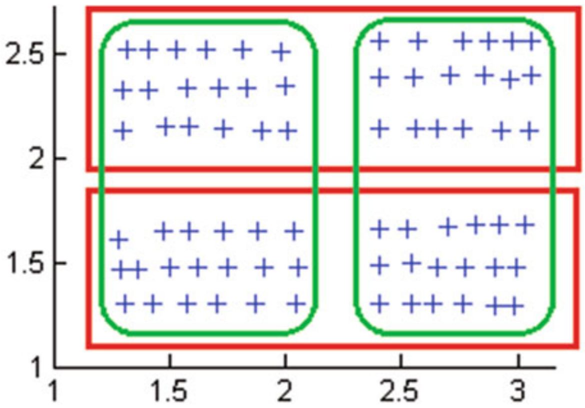

Fig. 2 A toy example for multi-dimensional clustering

multifaceted or, in other words, the objects may be grouped differently according to different criteria. As a toy example, consider Fig. 2. We can group this data into two clusters, both horizontally and vertically. The situation where data is clustered simultaneously according to different criteria is known as multi-dimensional clustering. While there are many works on model based multi-dimensional classification [6–10], there exist just a few works on model based multi-dimensional clustering [11–14]. However, there exist other methods in the BN literature that find and introduce latent variables in the network [15, 16]. However, these approaches have not been applied to clustering domains and we are not aware of their capabilities under clustering assumptions, i.e., cluster variables should not have any parents. Hence, we are not going to discuss them. This chapter will review model based multi-dimensional clustering works and contribute with two new algorithms for this task.

The rest of the chapter is organized as follows. Section 2 discusses model based uni-dimensional approaches, while Sect. 3 will discuss multi-dimensional clustering. Section 4 contains our proposed model based method for multi-dimensional clustering and some preliminary results are shown in Sect. 5. Finally, we will draw some conclusion and summarize the discussed methods in Sect. 6.

2 Uni-Dimensional Clustering

Uni-dimensional clustering or simply clustering means that only one cluster variable is available and we are looking for a model that best describes the data. Large number of methods are available for uni-dimensional clustering [2–5]. In model based clustering with BNs, we normally assume that we have one cluster variable C, which is hidden and is the parent of all attributes, and for each object, we are looking for the value of variable C that maximizes the posterior probability of the cluster variable given that object:

$$c = \text{argmax}_C P(C|A_1, A_2, \ldots, A_m). \tag{4}$$

According to Bayes formula:

$$P(C|A_1, A_2, \ldots, A_m) \propto P(C)P(A_1, A_2, \ldots, A_m|C). \tag{5}$$

However, as we mentioned before, working with $P(A_1, A_2, \ldots, A_m|C)$ may be inconvenient. With the help of a BN such as G, this term would be factorize as follow:

$$P(C|A_1, A_2, \ldots, A_m) \propto P(C) \prod_i P(A_i|\text{Pa}_G(A_i), C). \tag{6}$$

But, since the cluster variable C is hidden, computing the ML parameters for G is not an easy task. So, we have to estimate the ML parameters by iterating two steps: Fractional completion of the database and using ML estimation on the completed data. This process is known as expectation maximization (EM) [17]. However, EM merely learns the parameters. In order to learn both the structure and its parameters simultaneously, Peña et al. [2] proposed a method, which has been used by many others [3–5] afterwards. The main idea is to consider the expected value of C given the values of the attributes in that object as its real value. In this way we will have a complete data at hand. Having a complete data, any parameter estimation algorithm, such as ML or maximum a posteriori (MAP) can be used to estimate parameters in the r.h.s of (6). Figure 3 shows the general algorithm for learning BNs from incomplete data. However, as a special case, where one variable is hidden (cluster variable), we can use it for clustering so we call it Bayesian structural clustering (BSC). It is also known as Bayesian structural EM if we use EM algorithm as our parameter search step [18]. Although there exist other methods for parameter search step such as matrix decomposition methods [19], which is used for learning parameters of a hidden markov model, but we do not know if it can be applied to clustering as well. Furthermore, there exist other variants of EM (we will point them later on), but usually EM algorithm will be used for parameter learning step and different methods merely differ in their structure search step. Some methods do both the structure learning and parameter learning steps. However, since the

```
1. Choose initial structure and initial set of parameter values
2. Parameter search step
3. Compute $P(C|A_1 \dots A_m)$ for all objects to complete the dataset
4. Structure search step
5. Re-estimate parameter values for the new structure
6. If no change in the structure has been done
     then stop
     else go to 2.
```

Fig. 3 General algorithm for clustering via BNs (Bayesian structural clustering)

structure learning step is time consuming, there exist another group of methods which ignore this step and only focus on the parameter learning step. We will discuss both approaches in the following.

2.1 Known Structure

In some problems, the structure of the BN may be pre-defined. This could be the case when an expert has some knowledge about the problem and defines a specific structure for it or we may define a structure according to some assumptions. The former situation requires a strong and reliable expert, while the latter is very common and is widely used. The most well-known case for the latter is naive Bayes (NB) structure.

2.1.1 Naive Bayes

The simplest structure for a BN is to assume that all attributes are independent of each other given the cluster variable. This structure is called NB, also known as latent class model (LCM). NB structure is fixed and does not require any learning procedure. In this structure, the cluster variable is the sole parent of all attributes and no other edges are allowed. Figure 4 shows an NB structure. According to this assumption, we can rewrite (3) as follows:

$$P(C|A_1, A_2, \dots, A_m) \propto P(C) \prod_{i=1}^{m} P(A_i|C). \tag{7}$$

Having the known structure, we only need the parameters ($P(C)$ and $P(A_i|C)$) to be able to compute (7). Since the cluster variable is hidden, we are not able to use MLE or MAP to estimate the parameters. So, learning the parameters would

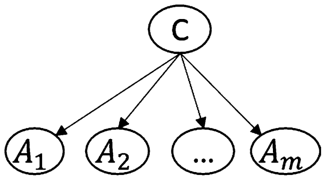

Fig. 4 Naive Bayes structure

```
Input
    $G$: A BN structure
    $\theta^0$: Random parameters
    $D$: Dataset
Output
    $\theta^k$: Estimated parameters
Main
    1. while {stopping criterion}
    2.     Compute the posterior probability of cluster variable given attributes for each object
    3.     Compute MLE/MAP parameters $\theta^{k+1}$(according to now complete data) ($\theta^{k+1}$)
    4. End while
```

Fig. 5 EM method for a known BN structure

not be as straightforward as it is in classification problems (where class nodes are observed). However, we can consider this problem as a parameter learning problem with missing data. Usually, in the literature, the EM algorithm will be used to estimate the parameter from missing data [20]. Figure 5 shows the procedure of EM when the BN structure is known. The algorithm requires an incomplete dataset D and since it assumes that the structure is known, it should also take a BN structure (G) along with its initial (random) parameters (θ^0) as its input. At the second line, for each object in D the algorithm will compute the posterior probability of the cluster variable (7) and complete the dataset. Line 3 computes either MLE or MAP parameters according to the completed dataset. These two steps should be done until a stopping criterion is met. Usually the difference between the log-likelihood of two consecutive steps ($\text{LL}(\theta_{k+1}|D, G) - \text{LL}(\theta_k|D, G)$) will be used as a stopping criterion. Having both structure and parameters, we can compute the posterior probability of cluster variable given each object and since we did not learn the structure, this method would have low time complexity. Although the NB assumption may not be realistic and we expect it to have detrimental effects, this model has proved to be very powerful and produce acceptable results. However, [21, 22] used expectation model averaging (EMA) instead of EM as the parameter search step to incorporate information about conditional (in)dependencies between attributes in parameters and compensate for the lack of conditional (in)dependencies that exist in the NB model.

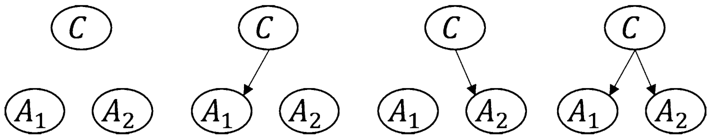

Fig. 6 All possible selective naive Bayes structures with two attributes

2.1.2 Expectation Model Averaging

The main idea of [21, 22] for learning a BN is the same as for learning NB. However, they differ in parameter search step. Santafé et al. [21] propose and use EMA algorithm instead of EM for the parameter search step. Like the EM algorithm, EMA has two steps. First, it will compute expected sufficient statistics and then it will average over the parameters of all possible selective naive Bayes (SNB) structures. SNB is an NB structure, where each attribute may or may not depend on cluster variable. Figure 6 shows all possible SNBs with two attributes. E step calculates the posterior probabilities of the cluster variable given each object to complete the dataset and will compute all expected sufficient statistics. Expected sufficient statistics should be computed for all SNB structures, so three different types of variables exist. Cluster variable (which has no parents) and an attribute with and without parent.

Having all sufficient statistics at hand, we can run model averaging (MA) algorithm to reestimate parameters for NB structure (this step mimics the M step in EM algorithm). In order to do so, MA algorithm computes $P(\theta|D)$ by averaging over the MAP parameters of all possible SNB structures. Computing these values for a single structure S is an easy task. We know that computing MAP parameters requires both prior knowledge ($P(\theta)$) and actual sufficient statistics computed from the complete data ($P(D|\theta, S)$):

$$P(\theta|S, D) \propto P(D|\theta, S) \times P(\theta). \tag{8}$$

We can use the expected sufficient statistics, computed in the E step, as an approximation for real ones. Also, knowing that the Dirichlet distribution is in the class of conjugate priors for multinomial distributions, assuming Dirichlet distribution as our prior knowledge helps us do our computation in closed form [23]. MAP parameters can be easily computed from the expected sufficient statistics and hyper parameters of the Dirichlet distribution [23].

However, the aim of EMA algorithm is to average over parameters of all possible SNBs. Hence, the MA calculations need a summation over all possible SNBs (2^m). Note that, since we need to average over all possible structures, there would be a lot of repetitions in the computations that need to be done only once. Hence, the algorithm would be time efficient. For more details on how to average parameters over all possible structures efficiently, see [21].

In this way, although the structure that we will have is an NB, its parameters will take into account some information about the conditional (in)dependencies between variables represented by all SNBs. In another attempt, to further improve the parameter search step, Santafé et al. [22] used the same idea as EMA, but this time they replaced SNB structures with tree augmented naive Bayes (TAN). TAN structure provides more realistic conditional (in)dependencies, so we would expect to have better results.

2.1.3 Expectation Model Averaging: Tree Augmented Naive Bayes

To improve the quality of the model which will be learnt by EMA, Santafé et al. [22] used a TAN model instead of SNB to capture more conditional (in)dependencies and incorporating them into parameter search step. The classical TAN model [24] is a structure in which the attributes form a tree and the hidden variable (cluster) is the parent of all attributes. Santafé et al. [22] removes these two constraints to widen the range of representable conditional (in)dependencies. First, they allow the attributes to form a forest instead of a tree and second, the cluster variable may or may not be the parent of an attribute. Also, they consider a fixed ordering for their structure. L^{π}_{TAN} refers to such a TAN with a fixed ordering of π. This new TAN is a superset of NB, SBN, and TAN models. For example, for a given ordering of $\pi = \{A_1, A_2, A_3\}$, the possible parent sets for A_2 will be:

$$\text{Pa}_{A_2} = \{\emptyset\}, \{C\}, \{A_1\}, \{A_1, C\}. \tag{9}$$

Santafé et al. [22] considers $\pi = C, A_1, A_2, \ldots, A_m$ for all cases. So, the number of parents for ith attribute will be 2^i. Like EMA algorithm, EMA-TAN has two steps. In the first step, just like EMA, we have to compute the expected sufficient statistics. The second step is to run MA algorithm according to the completed dataset. The MA step of EMA-TAN is the same as the MA step of EMA algorithm, which we discussed before. The only difference is that, now the structure S is a TAN instead of SNB. Once again, MAP parameters can be easily computed from the expected sufficient statistics and hyper parameters of the Dirichlet distribution for a single structure. So, we can average over all MAP parameters of all members of L^{π}_{TAN}. The averaging process is just like the EMA algorithm [22].

Since L^{π}_{TAN} is a superset of SNB, EMA-TAN will incorporate more information regarding the conditional (in)dependencies between variables in its parameter learning step in comparison to EMA. However, both EMA and EMA-TAN suffer from the NB assumption in their structure. NB structure is a very basic and simple minded model which is not true in many real-world problems. This is why many researchers tried to extend this model by removing the conditional independence assumptions. We will focus on them next.

2.2 Unknown Structure

There are many cases in which only raw data is available and no information regarding the structure of the BN is at hand. In these cases, the structure should be learnt from data itself. Several methods for learning BNs from complete data exist [20, 23]. However, in a clustering problem, since the cluster variable is hidden, we have to learn the structure from incomplete data. In order to do so, we have to use BSC (Fig. 3). There are two main steps in BSC: Parameter search and structure learning steps. As we saw earlier, some methods assume that the structure is known and only focus on the parameter search step. In this section we are going to mention those which learn the structure and try to remove the NB assumption. However, these extensions will happen at the cost of time complexity. The key point is to propose an algorithm that balances well between time complexity of the algorithm and quality of structure. Peña et al. [2] tried to extend NB in a way that both improved its model quality and keep its simplicity and they called it constructive induction learning.

2.2.1 Extended Naive Bayes

Peña et al. [2] tried to introduce a model, which maintains the simplicity of NB and still improves its model quality. This model is the same as NB with only one difference. The model allows some attributes to group together under the same node as fully correlated attributes (supernodes), so the number of nodes in the structure can be smaller than the original number of attributes. Hence, the posterior probability of a class given all attributes (7) can be rewritten as:

$$P(C|A_1, A_2, \ldots, A_m) \propto P(C) \prod_{i=1}^{e} P(A_i|C), \tag{10}$$

where e is the number of variables in the structure (supernodes and common nodes) and $e \leq m$. In this way a better performance than NB will be achieved at low cost. In order to find supernodes, we have to choose a score first. A good and efficient score is one which is both factorable and has a closed form computation. Both, log-likelihood and log marginal likelihood scores have these properties [2, used the latter]. Under standard assumptions, log marginal likelihood is:

$$\mathrm{LML}(D|G) = \log p(D|G) \propto \sum_{i=1}^{m} \log \prod_{j=1}^{q_i} \frac{(r_i - 1)!}{(N_{ij} + r_i - 1)!} \prod_{k=1}^{r_i} N_{ijk}!, \tag{11}$$

where m is the number of attributes, r_i the number of states that the ith attribute has, q_i the number of states that the parent set of the ith attribute has, N_{ijk} the number of cases in the dataset where A_i has its kth value and its parent set takes its jth value and $N_{ij} = \sum_{k=1}^{r_i} N_{ijk}$.

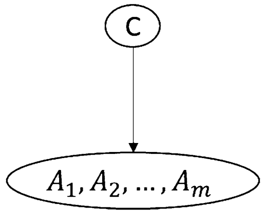

Fig. 7 Initial structure for backward structure search

For the search algorithm, Peña et al. [2] used constructive induction by using forward and backward search. In a forward structure search (FSS), we have to consider NB as the initial point. Then all pairs will be joined (creating a new variable which is the Cartesian product of two variables) and the one which increases the score most, will be chosen as supernode. This process will be iterated until no join action can improve the score of the current structure. On the other hand, backward structure search (BSS) use a fully correlated model as its starting point (Fig. 7). We have to consider all possible splitting actions (breaking one variable into two separate and conditionally independence variables). Then we will choose the action which leads to the highest increase in score. This process should be iterated until no splitting action can improve the score of the current structure. Using either FSS or BSS as the structure search and EM as the parameter search step of the general algorithm for clustering via BNs (Fig. 3) we are able to learn a locally optimal BN and its parameters and use it for clustering.

Peña et al. [3] enhances parameter search step and called it Bayesian structural BC+EM. The main idea of BC+EM algorithm is to alternate between bound and collapse (BC) [25] method and the EM algorithm. BC method bounds the set of possible estimates and then collapse them into a unique value according to a convex combination. The estimation of BC algorithm will be used as an initial point for EM algorithm. This way we will achieve a faster convergence rate and more robust algorithm in comparison to traditional EM. In their second effort to further improve the structure search step, Peña et al. [4] proposed recursive Bayesian multinets (RBMNs).

2.2.2 Recursive Bayesian Multinets

Peña et al. [4] in their effort to improve the structure for clustering, proposed RBMNs. Bayesian multinets (BMNs) consist of a distinguished variable Z and a set of component BNs for $\{A_1, \ldots, A_m, C\} \setminus Z$. Generally, the distinguished variable can be either the cluster variable or one of the attributes [26] but, in RBMNs, the distinguished variable is not allowed to be the cluster variable. Figure 8 shows an example of a BMN. As we can see, since BMNs only have one distinguished variable it will always be a depth 1 decision tree (consider the distinguished variable as root and the component BNs as leaves). RBMNs are extensions of BMNs, which can have more than one distinguished variable. Lets say a RBMN is a BMN, which

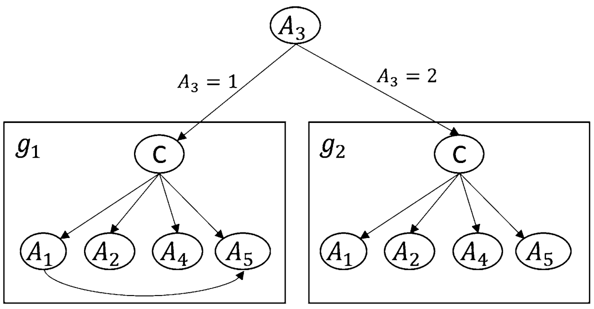

Fig. 8 An example of a BMN for data clustering with five attributes $(A_1, A_2, \ldots, A_5)$, one cluster variable (C) and two component BNs (g_1 and g_2). $Z = A_3$ is the distinguished variable

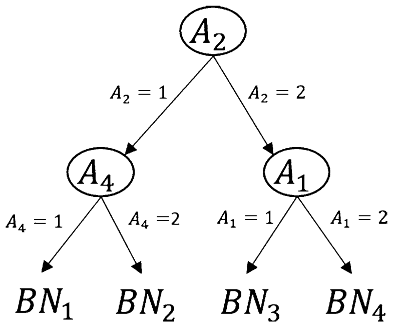

Fig. 9 An example of a RBMN for data clustering with five attributes $(A_1, A_2, \ldots, A_5)$ and one cluster variable (C). $Z = \{A_1, A_2, A_4\}$ is the set of distinguished variables

each of its components (leaves) can either be a BMN or just a BN. This way we can have a decision tree with a depth more than one. Peña et al. [4] referred to RBMNs as *distinguished decision tree* and defined it as follows:

- The cluster variable cannot be a distinguished variable.
- Every internal node is a distinguished variable.
- Every internal node has as many children or branches coming out from it as states for the variable represented by the node.
- No path to the leaf should contain a repeated variable.

So, RBMNs are more general than both BMNs and BNs. A BMN is a RBMN with only one internal node, while a BN is a RBMN with no internal nodes. Figure 9 is an example of RBMN structure. According to the fourth condition of RBMNs definition, in Fig. 9, BN_1 and BN_2 should only contain $\{A_1, A_3, A_5, C\}$ variables and BN_3 and BN_4 should include $\{A_3, A_4, A_5, C\}$ and no more.

The most compelling advantage of BMNs, or generally RBMNs, is that they can encode context-specific (in)dependencies [27], while the conditional (in)dependencies that are encoded by a BN are context-non-specific. A conditional (in)dependency is context specific if two sets of variables are conditionally independent of each other given a third set with a specific configuration and dependent given the same third set but with another configuration for its variables. This means that RBMNs are more flexible than simple BNs and they can support a wider range of conditional (in)dependencies. Also, RBMNs are very intuitive. Many examples can be found that have different models according to different values of one or more variables (distinguished variables). For example, the body types of male and female are different so they should have different models so, in this case, the Sex variable would be the distinguished variable. Age, Birth place, Height, Weight, and etc., are few examples of many possible distinguished variables in an example. So, now that the preliminaries of RBMNs have been introduced, it is time to provide a learning algorithm for the clustering purpose. We can use the general algorithm for clustering via BNs (Fig. 3). As for the parameter search step, both EM and BC + EM are eligible. Peña et al. [4] used an ENB for component BNs, so the process of learning component BNs will be the same as Sect. 2.2.1. Recall the fact that, in order to be able to search efficiently for an optimal BN among the vast number of possible BNs, the chosen score has to be both decomposable and in closed form. For learning ENB, log marginal likelihood (11) has been used, Thiesson et al. [26] extends this score for BMNs and [4] extends it further for RBMNs. Under standard assumptions, the log marginal likelihood for BMNs will be:

$$\log p(D|G_{\text{BMN}}) = \log p(D^Z) + \sum_{g=1}^{|Z|} \log p(D^{X,z}|bn_g), \tag{12}$$

where D^Z is the data restricted to distinguished variable Z, $|Z|$ is the number of component BNs (number of values of the distinguished variable), bn_g is the gth component BN, X is the set of all attributes and the cluster variable $(\{A_1, A_2, \ldots, A_m, C\})$ and $D^{X,z}$ is the data limited to $X \setminus \{Z\}$ and those objects in which $Z = z$. The first term on the r.h.s of (12) is the log marginal likelihood of a BN with only one variable (distinguished variable) and the second term is the sum of log marginal likelihood for all component BNs separately.

Moreover, under the same assumption as for BMNs, log marginal likelihood for RBMNs is:

$$\log p(D|G_{\text{RBMN}}) = \sum_{l=1}^{L} [\log p(D^{t(\text{root},l)}) + \log p(D^{X,t(\text{root},l)}|bn_l)], \tag{13}$$

where L is the number of leaves, $t(\text{root}, l)$ are the variables which are in the path between root and the lth leaf, $D^{t(\text{root},l)}$ is the data restricted to those variables in $t(\text{root}, l)$, bn_l is the lth component BN, and $D^{X,t(\text{root},l)}$ is the data restricted to variables which are not in $t(\text{root}, l)$ and those cases which are consistent with $t(\text{root}, l)$ set values. The first term in the summation of (13) is the log marginal likelihood of a trivial BN with a single node with as many states as leaves in the distinguished decision tree, which is easy to compute (11). Also, the second term is

Table 1 Different choices for clustering via BNs

Algorithm	Parameter search	Score	Structure	Structure search
Peña et al. [2]	BC + EM	LML	ENB	FSS/BSS
Peña et al. [3]	BC + EM	LML	Any BN	HC
Peña et al. [4]	BC + EM	LML	RBMN	FSS/BSS
Pham and Ruz [5]	EM	MI	CL multinet	MWST
Pham and Ruz [5]	EM	CMI	TAN	MWST
Pham and Ruz [5]	EM	CMI	SBN	MWST
Santafe et al. 2006 [21]	EMA	LL	NB	–
Santafe et al. 2006 [22]	EMA-TAN	LL	NB	–

the log marginal likelihood for each leaf (component BN). So, under the mentioned assumptions, both BMNs (12) and RBMNs (13) scores are factorable and have closed form calculation.

Now that we have a factorable and closed form score for RBMNs we can proceed with the learning algorithm for clustering purpose. Peña et al. [4] has used constructive induction as their learning algorithm. It starts with an empty distinguished structure and at each iteration, it will increase the depth of the structure by one. In each iteration, each leaf should be replaced with the best BMN that has the highest log marginal likelihood (12) for variables in $X \setminus t(\text{root}, l)$. This task should be iterated until either the depth of the structure reaches a specific number or there exist no more BMNs which can be replace a component BN, such that it increases the log marginal likelihood (13).

Generally, to do clustering with BNs, one can use BSC (Fig. 3) and use different methods for its parameter learning (EM, BC+EM, EMA, or EMA-TAN) and structure learning (HC, FSS, BSS, or MWST) steps. The vital point is that, no matter which structure learning algorithm we choose, we have to select a scoring function that is both factorable and in closed form. For example, [5] select log-likelihood as the scoring function, EM as the parameter search, TAN and maximum-weighted spanning tree (MWST) for structure learning step to propose a new clustering algorithm via BNs. Table 1 shows different choices that lead to different methods. Each of these methods has their own pros and cons. For example, BC + EM is more robust and has a faster convergence rate in comparison to EM or the tree structure which will be learnt by Pham and Ruz [5] according to (conditional) mutual information (CMI) is a global optimal tree while other structures in other methods are locally optimal. Also, the methods by Santafe et al. have the advantage of incorporating some information about conditional independencies into parameters (which obviates the need to do structure search step) and gives it the upper hand to do the calculations faster. The one thing that all of these methods have in common is the fact that they want to balance between time complexity and model quality.

3 Multi-Dimensional Clustering

As we said before, there are cases where data is multifaceted. This means that different experts may cluster data differently (Fig. 2). This disparity may emanate from choosing different subsets of attributes. However, two experts may consider all attributes and still do the clustering task differently. In Fig. 2, considering X or Y dimension, will result in different clustering outputs. As another example, consider that we want to partition customers of a supermarket. Considering "pet food" attribute, we can partition them into those who have pets and those who have not. However, if we consider another attribute, for example "cigar," then we may partition differently (smokers and non-smokers). Generally, there exist two types of methods for this purpose: Multi uni-dimensional and multi-dimensional. We refer to those that assume that all partitions are independent of each other as multi uni-dimensional and call the ones which consider potential relationship between partitions (this may result in either a connected or disconnected model), multi-dimensional. Galimberti and Soffritti [28] and Guan et al. [29] are examples of multi uni-dimensional clustering. One can just select an attribute subset and assign a latent variable to it to achieve a clustering model. And since it is assumed that all partitions are totally independent of each other, this problem could be reduced to a simple feature selection problem. Herman et al. [30] introduced and discussed many measures for this purpose. In this chapter we won't discuss multi uni-dimensional clustering. On the other hand, for multi-dimensional clustering, Zhang [14] introduced a model called latent tree models (LTMs) in 2004. In the next section we will focus on this model and its extensions.

3.1 *Latent Tree Models*

LTMs have two types of variables: Observed variables which are the attributes and latent variables. Each latent variable stands for a partition and can only be an internal node while each attribute should be a leaf. Figure 10 shows a possible LTM structure, which has three partitions which are correlated ($C_2 \leftarrow C_1 \rightarrow C_3$) and three conditionally independent attribute subsets $\{A_1, A_2, A_3\}$,$\{A_4\}$,$\{A_5, A_6\}$. From now on, we use the words "latent variable," "partition," and "cluster variable" interchangeably. Table 2 shows the general steps of LTM to perform multi-dimensional clustering. In the first step, we can either assume a pre-defined number for partitions or figure it out in the learning process. Like in the first step, one can set a pre-defined number for clusters in each partition for the second step. As for the third step, any structure learning algorithm can be use to learn the required structures. Also, one can use EM as its parameter estimation step. Since we can have a single partition, uni-dimensional clustering is a special case of LTMs. In the next section we will discuss both cases where the structure is known a priori or it is unknown.

Fig. 10 LTM structure with six observed and three latent variables

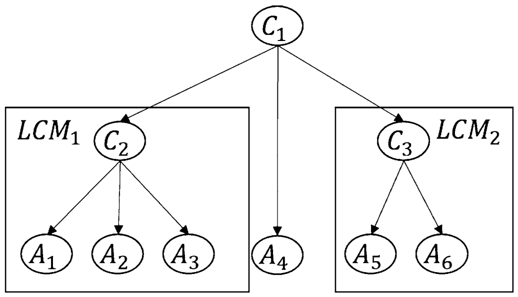

Table 2 Latent tree model steps for multi-dimensional clustering

1. Determine the number of cluster variables (partitions)
2. Determine the number of clusters for each partition (values for each cluster variable)
3. Learning the LTM structure
(a) Finding the relation between attributes and cluster variables (bridge model)
(b) Finding the relations only between attributes
(c) Finding the relations between cluster variables
4. Estimate model parameters

3.1.1 Known Structure

Knowing LTM structure obviates the need for steps 1–3 in Table 2. This means that we know the number of partitions and their cardinalities along with the relations between all variables (both latent and observed) so we only need to learn the parameters. As we mentioned before, when hidden variables (missing data) exist the first choice to learn parameters is the EM algorithm. However, due to its local optimality and its high time complexity, many researchers tried to improve it or propose a new algorithm. For LTMs, Mourad et al. [13] introduced LCM-based EM.

The main idea is to first detect all NB structures (they called it LCM) in a LTM and learn their parameters locally by running EM and then predict the latent variables values according to the estimated parameters. These two steps will iterate until we have an estimation for all parameters of all LCMs. Ultimately, a parameter refinement will be done by running EM algorithm globally. Due to using local EMs, this method will reduce computation time. But, especially in multi-dimensional clustering, knowing the structure is hardly the case. So we will focus on the case where the structure is unknown.

3.1.2 Unknown Structure

Lack of prior knowledge about the structure, leaves us with no choice but to learn from data all the components of a LTM. These are number of partitions, their cardinalities, relations between all variables and parameters. One possible option to do so is to use BSC algorithm (Fig. 3). Zhang [14] used such an algorithm by making some assumptions. They assumed that attributes cannot have any relations with each other and the initial structure is an NB structure. Also, two greedy search algorithms have been used for structure search step. One to determine the structure and number of partitions and the other for deciding the cardinalities of latent variables. The first greedy search has three operators: Node introduction, node elimination, and neighbor relocation. And the second one only has one operator which increases cardinality by 1. Due to running two hill climbing algorithms during its process, this method is also called double hill climbing (DHC). It is obvious that such an algorithm is very inefficient because of its double use of hill climbing search. In order to do the search once and reduce the search space, Chen et al. [11] introduced EAST algorithm ([13] called it advanced greedy search (AGS)). This algorithm has five operators, which are divided into three groups. The first group contains node introduction and cardinality increase operators. These two operators will expand the current structure so we call it expansion group. Second group only has the node relocation operator. This operator decides if any edge changes are needed for the current structure. And the last group includes node elimination and cardinality decrease operators. Since these two operators lead to simpler models, this group is called thin group [11]. These three groups will be applied to the current structure step- by-step and after each step, the current model may or may not change. At the end of these three steps, if the structure does not changed, then the algorithm will be terminated.

BIC is a common scoring criterion to decide between current structure and its neighbors. However, Zhang and Kocka [31] mentioned that, initiating the search with a naive structure and using BIC, we will always select cardinality increase over the node introduction operator in expansion step. So, they introduced a so-called improvement ratio criterion and use it instead of BIC in this step:

$$\mathrm{IR}^{\mathrm{BIC}}(T', T|D) = \frac{\mathrm{BIC}(T', D) - \mathrm{BIC}(T, D)}{\dim(T') - \dim(T)}. \tag{14}$$

The numerator of (14) is the difference of BIC scores between the candidate structure (T') and current structure (T) while denominator is their dimension difference.

Pouch latent tree model (PLTM) is another example of learning LTM structure based on score-search methods [32, 33]. It assumes that all attributes are continuous and they can be merged into one node called pouch node (pouch node is the same as supernodes in RBMNs, except that in a pouch, all attributes are continuous). Also, it introduces two new search operators (in addition to the five operators of EAST algorithm) for its hill climbing (pouching (PO) and unpouching (UP) for merging and splitting attributes, respectively). Yet, score-search based methods still suffer from tedious searches and the problem will be doubled in presence of latent variables (need for running EM before the first step).

So, there is another class of methods which uses a feature selection algorithm to group attributes and then try to construct the structure with inductive learning.

Liu et al. [12] proposed a new method called bridged islands (BI) and then [34] extend it. In the next section we will try to extend this algorithm by overcoming some of its deficiencies, so we will have a close scrutiny of this method.

BI do the clustering task by introducing a new concept called sibling clusters. Sibling clusters are sets of attributes which are grouped under the same latent variable. BI learns a LTM in four steps:

1. Form sibling clusters
2. Assign a latent variable to each sibling cluster and decide its cardinality
3. Learn a Chow-Liu tree for latent variables
4. Refine the model

To form sibling clusters we have an active subset of attributes. At first, all attributes are in the active set. Initially, the pair with the highest mutual information (MI) will be chosen and then each time an attribute which has the highest MI with the previously selected attributes will be added to them. The MI between a single attribute (A) and a set of attributes (W) is estimated as follows:

$$I(A;W) = \max_{Z \in W} I(A;Z). \tag{15}$$

With each attribute addition we will have a new naive model containing the selected attributes and one latent variable (g_1) and use EAST algorithm to find the best possible structure with exactly two latent variables (g_2) (for the same subset of attributes). The process of adding attributes continues until uni-dimensionality (UD) test fails. UD test runs on two LTMs (g_1 and g_2) to see which structure has larger score. If the model with two latent variables (g_2) has the larger score, then we say that UD test has failed and we stop adding more attributes. Two possible structures which can be formed according to the mentioned process is shown in Fig. 11. At this point we have to choose one of the attribute sets in g_2 ($\{A_1,A_2,A_3\}$ or $\{A_4,A_5\}$) as sibling cluster. We choose the one that contains the attributes of the pair with the highest MI (if the attributes of such a pair were on different sets, then we choose randomly). Finally, the attributes in the cluster sibling set have to be removed from the active set and the process restarts with the new active set until |Active Set| ≤ 2. Now, we have all sibling clusters and thus we know the number of partitions (we assign a latent variable to each sibling cluster), so we have to decide the cardinality

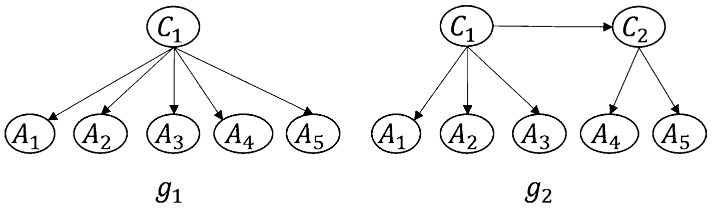

Fig. 11 Two possible structures for the UD test during forming the sibling clusters

of each latent variable. This step can be done easily using a greedy search. Third step is to find the relations between latent variables such that they form a tree. This step can be easily done by using the well-known Chow-Liu algorithm [24]. The last step tries to correct the probable mistakes in previous steps. It will investigate if any node relocation action or changing any latent variable cardinality will lead to a greater score, if it does that action should be taken and the model will change for one last time. Ultimately, EM algorithm runs globally to estimate the parameters. Note that, at the end of each step (after the model changes), EM algorithm will be run locally to refine parameters. However, Mourad et al. [13] discussed that when the number of attributes is large (e.g., 1000), it is better to use a forest structure instead of a tree. They justify it with the fact that, when the number of attributes is large, there exist many cluster variables which are not correlated to each other. So, they suggest to first learn a tree structure and then use any independence test to remove edges between cluster variables. It is obvious that BI is much faster than EAST algorithm [12, 34].

Another method which uses feature selection as its first step is binary trees (BT) [13]. Like BI, at the first step BT finds a pair with the highest MI and set a latent variable as their parents. Then it will remove the selected attributes from the observed variables set and complete the data for the introduced latent variable, next it will add (now observed) latent variable to the observed variables set. These steps will be repeated until only two variables left in the observed variables set. This algorithm is named as LCM-based LTM learning, because at each step a LCM structure is learnt.

In all of the mentioned methods (DHC, EAST, BI, and BT) there exists a step for finding the cluster variable cardinality. The simple way is to check the score of the structure for all possible cardinalities of all cluster variables. However, when the number of latent variables is large this approach would be intractable. So, generally a greedy search will be done to find a local optimum. But, this method still remains inefficient for large number of latent variables. There exist several approaches which tackle this issue [16, 35, 36], but we won't discuss them here.

With all this being said, there is a vital point in multi-dimensional clustering which we didn't heed to. In any multi-dimensional learning algorithm, each partition should express different concept from other partitions. This means that the learning algorithm should ensure the novelty of each partition. In the next section we will describe how LTM methods guarantee this property and how we can interpret the meaning of each partition.

3.2 Cluster Variables Novelty

This section describes how LTMs guarantee the novelty of their discovered partitions. If different partitions express different concepts, then we say that they are novel. So, let us begin by showing how we can infer the concept of a partition in LTMs. Information curve is widely used for this purpose [11–13]. Information curve is a measure which detects the most influential attributes for a latent variable in LTM structure. Both pairwise mutual information (PMI) and cumulative mutual information (CMI) are part of an information curve. PMI is the MI between a single latent variable (C) and an attribute (A_i) (MI(A_i, C)), while CMI is the MI between a single latent variable (C) and a set of attributes $\{A_1, A_2, \ldots, A_i\}$, where $i = 1, 2, \ldots, m$ (MI($C, \{A_1, \ldots, A_i\}$)). The following value is based on CMI and it is called the information coverage:

$$\mathrm{IC} = \frac{\mathrm{MI}(C, A_1, \ldots, A_i)}{\mathrm{MI}(C, A_1, \ldots, A_m)}. \tag{16}$$

When IC equals to 1, we can say that the first i attributes describe C perfectly and C is independent of all other attributes given the first i attributes.

So, to choose a set of attributes which best describe a latent variable C (influential set) we can select an attribute which has the highest PMI with C and then, at each step, add the attribute which increases the IC value most. A threshold for IC can be used as a stopping criterion. For example, we can stop adding attributes whenever IC reaches 90 %. One can use conditional entropy instead of IC in the same manner. At each step the attribute which decreases the conditional entropy of the latent variable C most, has to be added to the influential set. Whenever the conditional entropy of C given its influential set reaches zero (close to zero), we can stop adding attributes. Moreover, Herman et al. [30] discussed and introduced alternative MI-based methods for selecting informative attribute sets, which can be used to determine influential sets.

Having a set of influential attributes for each partition, we can infer the concept of each partition in the LTM structure. For instance, let “sex” and “age” be the only two attributes in the influential set of partition C, then we can infer that C is primarily based on these attributes and C thus gives us information regarding the age and sex of a person. Thus, in a LTM structure, if the influential sets for two partitions C_1 and C_2 are exactly the same (or nearly the same) we can claim that one of them is not novel.

LTMs guarantee the novelty of its partitions by not allowing them to share any attributes. This fact will ensure that the intersection of any two influential sets will be a null set. So, the novelty condition is always satisfied. However, not allowing partitions to share attributes result in two severe shortcomings. The first issue is straightforward, if an attribute belongs to a partition but does not belong to its influential set, then it is not allowed to be selected in any other influential set. Thus, many attributes will be useless, while this is not true in reality. The second problem is more subtle and we explain it with an example. Consider a case with two attributes “wet grass” and “slippery road” and two partitions “rain” and “sprinkler.” The most intuitive model would be the one in Fig. 12. We can see that “wet grass” is shared between two partitions and removing it from “rain” will increase the uncertainty of this partition and removing it from “sprinkle” will leave this partition with no attribute, which force us to remove the partition itself. In either

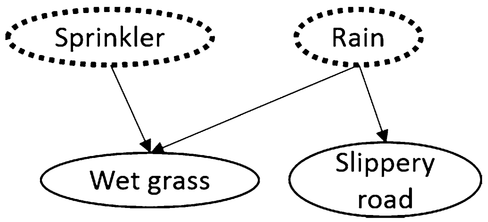

Fig. 12 A multi-dimensional clustering with two partitions and two attributes

case, not sharing "wet grass" between two partitions will be harmful. This motivates us to propose a new method for multi-dimensional clustering by overcoming the mentioned shortcomings of LTMs.

4 Our Approach

In this section we aim to propose a new method to learn a BN for multi-dimensional clustering by overcoming the aforementioned disadvantages of the previous methods. The most compelling drawbacks of LTMs were their conditional independence assumption between attributes and limiting each attribute to belong to only one partition. However, it is obvious that there exist many models which violate these assumptions. Also, there exist lots of multi-dimensional classification algorithms without such restrictions [8, 37–39], which have been applied to different problems successfully [40–43]. We try to borrow some ideas from these algorithms and apply them to the clustering field. The most intuitive model for multi-dimensional classification via BNs is called multi-dimensional Bayesian network classification (MBC). In such a model, there exist three structures for two subset of variables (class variables and attributes) (Fig. 13). The main algorithm for learning MBC is the same as BI for clustering without some of its restrictions. Unlike LTMs, there exist edges between attributes ($A_5 \rightarrow A_3$) and also two or more classes may share the same attribute ($C_2 \rightarrow A_3 \leftarrow C_3$). Van Der Gaag and De Waal [6] and de Waal and van der Gaag [44] used tree and polytree structures respectively for both class and attributes and learn the bridge structure with a greedy algorithm with accuracy as its stopping criterion. Bielza et al. [7] used a fixed bridge structure and a greedy algorithm for learning class and attribute structures. However, in clustering, the problem is that no knowledge about the value or number of class variables exist. In other words not only all classes are latent, but also we don't know their numbers.

A straightforward method would be to start from a naive model with one partition and then use BSC algorithm with seven operators (adding an edge, removing an edge, reversing an edge, partition introduction (PI), partition removal (PR), cardinality increment (CI) and cardinality decrement (CD)) for its structure learning step, to improve the structure. Generally, the only restriction for the structure is that there cannot exist an edge from an attribute toward a partition. The first three operators are standard operators, thus there is no need to discuss them. PI is the act

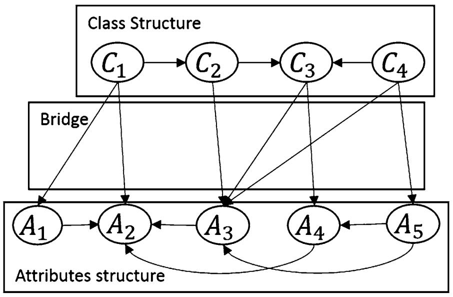

Fig. 13 A possible MBC model with four classes and five attributes

```
Input
    $D$: Data matrix with $n$ objects and $m$ attributes
    $G_{current}$: Initial BN structure
    $\theta_{current}$: Initial random parameters for $G_{current}$
Output
    $G_{current}$: BN structure in the $k^{th}$ iteration
    $\theta_{current}$: Parameters of $G_{current}$
Main
    1. Run EM to estimate the parameters
    2. Compute the posterior probability of cluster variable given attributes (for each object) and complete the dataset
    3. $BIC_{old} \leftarrow$ Compute the BIC score of the current model ($G_{current}$)
    4. Find all neighbors of $G_{current}$ with the following operators: add, remove, reverse, PI, PR, CI, CD
    5. Compute BIC score for all neighbors and select the one with the highest BIC score ($G_{new}$, $BIC_{new}$)
    6. If $BIC_{new} > BIC_{old} + \varepsilon$
           $G_{current} \leftarrow G_{new}$
           Compute ML parameters for $G_{current}$
           Go back to 1.
    7. Else
           Return $G_{current}$
```

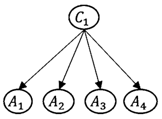

Fig. 14 The general algorithm for learning multi-dimensional clustering

of adding a binary latent variable to the model and consider it as the parent of all attributes and PR will remove a latent variable. The last two operators are for adjusting the cardinality of latent variables. Figure 14 shows the general schema of such an algorithm. The algorithm starts with an initial structure and its random parameters. At the first step, we run EM to estimate the parameters and complete the dataset according to posterior probabilities. Third step will compute the BIC score of the current DAG according to the completed data. Next, we have to find all neighbors of the current structure (with the help of mentioned operators) and compute the BIC score for each one of them. If the current model has the highest score in comparison to all its neighbors then we terminate the algorithm and return the current model as the best one, otherwise we will reestimate the best neighbor parameters and replace the current best model with it and run the same procedure again. Note that using BIC score will ensure the novelty of the partitions. If a partition is not novel then the log-likelihood of the new structure (resulting from using PI operator) will not increase and only the penalty term (structure complexity) will grow. This will lead the new structure to have lower BIC score in comparison to current one.

However, searching in such a large space, would be practically intractable for large number of attributes. In order to reduce the search space, we can either group operators and apply them sequentially (like EAST algorithm) or group random variables and making different sections in the original structure (like MBC) and learn one section at a time. Furthermore, like BSC for uni-dimensional case, any parameter learning and structure learning algorithm can be used for parameter and structure search steps, respectively.

Also, we can follow the idea of BI [12] and form cluster siblings to avoid the time-consuming greedy search required by our first proposed method (Fig. 14). Forming sibling clusters was the first step of BI. This step is done via an approximation for MI (15) and the stopping criterion was the result of a UD test. However, (15) will not consider the MI between the attributes and the cluster variable, which results in detecting attributes of a sibling cluster as redundant in the interpretation of each partition. This is due to the fact that BI or generally all LTM methods, merely use MI between attributes (15) to learn the structure and then measure the MI between the attributes and the cluster variable to interpret the meaning of each cluster. So, we propose a method that consider the dependence between attributes and partitions in the learning procedure. Figure 15 shows the steps of our approach.

Just like MBC, we have three different steps for learning. The first step tries to learn the bridge model, while the second one will learn partition structure and the third step aims to learn the attributes structure. Again, one can limit the attribute and cluster structures to an empty or tree structure. Learning the bridge model resembles the BI algorithm. At first, all attributes belong to the active set ({Active}). The pair with the highest MI in the active set will be chosen. If the pair includes two attributes

```
1. While there is an attribute left in the active set
    1. Find the pair with the highest MI in the active set (at least one of the variables should be an attribute)
    2. If the pair consist of two attributes ($A_i$, $A_j$)
          Set a latent variable ($C_k$) as their parents
          Remove them from the active set
          Add $C_k$ to the active set
          $k = k + 1$
          Run EM locally to determine the values of $C_k$ for each object
```

$$\begin{tikzcd} & C_k \arrow[dl] \arrow[dr] & \\ A_i & & A_j \end{tikzcd}$$

```
    3. If the pair consist of one attribute and one latent variable ($A_i$, $C_j$)
          Add $A_i$ to the child set of $C_j$
          Remove $A_i$ from the active set
          Run EM locally to update the values of $C_j$ for each object
2. Use a search algorithm to find the best structure for partitions
3. Use a search algorithm to find the best structure for attributes
```

Fig. 15 Our second proposed method for multi-dimensional clustering via BN

(A_i and A_j), then we have to introduce a new latent variable (C_k) and set it as the parent of both A_i and A_j. We have to estimate the values of C_k for each object via EM algorithm. Also we have to remove A_i and A_j and add C_k to the active set (step 1.2). But, if the chosen pair contains one latent variable and one attribute (A_i and C_j) then we only need to add A_i to the child set of C_j and then update the values of C_j by running EM algorithm. Also we have to remove A_i from the active set. Moreover, if the pair contains two latent variables (C_j and C_k), then we will just ignore it and find the next pair with the highest MI. We do these steps, until there are no attributes left in the active set. At this stage, we shall have our bridge model with k partitions, which would be an NB model. Then, at the second and third steps the partitions and attributes structures have to be learnt, respectively. First, we will run a hill-climbing algorithm only on partition variables to find any potential edge between them. Next, we run another hill-climbing algorithm, but this time for attributes, to find the relationships between them. Note that, for partition variables, which are hidden, we first predict their values according to their posterior probabilities and then try to learn a structure for them.

Our method relaxes the constraint on sharing attributes in LTMs by allowing indirect relations between attributes and partitions. Figure 16 shows a possible outcome of the proposed method. Since, both C_1 and C_2 are dependent on A_3 given A_2, we can say that they share A_3. Meanwhile, since the children of each partition are not allowed to have any intersection, the partitions are guaranteed to be novel. The advantages of our proposed method over LTMs can be summarized as follows:

- Incorporating the cluster variable in the computations, so there would be no redundant attributes for any partition.
- Allowing attributes to have relations with each other.
- Allowing partitions to share attributes indirectly.
- The model is allowed to be disconnected (if we choose to learn a forest structure for both partitions and attributes variables).

We scrutinize the benefit of the third advantage with an example. Back to our previous example (Fig. 12), assume that we have another variable "sprinkler direction" which can have two values "towards wall" and "towards grass." So, we can say that "sprinkler" and "wet grass" variables are dependent on each other conditioned on the direction of the sprinkler. Figure 17 shows such a model.

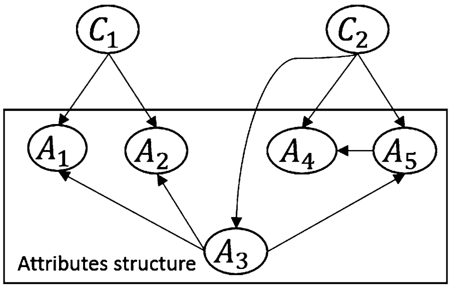

Fig. 16 A possible structure for our proposed method with two partitions and five attributes. The attributes' structure is a Chow-Liu tree

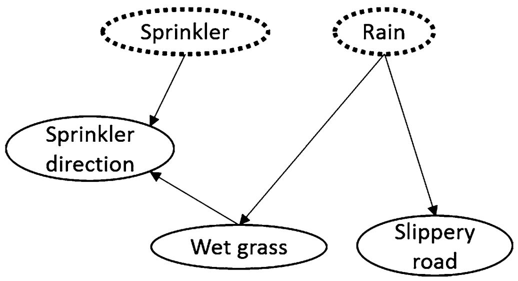

Fig. 17 An example of a possible model with indirect dependency between a cluster variable (Sprinkler) and an attribute (Wet grass)

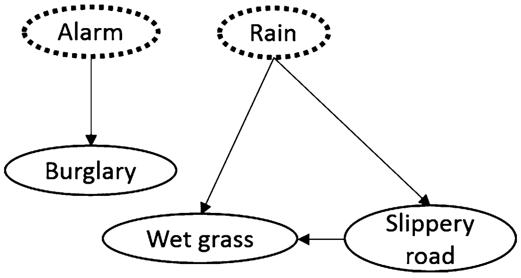

Fig. 18 An example of a model with completely independent partitions (Alarm and Rain)

We can see that "wet grass" will affect both partitions. However, in a LTM model "wet grass" could only affect one of the partitions. Furthermore, the second and fourth advantages will allow the algorithm to cover a wider range of models and be more intuitive. Since the merit of the second advantage is obvious, we only justify the usefulness of the fourth advantage via an example. Consider a problem with three attributes: "slippery road," "wet grass," and "burglary." Maybe the most probable model would be the one in Fig. 18 where two partitions are completely independent of each other (there is no path between them). Again, no LTM algorithm will be able to capture such independencies (alarm and rain), while the proposed method is capable to do so. The reason that LTMs cannot represent such an independent model (Fig. 18) is that, their structure is restricted to a tree, which is always connected. However, one may discuss that removing such a restriction would be easy. Recall that learning LTM structure was possible with two general approaches: search methods and feature selection. Due to their operators (node introduction, node elimination, and neighbor relocation) the former approach will always find a connected model. So, Mourad et al. [13] mentioned latent forest model (LFM) and discussed that one can learn an LTM first and then use some independency test to remove edges between partitions. On the other hand, BI can simply replace its third step (learning Chow-Liu tree) with learning a forest structure. Although these extensions are possible, according to our knowledge, nobody has used them in the LTM literature. In the next section, we are going to show some preliminary results for our second approach and discuss its ability to find meaningful partitions.

5 Preliminary Results

Since a full evaluation of the proposed methods is beyond the goal of this chapter, we only show some preliminary results. First, we will show some results from the first step of our algorithm (constructing the bridge model) to see if it can find meaningful partitions. Next, we will investigate the effect of the second and third steps to find out if these steps improve the structure or not. To test the first step, we used one real dataset and one synthetic dataset.

For our real dataset we choose NBA data, so that we can interpret the meaning of the partitions found according to our basic knowledge of basketball. The data contains the statistics for “all time regular seasons stats.”[1] We removed all objects with missing values. It includes 927 players along with 19 attributes. The attributes are related to average performance of players per game. Also, we discretize the attributes into four equal bins. Since we use EM in our approach, each time we run the algorithm we may get slightly different partitions, but some attributes are found to be in the same partition every time. Figure 19 shows those attributes. For example, the attributes of the first partition {OR, DR, TR, BLK} (offensive rebounds, defensive rebounds, total rebounds, and blocks, respectively) are all related to the height of a player which indicates if he is a forward or guard player. Third partition {FG%, 3P%, FT%} (field goal percentage, three point percentage, and free throw percentage, respectively) is obviously related to the shooting accuracy of a player. So we can claim that all introduced partitions are both meaningful and novel (Fig. 20).

Also, in order to achieve a more robust result we have done the experiment merely on the following eight attributes: {3PM, 3PA, 3P%, DR, OR, TR, GP, Min} (three point made, three point attempted, three point percentage, defensive rebounds, offensive rebounds, total rebounds, game played, and minutes played, respectively). In this case we almost found three partitions every time {3PM, 3PA, 3P%}, {DR, OR, TR}, and {GP, Min} which are very intuitive. Note that all three partitions are novel and connote different concepts.

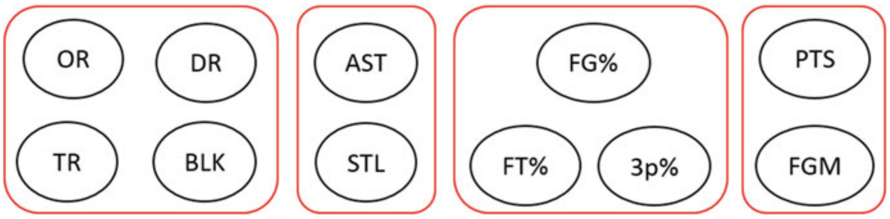

Fig. 19 The attributes which are usually grouped under the same partition for NBA data

[1]The data is available on: http://www.stats.nba.com/leaders/alltime/.

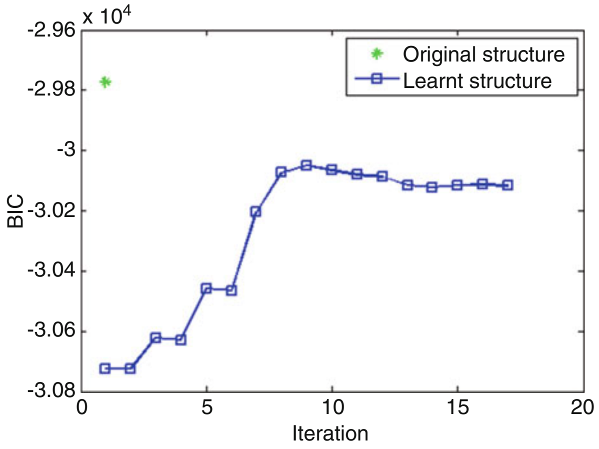

Fig. 20 BIC score of the learnt structures during the second and third steps

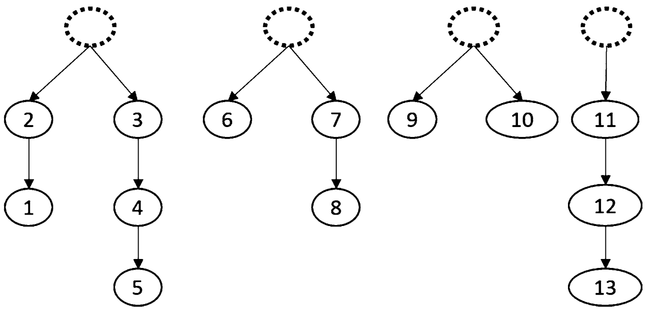

Fig. 21 The original BN for synthetic data with four partitions and 13 attributes

In order to further test the ability of our method, we also used synthetic data. We generated samples from a BN structure with four partitions (Fig. 21). Then we removed the data related to all four partitions and run our algorithm on all 13 observed attributes. The algorithm was able to find five partitions as shown in Fig. 22, which are very close to the original structure. This result is also support the goodness of our method in finding meaningful and novel partitions.

Along with its ability to find meaningful and novel partitions, our algorithm is very time efficient. For NBA data with 19 attributes it only takes 7 min (on average) to find partitions, which is fairly good. We run our algorithm on a computer with Core i7 3.40 GHz Intel CPU.

Now, it is time to see if second and third steps are able to improve the NB model, which is learnt in the first step. So, we have generated 10 random DAGs (each has 12 variables) and we have generated 4000 samples from each one. Next, we hide three variables and run our algorithm on remaining observed variables. In order to measure the effect of the second and third steps, we have compared the

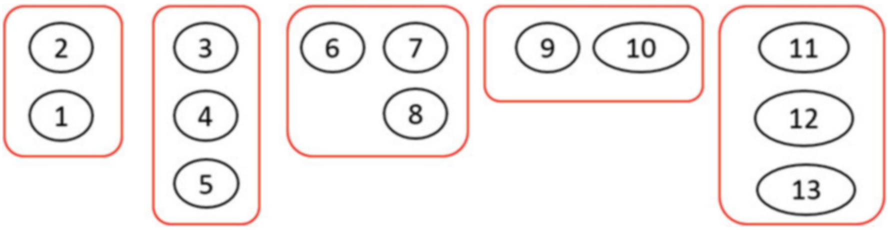

Fig. 22 The partitions found by our second proposed method for the synthetic data

Table 3 Average BIC score in 10 run

Avg initial BIC	Avg final BIC	Avg original BIC
−31137	−30752	−29874

BIC score of NB structure, resulted from the first step, with the BIC score of the final structure. Note that, for the sake of comparison, the BIC score is computed according to the original data (including the values for hidden variables). Table 3 shows the average BIC score for 10 runs. Also, we have measured the original structure's BIC on true data to see how close the resulting structure is to the optimal one. Figure 20 shows a result from one run. As it can be seen, the second and third steps, improve the structure, from the sense of BIC score, over time. Although in some iterations the BIC score may decrease (ninth iteration), but in overall the BIC score will increase in compare to the initial structure and it will get closer to the optimal point, which justifies the usefulness of the second and third steps of our algorithm.

6 Conclusion and Summary

In this chapter, we discussed many model based approaches for both uni- and multi-dimensional clustering and contribute with two new methods for the latter. The aim of model based clustering is to find the best possible model that captures the conditional (in)dependencies exist in the problem and for this purpose, in uni-dimensional clustering, the basic algorithm is BSC and other methods are just a special case of it. However, according to our knowledge, the only graphical model based approach for multi-dimensional clustering is LTMs, which has many limitations due to their assumptions. The main problem of LTMs was their inability to draw an edge between two attributes directly, which will limit their capability of modeling conditional (in)dependencies. Another problem is their inability to model independence partitions and adding attributes to a partition which may be found to be meaningless in the future. Based on these limitations, we proposed two algorithms for multi-dimensional clustering. The first algorithm (Fig. 14) was just a generalization of BSC to multi-dimensional cases and suffers from high time complexity. But, the second method is a combination of MBC and BI (one of the

LTM approaches) methods. We used the idea of grouping edges between variables into three groups from MBCs and used a sibling cluster concept from BI to propose a new method that avoids the mentioned disadvantages of LTMs. Since a comprehensive evaluation and comparison of different methods was beyond the aim of this chapter, we only provide a preliminary evaluation of our second proposed method, which justifies its ability for finding meaningful and novel partitions in an acceptable time.

Acknowledgements This work is funded by a so-called career contract at Linköping University, and by the Swedish Research Council (ref. 2010–4808).

References

1. McLachlan, G., Peel, D.: Finite Mixture Models. Wiley, New York (2004)
2. Peña, J.M., Lozano, J.A., Larrañaga, P.: Learning Bayesian networks for clustering by means of constructive induction. Pattern Recogn. Lett. **20**(11), 1219–1230 (1999)
3. Peña, J.M., Lozano, J.A., Larrañaga, P.: An improved Bayesian structural EM algorithm for learning Bayesian networks for clustering. Pattern Recogn. Lett. **21**(8), 779–786 (2000)
4. Peña, J.M., Lozano, J.A., Larrañaga, P.: Learning recursive Bayesian multinets for data clustering by means of constructive induction. Mach. Learn. **47**(1), 63–89 (2002)
5. Pham, D.T., Ruz, G.A.: Unsupervised training of Bayesian networks for data clustering. Proc. R. Soc. A Math. Phys. Eng. Sci. **465**(2109), 2927–2948 (2009)
6. Van Der Gaag, L.C., De Waal, P.R.: Multi-dimensional Bayesian network classifiers. In: Proceedings of the 3rd European Workshop in Probabilistic Graphical Models, pp. 107–114 (2006)
7. Bielza, C., Li, G., Larrañaga, P.: Multi-dimensional classification with Bayesian networks. Int. J. Approx. Reason. **52**, 705–727 (2011)
8. Sucar, L.E., Bielza, C., Morales, E.F., Hernandez-Leal, P., Zaragoza, J.H., Larrañaga, P.: Multi-label classification with Bayesian network-based chain classifiers. Pattern Recogn. Lett. **41**, 14–22 (2014)
9. Rodríguez, J.D., Lozano, J.A.: Multi-objective learning of multi-dimensional Bayesian classifiers. In: Proceedings of the 8th IEEE International Conference on Hybrid Intelligent Systems, pp. 501–506 (2008)
10. Read, J., Bielza, C., Larrañaga, P.: Multi-dimensional classification with super-classes. IEEE Trans. Knowl. Data Eng. **26**(7), 1720–1733 (2014)
11. Chen, T., Zhang, N.L., Liu, T., Poon, K.M., Wang, Y.: Model-based multidimensional clustering of categorical data. Artif. Intell. **176**(1), 2246–2269 (2012)
12. Liu, T., Zhang, N., Poon, K., Liu, H., Wang, Y.: A novel LTM-based method for multi-partition clustering. In: Proceedings of the 6th European Workshop on Probabilistic Graphical Models, pp. 203–210 (2012)
13. Mourad, R., Sinoquet, C., Zhang, N.L., Liu, T., Leray, P., et al.: A survey on latent tree models and applications. J. Artif. Intell. Res. **47**, 157–203 (2013)
14. Zhang, N.L.: Hierarchical latent class models for cluster analysis. J. Mach. Learn. Res. **5**, 697–723 (2004)
15. Elidan, G., Lotner, N., Friedman, N., Koller, D.: Discovering hidden variables: A structure-based approach. Neural Inf. Process. Syst. **13**, 479–485 (2000)
16. Elidan, G., Friedman, N.: Learning the dimensionality of hidden variables. In: Proceedings of the 17th Conference on Uncertainty in Artificial Intelligence, pp. 144–151 (2001)

17. McLachlan, G., Krishnan, T.: The EM Algorithm and Extensions, vol. 382. Wiley, New York (2007)
18. Friedman, N.: The Bayesian structural EM algorithm. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 129–138 (1998)
19. Mossel, E., Roch, S.: Learning nonsingular phylogenies and hidden markov models. In: Proceedings of the 37th Annual ACM Symposium on Theory of Computing, pp. 366–375 (2005)
20. Darwiche, A.: Modeling and Reasoning with Bayesian Networks. Cambridge University Press, Cambridge (2009)
21. Santafé, G., Lozano, J.A., Larrañaga, P.: Bayesian model averaging of naive Bayes for clustering. IEEE Trans. Syst. Man Cybern. B Cybern. **36**(5), 1149–1161 (2006)
22. Santafé, G., Lozano, J.A., Larrañaga, P.: Bayesian model averaging of TAN models for clustering. In: 3rd European Workshop on Probabilistic Graphical Models, pp. 271–278 (2006)
23. Neapolitan, R.E.: Learning Bayesian Networks, vol. 38. Prentice Hall, Upper Saddle River (2004)
24. Friedman, N., Geiger, D., Goldszmidt, M.: Bayesian network classifiers. Mach. Learn. **29**(2–3), 131–163 (1997)
25. Ramoni, M., Sebastiani, P.: Learning Bayesian networks from incomplete databases. In: Proceedings of the 13th Conference on Uncertainty in Artificial Intelligence, pp. 401–408 (1997)
26. Thiesson, B., Meek, C., Chickering, D.M., Heckerman, D.: Learning mixtures of DAG models. In: Proceedings of the 14th Conference on Uncertainty in Artificial Intelligence, pp. 504–513 (1998)
27. Geiger, D., Heckerman, D.: Knowledge representation and inference in similarity networks and Bayesian multinets. Artif. Intell. **82**(1), 45–74 (1996)
28. Galimberti, G., Soffritti, G.: Model-based methods to identify multiple cluster structures in a data set. Comput. Stat. Data Anal. **52**(1), 520–536 (2007)
29. Guan, Y., Dy, J.G., Niu, D., Ghahramani, Z.: Variational inference for nonparametric multiple clustering. In: Proceedings of the Workshop on Discovering, Summarizing and Using Multiple Clusterings (2010)
30. Herman, G., Zhang, B., Wang, Y., Ye, G., Chen, F.: Mutual information-based method for selecting informative feature sets. Pattern Recogn. **46**(12), 3315–3327 (2013)
31. Zhang, N.L., Kocka, T.: Efficient learning of hierarchical latent class models. In: Proceedings of the 16th IEEE International Conference on Tools with Artificial Intelligence, pp. 585–593 (2004)
32. Poon, L., Zhang, N.L., Chen, T., Wang, Y.: Variable selection in model-based clustering: to do or to facilitate. In: Proceedings of the 27th International Conference on Machine Learning, pp. 887–894 (2010)
33. Poon, L.K., Zhang, N.L., Liu, T., Liu, A.H.: Model-based clustering of high-dimensional data: variable selection versus facet determination. Int. J. Approx. Reason. **54**(1), 196–215 (2013)
34. Liu, T.-F., Zhang, N.L., Chen, P., Liu, A.H., Poon, L.K., Wang, Y.: Greedy learning of latent tree models for multidimensional clustering. Mach. Learn. **98**, 301–330 (2013)
35. Wang, Y., Zhang, N.L., Chen, T.: Latent tree models and approximate inference in Bayesian networks. J. Artif. Intell. Res., 879–900 (2008)
36. Harmeling, S., Williams, C.K.: Greedy learning of binary latent trees. IEEE Trans. Pattern Anal. Mach. Intell. **33**(6), 1087–1097 (2011)
37. Zaragoza, J.C., Sucar, L.E., Morales, E.F.: A two-step method to learn multidimensional Bayesian network classifiers based on mutual information measures. In: Proceedings of Florida Artificial Intelligence Research Society Conference (2011)
38. Zaragoza, J.H., Sucar, L.E., Morales, E.F., Bielza, C., Larrañaga, P.: Bayesian chain classifiers for multidimensional classification. In: Proceedings of the International Joint Conference on Artificial Intelligence, vol. 11, pp. 2192–2197 (2011)

39. Cheng, W., Hüllermeier, E., Dembczynski, K.J.: Bayes optimal multilabel classification via probabilistic classifier chains. In: Proceedings of the 27th International Conference on Machine Learning, pp. 279–286 (2010)
40. Borchani, H., Bielza, C., Martínez-Martín, P., Larrañaga, P.: Predicting EQ-5D from the Parkinson's disease questionnaire PDQ-8 using multi-dimensional Bayesian network classifiers. Biomed. Eng. Appl. Basis Commun. **26**(1), 1450015 (2014)
41. Mihaljevic, B., Bielza, C., Benavides-Piccione, R., DeFelipe, J., Larrañaga, P.: Multi-dimensional classification of GABAergic interneurons with Bayesian network-modeled label uncertainty. Front. Comput. Neurosci. **8**, 150 (2014)
42. Borchani, H., Bielza, C., Toro, C., Larrañaga, P.: Predicting human immunodeficiency virus inhibitors using multi-dimensional Bayesian network classifiers. Artif. Intell. Med. **57**(3), 219–229 (2013)
43. Borchani, H., Bielza, C., Martínez-Martín, P., Larrañaga, P.: Markov blanket-based approach for learning multi-dimensional Bayesian network classifiers: an application to predict the European quality of life-5Dimensions (EQ-5D) from the 39-item Parkinson's disease questionnaire (PDQ-39). J. Biomed. Inform. **45**, 1175–1184 (2012)
44. de Waal, P.R., van der Gaag, L.C.: Inference and learning in multi-dimensional Bayesian network classifiers. In: Proceedings of the 9th European Conference on Symbolic and Quantitative Approaches to Reasoning with Uncertainty, 501–511 (2007)