Discrete model-based clustering with overlapping subsets of attributes

Fernando Rodriguez-Sanchez FERNANDO.RODRIGUEZS@UPM.ES
Pedro Larrañaga PEDRO.LARRANAGA@FI.UPM.ES
Concha Bielza MCBIELZA@FI.UPM.ES
Computational Intelligence Group
Departamento de Inteligencia Artificial
Universidad Politécnica de Madrid, Spain

Abstract

Traditional model-based clustering methods assume that data instances can be grouped in a single "best" way. This is often untrue for complex data, where several meaningful sets of clusters may exist, each of them associated to a unique subset of data attributes. Current literature has approached this problem with models that consider disjoint subsets of attributes to define distinct clustering solutions. Each solution being represented by a cluster variable. However, restricting attributes to a single cluster variable diminishes the expressiveness and quality of these models. For this reason, we propose a novel kind of models that allows cluster variables to have overlapping subsets of attributes between them. In order to learn these models, we propose to combine a search-based method with an attribute clustering procedure. Experimental results with both synthetic and real-world data show the utility of our approach and its competitiveness with the state-of-the-art.

1. Introduction

Real-world applications often involve multifaceted data with several reasonable interpretations. In order to cluster this data, we need methods that are able to produce multiple dissimilar solutions. One of the first approaches that comes to mind when facing this problem is the naïve application of various clustering algorithms (Handl et al., 2005). However, this approach, while conceptually easy to understand, is difficult to apply given our inability to know how many algorithms should be used and how to properly evaluate clustering solutions from different methods.

Multi-partition clustering (MPC) methods avoid these issues by generating multiple clustering solutions, where each of them represents a *partition* of data and provides a different way to cluster it. Also referred to as non-redundant clustering, the MPC problem deals with finding solutions that are both dissimilar and of quality (Gondek and Hofmann, 2007). MPC methods usually rely on data transformations (Davidson and Qi, 2008) or orthogonal subspaces (Cui et al., 2010) to find diverse partitions. However, we advocate for a more interpretable approach called *facet determination* that considers both a cluster variable and a unique subset of attributes (also known as *facet*) in order to define a partition (Poon et al., 2013).

Our main concern through this paper is the combination of discrete model-based clustering with facet determination. In the model-based approach it is assumed that data has been generated by a finite mixture model that is composed of both observed and hidden variables, where the first group corresponds to data attributes while the second one corresponds to cluster variables. In order to apply this model, we must first estimate its parameters. However, the computational cost of their estimation exponentially increases with respect to the number of variables in the model. This cost can be diminished with the use of Bayesian networks, which reduce the number of model parameters by exploiting the conditional independences in data. Zhang (2004) follows this idea and introduces the *hierarchical latent class model* (HLCM), which is a discrete Bayesian network with a rooted tree structure whose leaf nodes correspond to observed variables while the rest are cluster variables. In this model, each partition is represented by a cluster variable and its disjoint set of observed children. While this model is capable of representing multiple partitions, their interpretation is complicated (Poon et al., 2018). This is because in an HLCM cluster variables are dependent, which implies that in order to analyze one of them we have to consider not only the variables of its corresponding facet, but all the observed variables in the model. Taking this limitation into consideration, we propose a novel type of model that is able to represent multiple partitions while also being easily interpretable. Instead of defining partitions with dependent cluster variables and disjoint facets, it is composed of partitions with independent cluster variables and overlapping facets.

The remainder of the paper is organized as follows. In Section 2 we describe in detail model-based clustering with multiple cluster variables and propose our new MPC model. In Section 3 we present our algorithm for learning this type of models. In Section 4 we empirically study our MPC procedure using both synthetic and real-world data. Finally, we report our conclusions and suggest future lines of study in Section 5.

2. Model-based MPC

Let $\mathcal{D} = \{\boldsymbol{d}^1, ..., \boldsymbol{d}^N\}$ be a set of N independent, identically distributed (i.i.d.) data instances with an associated set of discrete attributes $\mathbf{X} = \{X_1, ..., X_n\}$. Traditional model-based clustering algorithms assume that data has been generated by a probability distribution $\boldsymbol{P}(\boldsymbol{X})$ that can be expressed as a finite mixture of K cluster-specific *conditional probability distributions* (CPDs). In this model, each mixture component $\boldsymbol{P}(\boldsymbol{X} \mid C)$ represents the attributes' CPD given the cluster variable C, whose probability distribution is referred to as $\boldsymbol{P}(C)$. That is,

$$\boldsymbol{P}(\mathrm{X}) = \sum_{C} \boldsymbol{P}(C)\ \boldsymbol{P}(\mathrm{X} \mid C)\ .$$

This model assumes that there is a single partition whose facet is composed of all the observed variables in the model (X). In contrast, MPC methods make a different assumption about the mixture model that generated the data: there are several cluster variables $\mathbf{C} = \{C_1, ..., C_t\}$ with different cardinalities $\mathbf{K} = \{K_1, ..., K_t\}$. In this model, each cluster variable C_i corresponds to a unique partition of data, i.e.,

$$\boldsymbol{P}(\mathrm{X}) = \sum_{C_1} \cdots \sum_{C_t} \boldsymbol{P}(C_1, ..., C_t)\ \boldsymbol{P}(\mathrm{X} \mid C_1, ..., C_t)\ ,$$

where $\boldsymbol{P}(C_1, ..., C_t)$ represents the joint probability distribution of the cluster variables. Estimating the maximum-likelihood parameters of this model is not trivial given the hidden nature of cluster variables. For this reason, the parameter learning process requires a method that is able to handle incomplete data, such as the *expectation-maximization* (EM) algorithm (Dempster et al., 1977). The EM algorithm is able to estimate the model parameters by following a two-step iterative process. First, on its expectation step, it completes the missing data and generates an empirical distribution. Second, on its maximization step, it uses the expected distribution to estimate the parameters. Each step of the algorithm results in a new set of parameters that improves the likelihood of the model. This iterative process will continue until a convergence or stop condition is met. Once the model is fitted, the posterior probability $\boldsymbol{P}(C_1 = c_1, ..., C_t = c_t \mid \mathrm{X})$ is estimated using the Bayes theorem:

$$\boldsymbol{P}(C_1 = c_1, ..., C_t = c_t \mid \mathrm{X}) = \frac{\boldsymbol{P}(C_1 = c_1, ..., C_t = c_t)\ \boldsymbol{P}(\mathrm{X} \mid C_1 = c_1, ..., C_t = c_t)}{\sum\limits_{C_1} \cdots \sum\limits_{C_t} \boldsymbol{P}(C_1, ..., C_t)\ \boldsymbol{P}(\mathrm{X} \mid C_1, ..., C_t)} . \qquad (1)$$

2.1 Model-based MPC with Bayesian networks

Since the denominator of Equation (1) is constant with respect to $\{c_1, ..., c_t\}$, our main challenge is computing the numerator:

$$\boldsymbol{P}(C_1 = c_1, ..., C_t = c_t \mid \mathrm{X}) \propto \boldsymbol{P}(C_1 = c_1, ..., C_t = c_t)\ \boldsymbol{P}(\mathrm{X} \mid C_1 = c_1, ..., C_t = c_t) . \qquad (2)$$

However, given the exponential increase in model parameters with respect to the number of variables this computation becomes cumbersome when there is a large number of them in the model. It is therefore necessary to exploit the conditional independences that are present in data.

Two random variables X and Y are *conditionally independent* (c.i.) given another random variable Z if, for any assignment of values $X = x$, $Y = y$, $Z = z$, knowing the value of Y doesn't affect the probability of X when the value of Z is already known, i.e, $\boldsymbol{P}(X \mid Y, Z) = \boldsymbol{P}(X \mid Z)$.

Conditional independence is a central concept to *Bayesian networks* (BNs) (Pearl, 1988). When conditional independences are present, BNs produce a factorization of the joint probability distribution that substantially reduces the number of parameters. Given a set of variables $\mathbf{Y} = \{\boldsymbol{X}, \boldsymbol{C}\}$, a BN is defined by:

- A *directed acyclic graph* $\mathcal{G}$ that comprises the structure of $\boldsymbol{\mathcal{B}}$ and represents the conditional independences among the variables.

- A set of *parameters* $\boldsymbol{\theta}$ that represents the CPDs of each variable $Y_i \in \mathbf{Y}$ given its parents $\mathbf{Pa}_{\mathcal{G}}(Y_i)$ according to $\mathcal{G}$.

$\boldsymbol{\mathcal{B}}(\mathcal{G}, \boldsymbol{\theta})$ is a BN with respect to $\mathcal{G}$ if and only if it satisfies the local Markov property: each variable Y_i is c.i. of its non-descendants given its parent variables $\mathbf{Pa}_{\mathcal{G}}(Y_i)$. Accordingly, the application of the BN factorization allows us to simplify Equation (2) by considering the conditional independences that are expressed by the network structure:

$$\boldsymbol{P}(C_1, ..., C_t \mid \boldsymbol{X}) \propto \prod_i \boldsymbol{P}(C_i \mid \mathbf{Pa}_{\mathcal{G}}(C_i)) \prod_j \boldsymbol{P}(X_j \mid \mathbf{Pa}_{\mathcal{G}}(X_j)) ,$$

where $\mathbf{Pa}_{\mathcal{G}}(C_i) \subseteq \boldsymbol{C}$ and $\mathbf{Pa}_{\mathcal{G}}(X_j) \subseteq \boldsymbol{Y}$. Hence, by using a BN we need to learn not only the model parameters, but also its network structure. However, given the presence of hidden variables, the structure learning process requires the EM algorithm on each step, which becomes very computationally demanding (Friedman, 1997). Therefore, restricted models that reduce the space of possible structures, such as HLCMs, are preferred over unrestricted ones.

HLCMs are a generalization of *latent class models* (LCMs). An LCM is a discrete BN consisting of a cluster variable C and a set of observed variables that are conditionally independent given the value of C. Figure 1(a) presents an example of this model. Given their restriction to a single cluster variable, LCMs are not suitable for MPC. HLCMs extend this model by allowing multiple

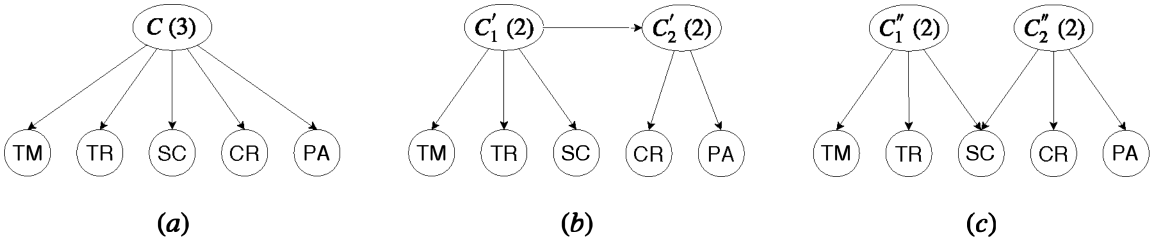


Figure 1: Examples of (a) an LCM, (b) an HLCM and (c) an OLCM for a hypothetical citizen survey data. The numbers between parenthesis represent the respective number of clusters.

hierarchically connected cluster variables, where each variable in the model only has one parent that is always a cluster variable. An example of this model is shown by Figure 1(b).

However, the structure of HLCMs may produce interpretation problems in MPC. To illustrate them, we have prepared an hypothetical study that aims at finding interesting groups of citizens with respect to their satisfaction on different areas. Their level of satisfaction (low, medium or high) is evaluated using five questions, each one corresponding to a different aspect: TM (Tax Management), TR (Tax Rates), SC (System Corruption), CR (Crime Rate) and PA (Police Actuation).

The HLCM presented in Figure 1(b) considers the existence of two partitions, each of them represented by a cluster variable C'_i and its disjoint set of observed variables. The first one, C'_1, is directly connected to the observed variables TM, TR and SC, while C'_2 is connected to CR and PA. However, these partitions are not independent. The arc $C'_1 \rightarrow C'_2$ establishes a dependence between their cluster variables, regardless of any evidence about other variables in the model.

The graphical concept of *d-separation* (Koller and Friedman, 2009) allows us to infer conditional independences in the model by simply analyzing its graph structure. By applying this concept on the HLCM of Figure 1(b), we notice that all the observed variables in the model need to be considered in order to interpret each cluster variable. For example, when interpreting C'_2, we need to consider the values of TM, TR and SC even though they are not directly related to C'_2. In our opinion, this makes HLCM partitions counterintuitive with respect to the definition of facet determination: we cannot understand the meaning of a cluster variable by only taking into consideration its subset of observed variables. For this reason, in the following section we propose a novel MPC model whose cluster variables can be independently analyzed by simply considering its respective subsets of observed variables, which may be overlapped.

2.2 Overlapping Latent Class Models

An *overlapping latent class model* (OLCM) is a discrete Bayesian network where the following conditions are met:

1. There only exist arcs from cluster to observed variables.

2. Any cluster variable must have at least two observed children.

Figure 1(c) illustrates an example of an OLCM that is composed of two partitions with C''_1 and C''_2 as their respective cluster variables. An important difference between OLCMs and HLCMs lies in their relationships between partitions. While HLCMs are characterized for representing partitions that are directly connected via their cluster variables, OLCMs consider indirect connections by allowing cluster variables to share observed children. An example of this behavior occurs with the observed variable SC, which relates $C_1^{''}$ and $C_2^{''}$ given their common effect onto it. When this type of connection $C_1^{''} \rightarrow$ SC $\leftarrow C_2^{''}$ exists, both cluster variables are independent unless the value of SC is known. Unlike HLCMs, this allows us to interpret the meaning of a cluster variable by simply considering its observed children. Therefore, we might define the first OLCM partition as “tax policies satisfaction”, given that is composed of the TM, TR and SC variables, and the second partition might be named “security policies satisfaction”, given it is composed of CR, PA and SC.

3. Learning OLCMs

In this section we propose the OLHC learning algorithm, a search-based method that hill-climbs the space of OLCMs. A hill-climbing procedure requires the specification of three aspects. First, the set of operators that delimit its space of models (Section 3.1). Second, the search process that defines how these operators are applied and which scoring metric is used to evaluate the quality of a model (Section 3.2). Finally, the initial model of the algorithm (Section 3.3).

3.1 OLHC search operators

OLHC uses six operators to alter the structure of an OLCM. The first two are the *cluster addition* (CA) and *cluster removal* (CR) operators, which are responsible for estimating the cardinalities of cluster variables in the model. Given an OLCM with at least one cluster variable, the application of the CA operator returns a new model with a new cluster in the chosen variable, thus increasing its cardinality by one. Alternatively, the CR operator removes a cluster of the variable, thus reducing its cardinality by one, where the minimum number of possible clusters is two.

The *arc addition* (AA) and *arc removal* (AR) operators constitute its second set and are responsible for modifying the composition of current partitions. The AA operator involves one cluster variable C_i and one observed variable X_j that is not currently a child of C_i. Its application produces an OLCM with a new arc $C_i \rightarrow X_j$, thus adding Xj to the partition of C_i. On the contrary, the AR operator yields the opposite model. Instead of adding an arc from C_i to X_j, AR removes it.

The last pair is composed of the *partition addition* (PA) and *partition removal* (PR), which are responsible for both changing the number and composition of current partitions. The PA operator involves two observed variables X_i and X_j that do not belong to the same partition. By applying this operator a new OLCM is created with a cluster variable C_{new} as their parent. In order to reduce the complexity of this operation, we assume that the number of clusters in C_{new} is two. On the other hand, the PR operator creates a new model by removing a cluster variable (including its arcs) and reallocating each of its observed variables in the best fitting partition. This reallocation process is done in a greedy iterative way: each variable is assigned to one of the current partitions only if the scoring metric increases, where it always selects the one that produces the highest scoring model. This process continues until it has tried to reallocate all the observed variables, where their order is given by X. Note this is the most expensive operator because it requires the application of an intrinsic iterative process.

Algorithm 1: OLHC algorithm.

```
Arguments: A data set $\mathcal{D}$ with a set of attributes $\mathbf{X}$
Result    : An OLCM ($M_0$) over $\mathbf{X}$
1  $M_0 \leftarrow$ attributeClustering($\mathcal{D}$);
2  while true do
3      $M_1 \leftarrow$ expand($M_0, \mathcal{D}$);
4      if $BIC(M_1 \mid \mathcal{D}) \leq BIC(M_0 \mid \mathcal{D})$ then
5          return $M_0$;
6      while $keepExpansion$ do
7          $M_1' \leftarrow$ expand($M_1, \mathcal{D}$);
8          if $BIC(M_1' \mid \mathcal{D}) \leq BIC(M_1 \mid \mathcal{D})$ then
9              $keepExpansion \leftarrow$ false;
10         else
11             $M_1 \leftarrow M_1'$;
12     $M_2 \leftarrow$ simplify($M_1, \mathcal{D}$);
13     if $BIC(M_2 \mid \mathcal{D}) \leq BIC(M_1 \mid \mathcal{D})$ then
14         return $M_1$;
15     while $keepSimplification$ do
16         $M_2' \leftarrow$ simplify($M_2, \mathcal{D}$);
17         if $BIC(M_2' \mid \mathcal{D}) \leq BIC(M_2 \mid \mathcal{D})$ then
18             $keepSimplification \leftarrow$ false;
19         else
20             $M_2 \leftarrow M_2'$;
21     $M_0 \leftarrow M_2$;
```

3.2 OLHC search process

Our hill-climbing procedure (Algorithm 1) organizes the previous operators into two groups. The first one is called the *expansion operators* and it is composed of CA, AA and PA. Its purpose is to improve the model quality by increasing its complexity. The second group is called the *simplification operators* and its is composed of SR, AR and PR. It applies the opposite approach, it improves the model quality by removing unnecessary complexities.

Each iteration of the algorithm is divided into two phases: expansion and simplification. In each phase, the `expand` (line 3) and `simplify` (line 12) methods apply all the operators from their corresponding groups and store the highest scoring model. Each phase is repeated until the score ceases to increase, where the iteration through these phases continues until any of them cannot find a better model. This process is inspired by the work of Chen et al. (2012).

We advocate for the *Bayesian information criterion* (BIC) as our scoring metric due to its intrinsic penalization of complex models. The BIC score of an OLCM M is defined as:

$$BIC(M \mid D) = \log P(D \mid M) - \frac{dim(M)}{2} \log N \ ,$$

Algorithm 2: OLHC.attributeClustering.

```
Arguments: A data set $\mathcal{D}$ with an associated set of attributes $\mathbf{X}$
Result   : An OLCM ($M_{init}$) over $\mathbf{X}$
Calculate the empirical NMI between each pair of attributes in $\mathbf{X}$;
Let $\mathbf{V} \leftarrow \mathbf{X}$ and $\mathbf{L} \leftarrow \emptyset$;
while $\mathbf{V} \neq \emptyset$ do
    if $\mathbf{L} = \emptyset$ then
        $\mathbf{P} \leftarrow$ the pair of attributes with the highest NMI;
        $M_L \leftarrow$ Learn an LCM with $\mathbf{P}$ as its observed variables;
        Let $\mathbf{L} \leftarrow \mathbf{L} \cup \{M_L\}$ and $\mathbf{V} \leftarrow \mathbf{V} \setminus \mathbf{P}$;
    else
        Let $H_{best} \leftarrow 0$ and $\mathbf{V}_{best} \leftarrow \emptyset$ ;
        for each $V_i \in \mathbf{V}$ do
            for each $M_j \in \mathbf{L}$ do
                $M \leftarrow$ Learn an LCM whose observed variables are the ones in $M_j$ plus $V_i$;
                $H_{inc} \leftarrow DH_{avg}(M) - DH_{avg}(M_j)$;
                if $H_{inc} > H_{best}$ then
                    Let $H_{best} \leftarrow H_{inc}$ and $\mathbf{V}_{best} \leftarrow V_i$;
                    Let $M_{old} \leftarrow M_j$ and $M_{new} \leftarrow M$;
        if $H_{best} = 0$ then
            $\mathbf{P} \leftarrow$ the pair of attributes with the highest NMI;
            $M_L \leftarrow$ Learn an LCM with $\mathbf{P}$ as its observed variables;
            Let $\mathbf{L} \leftarrow \mathbf{L} \cup \{M_L\}$ and $\mathbf{V} \leftarrow \mathbf{V} \setminus \mathbf{P}$;
        else
            Let $\mathbf{L} \leftarrow \mathbf{L} \setminus \{M_{old}\}$ and $\mathbf{L} \leftarrow \mathbf{L} \cup \{M_{new}\}$;
$M_{init} \leftarrow$ Form an OLCM by combining all the LCMs in $\mathbf{L}$;
```

where $dim(M)$ corresponds to the model's dimension, i.e., the number of free parameters in M, and N is the number of data instances. However, in order to apply this process we need to choose an appropriate initial model (see next section).

3.3 OLHC Initialization

In order to create a feasible initial model for the OLHC search process, we apply the attribute clustering procedure described in Algorithm 2. The first step of `attributeClustering` consists on measuring the similarity between each pair of data attributes. In order to do so, we compute their dependence in terms of *mutual information* (MI). MI has interesting properties such as always being positive and symmetric, while being zero when both attributes are statistically independent. The MI between two discrete attributes X_i and X_j is defined as

$$MI(X_i;X_j) = \sum_{x_i \in X_i} \sum_{x_j \in X_j} P(x_i,x_j)\ log\frac{P(x_i,x_j)}{P(x_i)P(x_j)} .$$

However, normalization is necessary to compensate the MI bias towards multivalued uniform variables and to restrict its range to [0,1]. For this reason, we define the *normalized mutual information* (NMI) of X_i and X_j as the division of $MI(X_i;X_j)$ by their minimum individual entropies, where the entropy of an attribute X is defined by $H(X) = \sum_{x \in X} \boldsymbol{P}(x)\ \boldsymbol{log}\boldsymbol{P}(x)$. This normalization approach is proposed by Kvalseth (1987).

Once the algorithm has computed the NMI between each pair of attributes, it produces the set of partitions that will form the initial OLCM (M_{init}). Each of these partitions has an associated disjoint subset of observed variables and it is represented by an LCM. In order to produce M_{init}, we first generate a series of LCMs (**L**) and then combine them into a single OLCM. This process consists on a loop that iterates through the set of attributes until all of them belong to an LCM.

Initially, if **L** is empty, `attributeClustering` forms a new LCM using the pair of attributes with the highest NMI (lines 4-7). Otherwise, it tries to find the LCM (M_{new}) whose clustering quality is most improved by introducing one of the current attributes in **V** (lines 9-16). This is done in a greedy manner, where only one attribute (V_{best}) is selected on each iteration. If our method is unable to find an LCM that improves its clustering quality, it simply forms a new LCM using the pair of attributes with the highest NMI (lines 17-20).

We propose to quantify the clustering quality of an LCM using the *Hellinger distance* (Hellinger, 1909), which quantifies the similarity between two probability distributions and it is normalized for the range $[0, 1]$. Let M be a LCM with a set of attributes $\mathbf{X}^{'} \subseteq \mathbf{X}$ and a cluster variable C with two clusters $\{c_1, c_2\}$, the Hellinger distance between its mixture components is denoted by $DH(\boldsymbol{P}(\mathrm{X}^{'} \mid C = c_1), \boldsymbol{P}(\mathrm{X}^{'} \mid C = c_2))$ and it is defined as

$$DH(\boldsymbol{P}(\mathrm{X}^{'} \mid C = c_1), \boldsymbol{P}(\mathrm{X}^{'} \mid C = c_2)) = \sqrt{1 - \sqrt{\sum_{\mathbf{X}'} \boldsymbol{P}(X' \mid C = c_1)\boldsymbol{P}(X' \mid C = c_2)}}\ . \quad (3)$$

Therefore our objective is to maximize this distance, thus improving the separation and compactness of the LCM clusters. Since equation (3) can only be applied to pairs of clusters, we simply calculate the average Hellinger distance (DH_{avg}) when C has more than two clusters. In addition, given that DH is symmetric, we only need to operate with its triangular matrix.

4. Experiments

We conducted experiments with both synthetic and real-world data sets to test our new MPC method and model. All of them were executed on a desktop computer with an Intel Core i7 processor and 32GB of RAM. We only used model-based clustering methods in our study, all of them with the same configuration for the EM algorithm: BIC as the scoring metric with a threshold of 0.001 (i.e., the EM is stopped if a model score does not surpass the threshold) and a maximum of 400 iterations. In order to avoid getting stuck into a local maximum, we employed the strategy of multiple restarts proposed by Chickering and Heckerman (1997), where the number of restarts was 64.

4.1 Experiments with synthetic data

The purpose of these experiments was to examine the ability of OLHC to recover models from data. We evaluated our algorithm five times using the OLCMs presented in Figure 2, each model composed of a different structure. Model (a) is comprised of four partitions, each of them with a disjoint set of quaternary observed variables. Model (b), on the other hand, is comprised of

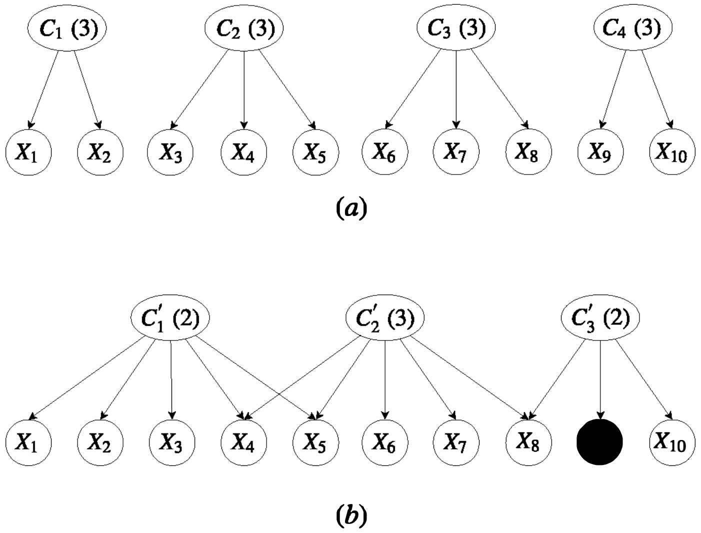


Figure 2: The synthetic OLCMs that were created to test the OLHC algorithm.

three partitions, where there is some overlapping between their facets and each observed variable is binary. Two data sets of 5000 instances were sampled from each model, one for training and one for testing, where cluster information was hidden during training.

All in all, the OLHC algorithm was able to fully retrieve the underlying structures of data. The only discrepancy occurred with model (b), where an additional arc was found from C_3 to X_5. In addition, the score differences between learned and original models were negligible (the mean difference of log-likelihood score with testing data was lesser than 0.01). We attribute this small score variation to the combination of a successful structure retrieval with a multiple restart approach in the EM algorithm.

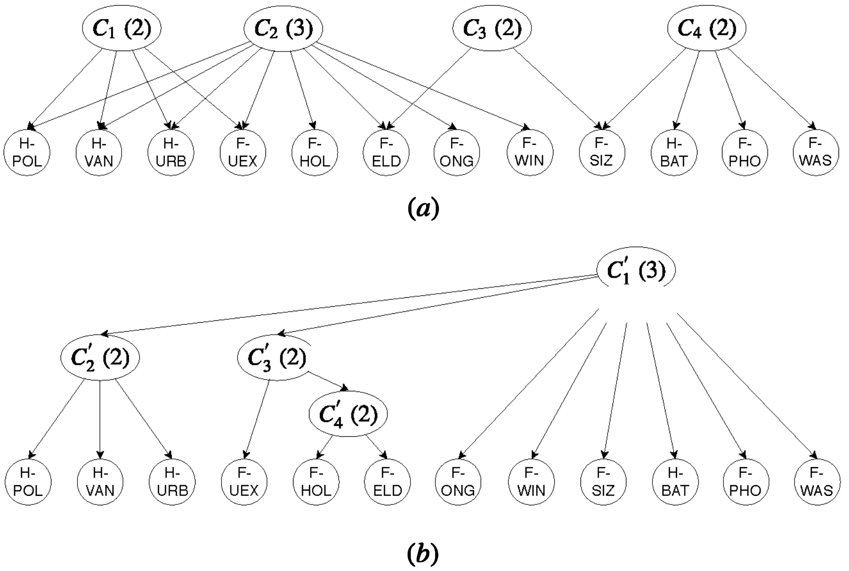
4.2 Experiments with real-world data

Our empirical investigation with real-world data has two goals. First, we want to evaluate the ability of OLHC to generate models that represent the underlying probability distribution of a given data set. Second, we want to compare our resulting OLCM partitions with those found by the state-of-the-art methods. We carried out our experiments on data from a survey on Spanish family life conditions in 2016, which has been made public by the Spanish Statistical Office. This data set consists of 12 discrete attributes, each corresponding to a survey question, and 14,196 data instances. Survey questions are binary (yes/no) with the exception of H-URB and F-SIZ, which have three and five states respectively. They are also organized into two groups: family (attributes starting with "F-") and home (attributes starting with "H-").

Our comparative study includes both traditional and multi-partition model-based clustering methods. The first of them is a LCM-based hill-climbing method (referred to as LCM-HC) whose only operator is the *cluster addition* (this operator acts identically to the one presented in Section 3.2). This algorithm starts with an LCM with two clusters and then keeps increasing its cluster variable cardinality until the score ceases to increase. On the MPC side, we selected two state-

Algorithm	Log-likelihood	BIC	# parameters	Time (s)
LCM-HC	72633 ± 50	73091 ± 2	99	79 ± 8
BI	72665 ± 41	73004 ± 29	86	$\mathbf{22 \pm 1}$
EAST	72688 ± 12	72980 ± 4	$\mathbf{60}$	209 ± 26
OLHC	$\mathbf{72575 \pm 15}$	$\mathbf{72967 \pm 8}$	78	25054 ± 1358

Table 1: Spanish survey data results.



(*a*)

(*b*)

Figure 3: Structures of two learned models from the Spanish survey data. (a) shows the OLHC result, and (b) shows the EAST result. Attributes meaning: H-POL (Pollution), H-VAN (Vandalism), H-URB (Urbanization Degree), F-UEX (Afford Unexpected Expenses), F-HOL (Afford Holidays), F-ELD (Elders at Home), F-ONG (ONG Help), F-WIN (Afford Winter), F-SIZ (Family Size), H-BAT (Have Bathtub), F-PHO (Have Phone) and F-WAS (Have Washer).

of-the-art HLCM-based methods. The first one is the *expansion, adjustment, simplification until termination* (EAST) algorithm (Chen et al., 2012). This hill-climbing method improves the work of Zhang (2004) by simultaneously performing the structure search and the estimation of cluster variables cardinality. The second method is the *bridged-islands* (BI) algorithm (Liu et al., 2013), which first produces a set of LCMs using an attribute clustering procedure based on pairwise MI and then combines them into a single HLCM.

The results of five executions are presented in Table 1 . Note that the OLHC algorithm was the best at modeling the underlying probability distribution, visible on its log-likelihood and BIC score. However, it took longest to learn, mainly due to it using a full EM process on each iteration. Both BI and EAST use an EM approximation called *local-EM* (Chen et al., 2012) that significantly reduces their learning time. BI is specially fast for two reasons. First it produces an HLCM by combining a set of previously learned LCMs whose number of attributes is small. Second the combination of LCMs into an HLCM is done using Chow-Liu's algorithm (Chow and Liu, 1968). Finally, it is also interesting to note that LCM-HC produced a model that was unable to beat any of the MPC algorithms, even with a higher number of parameters. This suggest us that a single partition may not be enough for this case.

Figure 3 presents a comparison of the models resulted from the execution of OLHC (a) and EAST (b). In order to measure the quality of a clustering solution we used the Hellinger distance and we also used the NMI as a measure of clustering similarity. We chose the EAST algorithm because it returned the best fitting HLCM. Our OLCM is composed of four partitions, each of them representing a unique concept: C_1 groups instances by its home location characteristics, C_2 forms a general economic partition that combines both home and family attributes, C_3 is connected to observed variables that are related to aspects of the family composition and C_4 refers to family quality of life. Alternatively, EAST also produced four partitions with similar intuitive meanings: $C_1^{'}$ represents a general economic partition, $C_2^{'}$ relates to home location attributes, while both $C_3^{'}$ and $C_4^{'}$ refer to the relationship between elder caring and family economy.

After analyzing each model cluster variables, we conclude that C_1 is the only one able to properly separate instances with respect to their respective economic situation. This is because it has an average Hellinger distance of 0.82 in comparison to the 0.77 of the EAST algorithm. Interestingly, C_1 has a NMI of 0.93 with the cluster variable of the LCM-HC solution, while having two less clusters. C_1 groups instances into "bad" (11%), "regular" (27%) and "good" (62%) economic status. The equivalent EAST partition, $C_1^{'}$ also produces three clusters, but two of them are a combination of "regular" and "good" economic status which complicates the analysis. We believe this is related to its worse clustering quality.

5. Conclusions and future work

In this paper we propose OLCMs as a new type of MPC model that is composed of independent cluster variables and overlapping subsets of observed variables. In order to learn this type of models, we present the OLHC algorithm, which combines a hill-climbing method with an attribute clustering procedure to produce quality OLCMs from data. Our results show the ability of OLHC for learning these models and their advantage over the state-of-the-art.

A drawback of this method lies in its execution time, being much slower than current model-based MPC alternatives. We argue that this difference is caused by their use of the local-EM approximation, which produces a reduction in both execution time and model quality. However, in contrast to these methods, we advocate for the *Structural EM* algorithm (Friedman, 1997) as a way to remove this weakness. By using this algorithm we also intend to expand our set of possible structures, thus producing more interpretable MPC models.

Acknowledgments

This work has been partially supported by the Spanish Ministry of Economy and Competitiveness through the Cajal Blue Brain (C080020-09; the Spanish partner of the Blue Brain initiative from EPFL) and TIN2016-79684-P projects, by the Regional Government of Madrid through the S2013/ICE-2845-CASI-CAM-CM project, and by Fundación BBVA grants to Scientific Research Teams in Big Data 2016.

References

T. Chen, N. L. Zhang, T. F. Liu, L. K. Poon, and Y. Wang. Model-based multidimensional clustering of categorical data. *Artificial Intelligence*, 176(1):2246–2269, 2012.

D. M. Chickering and D. Heckerman. Efficient approximations for the marginal likelihood of Bayesian networks with hidden variables. *Machine Learning*, 29(2-3):181–212, 1997.

C. Chow and C. Liu. Approximating discrete probability distributions with dependence trees. *IEEE Transactions on Information Theory*, 14(3):462–467, 1968.

Y. Cui, X. Z. Fern, and J. G. Dy. Learning multiple nonredundant clusterings. *ACM Transactions on Knowledge Discovery and Data Mining*, 4(15), 2010.

I. Davidson and Z. Qi. Finding alternative clusterings using constraints. *Eighth IEEE International Conference on Data Mining*, pages 773–778, 2008.

A. P. Dempster, N. M. Laird, and D. B. Rubin. Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B*, pages 1–38, 1977.

N. Friedman. Learning belief networks in the presence of missing values and hidden variables. *Fourteenth International Conference on Machine Learning*, pages 125–133, 1997.

D. Gondek and T. Hofmann. Non-redundant data clustering. *Knowledge and Information Systems*, 12(1):1–24, 2007.

J. Handl, J. Knowles, and D. B. Kell. Computational cluster validation in post-genomic data analysis. *Bioinformatics*, 21(15):3201–3212, 2005.

E. Hellinger. Neue begründung der theorie quadratischer formen von unendlichvielen veränderlichen. *Journal für die reine und angewandte Mathematik*, 136:210–271, 1909.

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

T. O. Kvalseth. Entropy and correlation: Some comments. *IEEE Transactions on Systems, Man, and Cybernetics*, 17(3):517–519, 1987.

T. F. Liu, N. L. Zhang, P. Chen, A. H. Liu, L. K. Poon, and Y. Wang. Greedy learning of latent tree models for multidimensional clustering. *Machine Learning*, 98(1-2):301–330, 2013.

J. Pearl. *Probabilistic Reasoning in Intelligent Systems: Networks of Plausible Inference*. Morgan Kaufmann Publishers Inc., 1988.

L. K. Poon, N. L. Zhang, T. Liu, and A. H. Liu. Model-based clustering of high dimensional data: Variable selection versus facet determination. *International Journal of Approximate Reasoning*, 54(1):196–215, 2013.

L. K. Poon, A. H. Liu, and N. L. Zhang. UC-LTM: Unidimensional clustering using latent tree models for discrete data. *International Journal of Approximate Reasoning*, 92:392–409, 2018.

N. L. Zhang. Hierarchical latent class models for cluster analysis. *Journal of Machine Learning Research*, 5:697–723, 2004.