Journal of Artificial Intelligence Research ? (????) ??-?? Submitted 08/10; published ??/??

Multiple Clusterings with Pouch Latent Tree Models

Leonard K. M. Poon† LKMPOON@CSE.UST.HK
Nevin L. Zhang† LZHANG@CSE.UST.HK
Tao Chen‡ TAO.CHEN@SUB.SIAT.AC.CN
Tengfei Liu† LIUTF@CSE.UST.HK
Yi Wang† WANGYI@CSE.UST.HK

†*Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong, China*

‡*Shenzhen Institutes of Advanced Technology, Chinese Academy of Sciences, 1068 Xueyuan Avenue, Shenzhen University Town, Shenzhen, China*

Abstract

Variable selection for cluster analysis, though often used for high-dimensional data, can be a difficult problem. The difficulty originates not only from the lack of class information but also the fact that high-dimensional data are often multifaceted and can be meaningfully clustered in multiple ways. In such a case the effort to find one subset of attributes that presumably gives the "best" clustering may be misguided. It makes more sense to facilitate variable selection by domain experts with more than one clusterings, that is, to systematically identify various facets of a data set (each being based on a subset of attributes), cluster the data along each one, and present the results to the domain experts for appraisal and selection.

In this paper, we propose a model-based method for multiple clusterings using a generalization of Gaussian mixture models. We show its ability to cluster data along multiple facets, and demonstrate that this multiple-clustering approach is often more reasonable to variable selection. Some recent methods for multiple clusterings also address the multiple facets of data. We compare our proposed method with some of those methods on synthetic and real-world data. We further demonstrate the features of the proposed method using an exploratory analysis on NBA data.

1. Introduction

Clustering aims to partition data into groups of similar objects. The cluster structure of interest to domain experts can often be best described using a subset of attributes. The inclusion of other attributes can degrade clustering performance and complicate cluster interpretation. Variable selection methods have been proposed to deal with this issue (e.g. Dy & Brodley, 2004; Steinley & Brusco, 2008; Zeng & Cheung, 2009).

In classification, variable selection is a clearly defined problem, i.e., to find the subset of attributes that gives the best classification performance. The problem is less clear for cluster analysis due to the lack of class information. Our work is based on two observations. First, while clustering algorithms identify clusters in data based on the characteristics of data, domain experts are ultimately the ones to judge the interestingness of the clusters found. Second, high-dimensional data are often multifaceted in the sense that there may be multiple meaningful ways to partition them. The first observation is the reason why

variable selection for clustering is such a difficult problem, whereas the second one suggests that the problem may be ill-conceived from the start.

For example, we may be interested to do cluster analysis on some game statistics of basketball players. The data may contain attributes such as number of minutes played, number of rebounds collected, three-pointer percentage, etc (more details are given in Section 9). There are two possible sources of heterogeneity on this data. One possible source is due to the role of a player (whether he starts a game or not), another is due to the position of the player. If we perform clustering with variable selection on this data, it may find a clustering based on either one of these sources of heterogeneity. If it selects attributes related to minutes played, it will find a clustering based on the role of player. If it selects attributes related to rebounds and three-pointers, it will find a clustering based on the position of player. Either way, it will give up some meaningful information from this data, because it is limited to have only one clustering. In case it selects all attributes on this data, the resulting clustering will be based on a blend of the role and position of player. This will lead to a larger number of clusters and make it more difficult to interpret the result. This will defeat the purpose of variable selection.

Therefore, instead of performing variable selection, we advocate to facilitate variable selection by domain experts with more than one clusterings. The idea is to systematically identify all the different facets of a data set, cluster the data along each one, and present the results to the domain experts for appraisal and selection. The analysis would be useful if at least one of the clusterings is found interesting.

To realize the idea, we generalize Gaussian mixture models (GMM) to allow multiple latent variables. For computational tractability, we restrict that each attribute can be connected to only one latent variable and the relationships among the latent variables can be represented as a tree. The result is what we call pouch latent tree models (PLTMs). Analyzing data using a PLTM may result in multiple latent variables. Each latent variable represents a partition (clustering) of the data and is usually related primarily to only a subset of attributes. Consequently, the data is clustered along multiple dimensions and the results can be used to facilitate variable selection.

Some methods produce more than one clusterings as a outcome of cluster analysis on a data set. We refer to these methods as *multiple-clustering* methods. These methods have recently been proposed to address the multiple facets of data. However, most of these methods are distance-based. In contrast, we propose a model-based method using PLTMs. One advantage of a model-based approach is that it allows statistical interpretation of data through the models. Another advantage is that it allows model selection so that no knowledge on the number of clusterings or the numbers of clusters are required. We demonstrate these advantages in an exploratory analysis on some basketball data with PLTMs.

Our contributions in this paper are two-folded. First, we generalize GMMs with PLTMs and develop algorithms for inference and learning on PLTMs. Second, we propose PLTMs for model-based multiple clusterings. We empirically compare this method with variable selection methods and other multiple-clustering methods and show that PLTM analysis is better than the other methods.

The rest of the paper is organized as follows. We first review some background on model-based clustering with GMMs in Section 2. Then we review the related work in

Section 3. In Section 4, we introduce PLTMs. Learning PLTMs from data is described in Sections 5, 6, and 7. After that, we empirically compare PLTM with variable selection methods and multiple-clustering methods in Section 8. In Section 9, we demonstrate the features of PLTM analysis using an exploratory analysis on NBA data. Finally, we conclude this paper in Section 10.

2. Background on Traditional Model-Based Clustering

Finite mixture models (FMMs) are commonly used in model-based clustering (McLachlan & Peel, 2000; Fraley & Raftery, 2002). In finite mixture modeling, the population is assumed to be made up from a finite number of clusters. Suppose a variable Y is used to indicate this cluster, and variables $\boldsymbol{X}$ represent the attributes in the data. The variable Y is referred to as a *latent* (or unobserved) variable, and the variables $\boldsymbol{X}$ as *manifest* (or observed) variables. The manifest variables $\boldsymbol{X}$ is assumed to follow a mixture distribution

$$P(\boldsymbol{x}) = \sum_y P(y)P(\boldsymbol{x}|y).$$

The probability values of the distribution $P(y)$ are known as *mixing proportions* and the conditional distributions $P(\boldsymbol{x}|y)$ are known as *component distributions*. To generate a sample, the model first picks a cluster y according to the distribution $P(y)$ and then uses the corresponding component distribution $P(\boldsymbol{x}|y)$ to generate values for the observed variables.

Gaussian distributions are often used as the component distributions due to computational convenience. A Gaussian mixture model (GMM) has a distribution given by

$$P(\boldsymbol{x}) = \sum_y P(y)\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y),$$

where $\mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ is a multivariate Gaussian distribution, with mean vector $\boldsymbol{\mu}_y$ and covariance matrix $\boldsymbol{\Sigma}_y$ conditional on the value of Y.

The Expectation-Maximization (EM) algorithm (Dempster, Laird, & Rubin, 1977) can be used to fit the model. Once the model is fit, the probability that a data point $\boldsymbol{d}$ belongs to cluster y can be computed by

$$P(y|\boldsymbol{d}) \propto P(y)\mathcal{N}(\boldsymbol{d}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y),$$

where the symbol $\propto$ means that the exact values of the distribution $P(y|\boldsymbol{d})$ can be obtained by using the sum $\sum_y P(y)\mathcal{N}(\boldsymbol{d}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ as a normalization constant. The data point $\boldsymbol{d}$ is assigned to each cluster y by a different degree of association $P(y|\boldsymbol{d})$. This is known as *soft assignment*. The latent variable Y hence represents a (soft) *partition* of data. We can see that since GMM has only one latent variable, it is limited to single clustering.

The number of clusters can be given or determined by model selection. In the latter case, a score is used to evaluate a model with G number of clusters. The G that leads to the highest score is then chosen as the optimal number of clusters. Many scores have been proposed, and the BIC score has been empirically shown to perform well among them (Fonseca & Cardoso, 2007).

3. Related Work

Multiple clusterings can be considered as facilitating variable selection by simultaneously doing different variable selection with each clustering. Therefore, we review some work on variable selection in this section, with a focus on model-based methods. After that, we review some work on multiple clusterings. Since there are few model-based methods, we include those distance-based methods also. At the end of this section, we briefly discuss some other related work.

3.1 Variable Selection

Several variable selection methods have been proposed for model-based clustering. Most of them introduce flexibility into the generative mixture model to allow clusters to be related to subsets of (instead of all) attributes and determine the subsets alongside parameter estimation or during a separate model selection phase. Raftery and Dean (2006) consider a variation of the GMM where the latent variable is related to a subset of attributes and is independent of other attributes given the subset. A greedy algorithm is proposed to search among those models for one with high BIC score. At each step of search, two nested models are compared using the Bayes factor and the better one is chosen to seed the next step of search. Maugis, Celeux, and Martin-Magniette (2009a, 2009b) extend this work by considering different dependency possibilities among the relevant and irrelevant attributes.

Law, Figueiredo, and Jain (2004) start with the Naïve Bayes model (i.e., GMM with diagonal covariance matrices) and add a saliency parameter for each attribute. The parameter ranges between 0 and 1. When it is 1, the attribute depends on the latent variable. When it is 0, the attribute is independent of the latent variable and its distribution is assumed to be unimodal. The saliency parameters are estimated together with other model parameters using the EM algorithm. The work is extended by Li, Dong, and Hua (2009) so that the saliency of an attribute can vary across clusters.

The third line of work is based on GMMs where all clusters share a common diagonal covariance matrix, while their means may vary. If the mean of a cluster along an attribute turns out to coincide with the overall mean, then that attribute is irrelevant to cluster. Both Bayesian methods (Liu, Zhang, Palumbo, & Lawrence, 2003; Hoff, 2006) and regularization methods (Pan & Shen, 2007) have been developed based on this idea.

3.2 Multiple Clusterings

The above variable selection methods make an assumption that there is only one single way to partition data. This assumption becomes invalid on multifaceted data. In contrast, some methods aim to produce multiple clusterings. Zhang (2002, 2004) proposes one of the earliest methods for multiple clusterings. He proposes a model-based method using a tree-structured Bayesian network called hierarchical latent class model. The model contains multiple discrete latent variables, which the author considers as an advantage since they allow multiple ways of clustering. However, the learning process for this model is slow and the paper considers data with only fewer than a dozen attributes. Moreover, these models can handle only discrete data.

Galimberti and Soffritti (2006) propose another model-based method for multiple clusterings. This method partitions the attributes and learns a GMM on each attribute subset. This collection of independent GMMs forms the resulting model. To look for the optimal partition of attributes, the method starts with each attribute in a separate subset. It repeatedly merges the two subsets of attributes that lead to the largest improvement of BIC of the resulting model until it cannot find any improvement. The latent variables in the resulting model are independent to each other. They are different from those latent variables in a PLTM, which are connected by a tree structure. Galimberti and Soffritti (2007) suggest to speed up this work by using variable clustering to find the initial attribute partition.

The other existing multiple-clustering methods are mostly distance-based. They aim to find multiple clusterings that are dissimilar from each other. Two approaches are mainly used to find those dissimilar clusterings. The first approach finds them sequentially. It finds an alternative clustering based on a given clustering. This alternative clustering, which is dissimilar to the given one, can be found based on conditional information bottleneck (Gondek & Hofmann, 2004), by "cannot-links" constraints (Bae & Bailey, 2006), by orthogonal projection (Cui, Fern, & Dy, 2007), or based on an optimization problem constrained by the similarity between clusterings (Qi & Davidson, 2009).

The second approach finds multiple clusterings simultaneously. Jain, Meka, and Dhillon (2008) propose a method called decorrelated k-means, which simultaneously finds two clusterings based on k-means. It iteratively minimizes the sum-squared errors of each clustering along with the correlation between the two clusterings. Niu, Dy, and Jordan (2010) propose a method that simultaneously finds two clusterings based on spectral clustering. It adds to its objective function a term that penalizes the the similarity between the two clusterings based on the Hilbert-Schmidt independence criterion. Dasgupta and Ng (2010) propose a method that produces multiple clusterings, each with 2 clusters, using the suboptimal solutions of spectral clustering.

There are three main differences between these distance-based methods and PLTM. First, these methods aim to find dissimilar clusterings while PLTM models the probabilistic dependencies between clusterings. Second, these distance-based methods do not consider model selection. They either have fixed number of clusterings or require the knowledge of it. Also, they need to specify the number of clusters in each of these clusterings. Third, they often need to specify the tradeoff between the quality of clusterings and the dissimilarity between clusterings. Using a wrong parameter may significantly affect their performance.

3.3 Other related work

Caruana, Elhawary, Nguyen, and Smith (2006) generate numerous clusterings by applying random weights on the attributes. The clusterings are presented in an organized way so that a user can pick the one he deems the best. Subspace clustering (e.g., Kriegel, Kröger, & Zimek, 2009) tries to identify all clusters in all subspaces. A data point may belong to more than one clusters. Biclustering (e.g., Madeira & Oliveria, 2004) attempts to find groups of objects that exhibit the same pattern (e.g., synchronous rise and fall) over a subset set of attributes. Unlike our approach, these methods are distance-based rather than model-based.

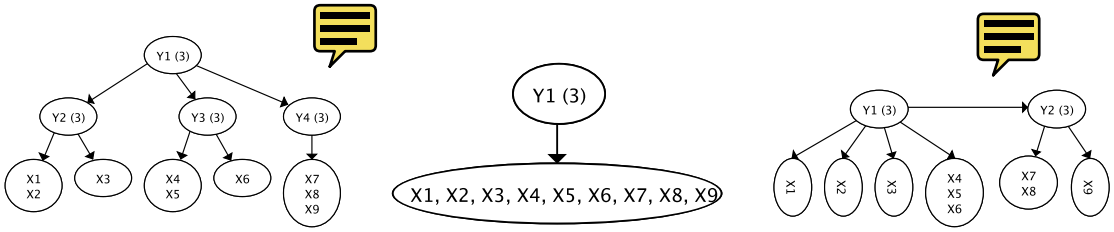


Figure 1: (a) An example of PLTM. The numbers in parentheses show the cardinalities of the discrete variables. (b) GMM as a special case of PLTM. (c) Generative model for synthetic data.

4. Pouch Latent Tree Models

A *pouch latent tree model* (PLTM) is a rooted tree, where each internal node represents a latent variable, and each leaf node represents a set of manifest variables. All the latent variables are discrete, while all the manifest variables are continuous. A leaf node may contain a single manifest variable or several of them. Because of the second possibility, leaf nodes are called *pouch nodes*. Figure 1a shows an example of PLTM. In this example, Y_1–Y_4 are discrete latent variables, each having three possible values. X_1–X_9 are continuous manifest variables, which are grouped into five pouch nodes, $\{X_1, X_2\}$, $\{X_3\}$, $\{X_4, X_5\}$, $\{X_6\}$, and $\{X_7, X_8, X_9\}$.

In this paper, we use capital letters such as X and Y to denote random variables, lower case letters such as x and y to denote their values, and bold face letters such as $\boldsymbol{X}$, $\boldsymbol{Y}$, $\boldsymbol{x}$, and $\boldsymbol{y}$ to denote sets of variables or values. When the meaning is clear from context, we use the terms 'variable' and 'node' interchangeably. We use $\pi(V)$ to indicate the parent variable of a variable V, and $\pi(v)$ to denote its value. We denote the set of continuous variables of a pouch node by $\boldsymbol{X}_i$, where $i \in \mathcal{I}$ is an index of the pouch node and $\mathcal{I}$ is the set of indices of pouch nodes. Note that $\boldsymbol{X}_i$ is different from $\boldsymbol{X}$, which denotes the set of all manifest variables, and $\boldsymbol{X}_i \subseteq \boldsymbol{X}$. Note also that $|\mathcal{I}| \leq |\boldsymbol{X}|$, since a pouch node may contain more than one variable.

In a PLTM, the dependency of a discrete latent variable Y on its parent $Y' = \pi(Y)$ is characterized by a conditional discrete distribution $P(y|y')$.[1] Let $\boldsymbol{X}_i$ be the variables in a pouch node with parent node $Y = \pi(\boldsymbol{X}_i)$. We assume that, given a value y of Y, $\boldsymbol{X}_i$ follow the conditional Gaussian distribution $P(\boldsymbol{x}_i|y) = \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ with mean vector $\boldsymbol{\mu}_y$ and covariance matrix $\boldsymbol{\Sigma}_y$. We write a PLTM as a pair $M = (m, \boldsymbol{\theta})$, where m denotes the structure of the model and $\boldsymbol{\theta}$ denotes the parameters.

Example 1. *Consider a PLTM $M = (m, \boldsymbol{\theta})$. The model structure m is shown in Figure 1c. There are two latent variables Y_1 and Y_2 and each of them have three values $\{s_1, s_2, s_3\}$. There are six pouch nodes representing continuous variables, namely $\{X_1\}$, $\{X_2\}$, $\{X_3\}$, $\{X_4, X_5, X_6\}$, $\{X_7, X_8\}$, and $\{X_9\}$.*

1. The root node is regarded as the child of a dummy node with only one value, and hence is treated in the same way as other latent nodes.

The parameters $\boldsymbol{\theta}$ consists of the parameters that specify the following distributions. Each node is associated with a conditional distribution given the value of its parent variable. The distribution of the root node Y_1, which has no parent, is given by $P(y_1) = 1/3$, where $y_1 \in \{s_1, s_2, s_3\}$. The conditional distribution of Y_2 is given by

$$P(y_2|y_1) = \begin{cases} 0.6 & : \quad y_2 = y_1 \\ 0.2 & : \quad y_2 \neq y_1 \end{cases}, \text{ where } y_1, y_2 \in \{s_1, s_2, s_3\}.$$

The conditional distribution of each pouch node follows a Gaussian distribution and has parameters $\boldsymbol{\mu}_{\pi(\boldsymbol{x}_i)}$ and $\boldsymbol{\Sigma}_{\pi(\boldsymbol{x}_i)}$. The conditional means $\mu_{\pi(\boldsymbol{x}_i)}$ of each singular variable X_i are -2.5, 0, and 2.5, when its parent variable $\pi(X_i)$ has values of s_1, s_2, and s_3, respectively. The variance of each singular variable X_i is 1, and the covariance between any pair of these variables in the same pouch is 0.5. In other words, $\boldsymbol{\Sigma}_{\pi(\boldsymbol{x}_i)}$ is a matrix with entries of 1 on the diagonal and 0.5 otherwise. This completes the specification of a PLTM. □

PLTMs have an interesting two-way relationship with GMMs. On the one hand, PLTMs generalize the structure of GMMs and allow more than one latent variables. Thus, a GMM can be considered as a PLTM with only one latent variable and one pouch node containing all manifest variables. As an example, a GMM is depicted as a PLTM in Figure 1b, in which Y_1 is the latent variable and X_1–X_9 are the continuous manifest variables.

On the other hand, the distribution of a PLTM over the manifest variables can be represented by a GMM. Suppose $\boldsymbol{X}$ and $\boldsymbol{Y}$ are the sets of manifest variables and latent variables in a PTLM M, $\boldsymbol{X}_i$ are the continuous variables of a pouch node, and $Y_j \in \boldsymbol{Y}$ is the discrete variable of a latent node. Let $\mathcal{I}$ be the set of indices of all pouch nodes, and $\mathcal{J}$ be the set of indices of all latent nodes. The probability distribution defined by M over the manifest variables $\boldsymbol{X}$ is given by

$$\begin{aligned} P(\boldsymbol{x}) &= \sum_{\boldsymbol{y}} P(\boldsymbol{x}, \boldsymbol{y}) \\ &= \sum_{\boldsymbol{y}} \prod_{i \in \mathcal{I}} \mathcal{N}(\boldsymbol{x}_i | \boldsymbol{\mu}_{\pi(\boldsymbol{x}_i)}, \boldsymbol{\Sigma}_{\pi(\boldsymbol{x}_i)}) \prod_{j \in \mathcal{J}} P(y_j | \pi(y_j)) \qquad (1) \\ &= \sum_{\boldsymbol{y}} P(\boldsymbol{y}) \mathcal{N}(\boldsymbol{x} | \mu_{\boldsymbol{y}}, \Sigma_{\boldsymbol{y}}), \qquad (2) \end{aligned}$$

where Equation 1 comes from the definition of the model, while Equation 2 follows from the fact that $\pi(\boldsymbol{x}_i), \pi(y_j) \in \boldsymbol{Y}$ and the multiplication of a Gaussian distribution is also a Gaussian distribution. Equation 2 shows that $P(\boldsymbol{x})$ has a mixture of Gaussian distribution. Although this shows that PLTMs are not more expressive than GMMs on the observed data, we prefer PLTMs over GMMs for two reasons. First, parameters can be reduced in PLTM by exploiting the conditional independence between variables, as expressed in the factorizations in Equation 1. This parsimonious representation by PLTM allows a larger number of components in the equivalent mixture distributions before over-fitting. Second, the multiple latent variables in PLTM allow multiple clusterings on data.

In addition to the relationship with GMM, PLTMs also resemble the hierarchical latent class models (Zhang, 2004). However, the latter model consists of only discrete variables and do not have pouch nodes. Besides, PLTM is different from a Bayesian network (BN)

only by the existence of pouch nodes. Since any nonsingular multivariate Gaussian distribution can be converted to a complete Gaussian Bayesian network (GBN) with equivalent distribution (Geiger & Heckerman, 1994), a pouch node can be considered as a shorthand notation of a complete GBN. By converting each pouch node into a complete GBN, a PLTM becomes a conditional Gaussian Bayesian network (i.e., a BN with discrete distributions and conditional Gaussian distributions). Let $X' \in \boldsymbol{X}_i$ be a node in the complete GBN converted from a pouch node $\boldsymbol{X}_i$. In the converted model, if $Y = \pi(\boldsymbol{X}_i)$ is a parent node of $\boldsymbol{X}_i$ in the original PLTM, it is also one of the parent nodes of node X', meaning that $Y \in \pi(X')$, $\forall X' \in \boldsymbol{X}$. Consequently, a PLTM can be considered to be a BN.

Cluster analysis based on PLTM requires learning a PLTM from data. It involves parameter estimation and structure learning. We discuss the former problem in Section 6, and the two problems as a whole in Section 7. But before that, we first discuss the inference problem in the next section, since it is required by parameter estimation.

5. Inference

PLTM defines a probability distribution $P(\boldsymbol{x}, \boldsymbol{y})$ over manifest variables $\boldsymbol{X}$ and latent variables $\boldsymbol{Y}$. Consider observing values $\boldsymbol{e}$ for the evidence variables $\boldsymbol{E} \subseteq \boldsymbol{X}$. We are often required to compute the posterior probability $P(\boldsymbol{q}|\boldsymbol{e})$ over a subset of variables $\boldsymbol{Q} \subseteq \boldsymbol{X} \cup \boldsymbol{Y}$, such as for parameter estimation or assignment of data to clusters. This computation can be done based on the clique tree propagation for conditional Gaussian Bayesian networks by Lauritzen and Jensen (2001). However, due to the existence of pouch nodes in PLTM, this propagation algorithm requires some minor adaptions. We focus on the details related to the pouch nodes in this section, and refer the readers to other source for background on the clique tree propagation algorithm (see Cowell, Dawid, Lauritzen, & Spiegelhalter, 1999).

The propagation algorithm involves four steps: construction of clique tree, initialization of clique tree, incorporation of evidence, and message passing. A clique tree can be constructed simply from PLTM due to its tree structure. Each clique in the clique tree corresponds to an edge in the PLTM, and contains the variables of a node $\boldsymbol{V}$ and variable of its parent node $\pi(\boldsymbol{V})$. The resulting clique tree thus contains two types of cliques. A discrete clique contains two discrete variables, and a mixed clique contains one discrete variable and the continuous variables of a pouch node.

During initialization, the potential of each clique is first initialized to $\psi(\boldsymbol{v}, \pi(\boldsymbol{v})) = 1$. The conditional distribution $P(\boldsymbol{v}|\pi(\boldsymbol{v}))$ of each node $\boldsymbol{V}$ of the PLTM is then multiplied to the potential of any one of the cliques that contains both $\boldsymbol{V}$ and $\pi(\boldsymbol{V})$. There exists only one such clique except for the distribution of the root node. After initialization, the potential of a mixed clique with continuous variables $\boldsymbol{X}_i$ and discrete variable $Y = \pi(\boldsymbol{X}_i)$ becomes

$$\psi(\boldsymbol{x}_i, y) = P(\boldsymbol{x}_i|y) = \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y).^2$$

2. If a mixed clique contains the variable of the root node Y, its potential may be multiplied by $P(y)$ during initialization and becomes $\psi(\boldsymbol{x}_i, y) = P(y)P(\boldsymbol{x}_i|y))$ after initialization. This potential is handled similarly in the following, except with an extra factor $P(y)$. We omit the treatment of this special case for brevity.

The next step is the incorporation of evidence. Denote $\boldsymbol{X}'_i = \boldsymbol{X}_i \backslash \boldsymbol{E}_i$, $\boldsymbol{\mu}_{\{\boldsymbol{S}\}}$ as the part of mean vector $\boldsymbol{\mu}$ containing elements corresponding to variables $\boldsymbol{S}$, and $\boldsymbol{\Sigma}_{\{\boldsymbol{ST}\}}$ as the part of covariance matrix $\boldsymbol{\Sigma}$ that has rows and columns corresponding to variables $\boldsymbol{S}$ and $\boldsymbol{T}$, respectively. To incorporate the evidence $\boldsymbol{e}_i$ for $\boldsymbol{E}_i \subseteq \boldsymbol{X}_i$, the potential of a mixed clique changes from $\psi(\boldsymbol{x}_i, y)$ to

$$\psi'(\boldsymbol{x}_i, y) = P(\boldsymbol{e}_i|y) \times P(\boldsymbol{x}_i|y, \boldsymbol{e}_i) = \mathcal{N}(\boldsymbol{e}_i|\boldsymbol{\mu}_{y,\{\boldsymbol{E}_i\}}, \boldsymbol{\Sigma}_{y,\{\boldsymbol{E}_i\boldsymbol{E}_i\}}) \times \mathcal{N}(\boldsymbol{x}_i|\boldsymbol{\mu}'_y, \boldsymbol{\Sigma}'_y),$$

where $\boldsymbol{\mu}'_y$ and $\boldsymbol{\Sigma}'_y$ are given by

$$\boldsymbol{\mu}'_{y,\{\boldsymbol{X}'_i\}} = \boldsymbol{\mu}_{y,\{\boldsymbol{X}'_i\}} + \Sigma_{y,\{\boldsymbol{X}'_i\boldsymbol{E}_i\}}\Sigma^{-1}_{y,\{\boldsymbol{E}_i\boldsymbol{E}_i\}}\left[\boldsymbol{e}_i - \mu_{y,\{\boldsymbol{E}_i\}}\right],$$

$$\Sigma'_{y_i,\{\boldsymbol{X}'_i\boldsymbol{X}'_i\}} = \Sigma_{y,\{\boldsymbol{X}'_i\boldsymbol{X}'_i\}} - \Sigma_{y,\{\boldsymbol{X}'_i\boldsymbol{E}_i\}}\Sigma^{-1}_{y,\{\boldsymbol{E}_i\boldsymbol{E}_i\}}\Sigma_{y,\{\boldsymbol{E}_i\boldsymbol{X}'_i\}},$$

$$\mu'_{y,\{\boldsymbol{E}_i\}} = \boldsymbol{e}_i, \Sigma'_{y,\{\boldsymbol{X}'_i\boldsymbol{E}'_i\}} = \Sigma'_{y,\{\boldsymbol{E}'_i\boldsymbol{X}'_i\}} = \Sigma'_{y,\{\boldsymbol{X}'_i\boldsymbol{X}'_i\}} = \boldsymbol{0}.$$

In the message passing step, two operations, marginalization and combination, are required to send messages from and receive messages at the mixed cliques, respectively. The marginalization of $\psi'(\boldsymbol{x}_i, y)$ over $\boldsymbol{X}_i$ is given by a potential involving only the discrete variable y, as follows:

$$\psi'(y) = P(\boldsymbol{e}_i|y) = \mathcal{N}(\boldsymbol{e}_i|\mu_{y,\{\boldsymbol{E}_i\}}, \Sigma_{y,\{\boldsymbol{E}_i\boldsymbol{E}_i\}}).$$

The combination of the potential $\psi'(\boldsymbol{x}_i, y)$ with a discrete potential $\phi(y)$ is given by $\psi''(\boldsymbol{x}, y) = \psi'(\boldsymbol{x}_i, y) \times \phi(y)$. When the message passing completes with normalization, the potential holds the distribution

$$\psi''(\boldsymbol{x}_i, y) = P(y|\boldsymbol{e}) \times P(\boldsymbol{x}_i|y, \boldsymbol{e}) = P(\boldsymbol{x}_i|y, \boldsymbol{e}),$$

from which posterior probability $P(y|\boldsymbol{e})$ can also be obtained. The posterior probability $P(y, y'|\boldsymbol{e})$ of a discrete variable Y and its parent Y' can be obtained from the potential $\psi''(y, y')$ of the discrete clique that contains these variables.

Let n be the number of nodes in a PLTM, c be the maximum cardinality of a discrete variable, and p be the maximum number of variables in a pouch node. The time complexity of the inference is dominated by the steps related to message passing and incorporation of evidence on continuous variables. The former step requires $O(nc^2)$, since each clique contains at most two discrete variables. The latter steps requires $O(ncp^3)$. The last term shows that inference becomes faster when a large pouch is split into smaller pouches. This indicates that inference on PLTM can sometimes be more efficient than GMM even though PLTM has a more complex model structure.

6. Parameter Estimation

Suppose there is a data set $\mathcal{D}$ with N samples $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N$. Each sample consists of values for the manifest variables. Consider computing the maximum likelihood estimate (MLE) $\boldsymbol{\theta}^*$ of the parameters for a given PLTM structure m. We do this using the EM algorithm. The algorithm starts with an initial estimate $\boldsymbol{\theta}^{(0)}$ and improves the estimate iteratively.

Suppose the parameter estimate $\boldsymbol{\theta}^{(t-1)}$ is obtained after $t-1$ iterations. The t-th iteration consists of two steps, an E-step and an M-step. In the E-step, we compute, for each latent node Y and its parent Y', the distributions $P(y, y'|\boldsymbol{d}_i, \boldsymbol{\theta}^{(t-1)})$ and $P(y|\boldsymbol{d}_i, \boldsymbol{\theta}^{(t-1)})$. This is done by the inference algorithm discussed in the previous section. For each k, let $\boldsymbol{x}_{i,k}$ be the values of variables $\boldsymbol{X}_i$ of a pouch node for the sample $\boldsymbol{d}_k$. In the M-step, the new estimate $\boldsymbol{\theta}^{(t)}$ is obtained as follows:

$$P(y|y', \boldsymbol{\theta}^{(t)}) \propto \sum_{k=1}^{N} P(y, y'|\boldsymbol{d}_k, \boldsymbol{\theta}^{(t-1)})$$

$$\boldsymbol{\mu}_y^{(t)} = \frac{\sum_{k=1}^{N} P(y|\boldsymbol{d}_k, \boldsymbol{\theta}^{(t-1)}) \boldsymbol{x}_{i,k}}{\sum_{k=1}^{N} P(y|\boldsymbol{d}_k, \boldsymbol{\theta}^{(t-1)})}$$

$$\boldsymbol{\Sigma}_y^{(t)} = \frac{\sum_{k=1}^{N} P(y|\boldsymbol{d}_k, \boldsymbol{\theta}^{(t-1)})(\boldsymbol{x}_{i,k} - \boldsymbol{\mu}_y^{(t)})(\boldsymbol{x}_{i,k} - \boldsymbol{\mu}_y^{(t)})'}{\sum_{k=1}^{N} P(y|\boldsymbol{d}_k, \boldsymbol{\theta}^{(t-1)})},$$

where $\boldsymbol{\mu}_y^{(t)}$ and $\boldsymbol{\Sigma}_y^{(t)}$ here correspond to the conditional distribution $P(\boldsymbol{x}_i|y)$ for each node $\boldsymbol{X}_i$ and $Y = \pi(\boldsymbol{X}_i)$. The EM algorithm proceeds to the $(t+1)$-th iteration unless the improvement of log-likelihood $\log P(\mathcal{D}|\boldsymbol{\theta}^{(t)}) - \log P(\mathcal{D}|\boldsymbol{\theta}^{(t-1)})$ falls below a certain threshold.

The starting values of the parameters $\boldsymbol{\theta}^{(0)}$ are chosen as follows. For $P(y|y', \boldsymbol{\theta}^{(0)})$, the probabilities are randomly generated from a uniform distribution over the interval $(0, 1]$ and are then normalized. Let $\boldsymbol{\mu}_{\mathcal{D},\{\boldsymbol{X}_i\}}$ and $\boldsymbol{\Sigma}_{\mathcal{D},\{\boldsymbol{X}_i\}}$ be the sample mean and covariance, respectively, of $\mathcal{D}$ projected on variables $\boldsymbol{X}_i$ of a pouch node. To initialize the conditional Gaussian distribution of $\boldsymbol{X}_i$, the means $\boldsymbol{\mu}_y^{(0)}$ are generated from $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{D},\{\boldsymbol{X}_i\}}, \boldsymbol{\Sigma}_{\mathcal{D},\{\boldsymbol{X}_i\}})$, and the covariances $\boldsymbol{\Sigma}_y^{(0)}$ are set to $\boldsymbol{\Sigma}_{\mathcal{D},\{\boldsymbol{X}_i\}}$.

Like in the case of GMM, the likelihood is unbounded in the case of PLTM. This might lead to spurious local maxima (McLachlan & Peel, 2000). We mitigate this problem using a variant of the method by Ingrassia (2004). In the M-step of EM, we need to compute the covariance matrix $\boldsymbol{\Sigma}_y^{(t)}$ for each pouch node $\boldsymbol{X}_i$. We impose the following constraints on the eigenvalues $\lambda^{(t)}$ of the covariance matrix: $\sigma_{min}^2/\gamma \leq \lambda^{(t)} \leq \gamma \times \sigma_{max}^2$, where σ_{min}^2 and σ_{max}^2 are the minimum and maximum of the sample variances of the variables $\boldsymbol{X}_i$ on $\mathcal{D}$ (i.e. the diagonal values of the matrix $\boldsymbol{\Sigma}_{\mathcal{D},\{\boldsymbol{X}_i\}}$) and γ is a parameter for our method.

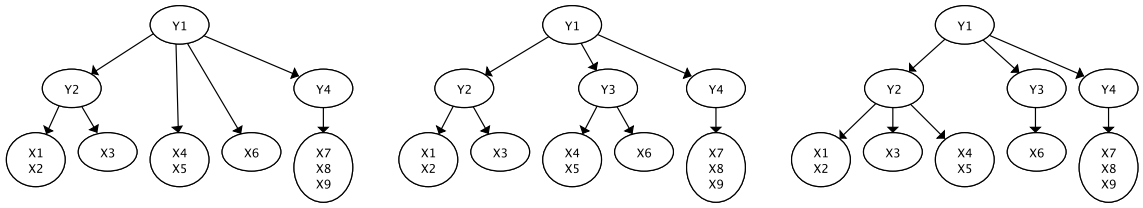
7. Structure Learning

Given a data set $\mathcal{D}$, we aim at finding the model m^* that maximizes the BIC score (Schwarz, 1978):

$$BIC(m|\mathcal{D}) = \log P(\mathcal{D}|m, \boldsymbol{\theta}^*) - \frac{d(m)}{2} \log N,$$

where $\boldsymbol{\theta}^*$ is the MLE of the parameters and $d(m)$ is the number of independent parameters in m. The first term is known as the likelihood term. It favors models that fit data well. The second term is known as the penalty term. It discourages complex models. Hence, the BIC score provides a trade-off between model fit and model parsimoniousness.

We have developed a hill-climbing algorithm to search for m^*. It starts with a model $m^{(0)}$ that contains one latent node as root and a separate pouch node for each manifest



(a) m_1 (b) m_2 (c) m_3

Figure 2: Examples of applying the node introduction, node deletion, and node relocation operators. Introducing Y_3 to mediate between Y_1, $\{X_4, X_5\}$ and $\{X_6\}$ in m_1 gives m_2. Relocating $\{X_4, X_5\}$ from Y_3 to Y_2 in m_2 gives m_3. In reverse, relocating $\{X_4, X_5\}$ from Y_2 to Y_3 in m_3 gives m_2. Deleting Y_3 in m_2 gives m_1.

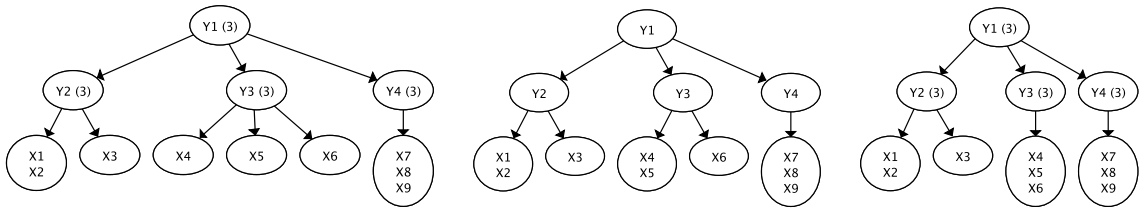
variable as a leaf node. The latent variable at the root node has two possible values. Suppose a model $m^{(j-1)}$ is obtained after $j-1$ iterations. In the j-th iteration, the algorithm uses some search operators to generate candidate models by modifying the base model $m^{(j-1)}$. The BIC score is then computed for each candidate model. The candidate model m' with the highest BIC score is compared with the base model $m^{(j-1)}$. If m' has a higher BIC score than $m^{(j-1)}$, m' is used as the new base model $m^{(j)}$ and the algorithm proceeds to the $(j+1)$-th iteration. Otherwise, the algorithm terminates and returns $m^* = m^{(j-1)}$ (together with the MLE of the parameters).

7.1 Search Operators

There are four aspects of the structure m, namely, the number of latent variables, the cardinalities of these latent variables, the connections between variables, and the composition of pouches. The search operators used in our hill-climbing algorithm modify all these aspects to effectively explore the search space. There are totally seven search operators, five of which are borrowed from Zhang and Kočka (2004), while two of them are new to PLTM.

A node introduction (NI) operator involves one latent node Y and two of its neighbors. It creates a new model by introducing a new latent node Y' to mediate between the latent variable and the two neighbors. The cardinality of Y' is set to be that of Y. A node deletion operator (ND) is the opposite of NI. It involves two neighboring latent nodes Y and Y'. It creates a new model by deleting Y' and making all neighbors of Y' other than Y neighbors of Y. For the sake of computational efficiency, we do not consider introducing a new node to mediate Y and more than two of its neighbors. This restriction will be compensated in search control. Given a latent variable in PLTM, a state introduction (SI) operator creates a new model by adding a state to the domain of the variable. A state deletion (SD) operator does the opposite. A node relocation (NR) involves a node V, one of its latent node neighbors Y and another latent node Y'. It creates a new model by relocating V to Y', i.e., removing the link between V and Y and adding a link between V and Y'. Figure 2 gives some examples of the use of NI, ND, and NR operators.

The two new operators are pouching (PO) and unpouching (UP) operators. The PO operator creates a candidate model by combining two sibling pouch nodes $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$ into one pouch node $\boldsymbol{X}' = \boldsymbol{X}_1 \cup \boldsymbol{X}_2$. The UP operator creates a candidate model by separating



(a) m_1 (b) m_2 (c) m_3

Figure 3: Examples of applying the pouching and unpouching operators. Pouching $\{X_4\}$ and $\{X_5\}$ in m_1 gives m_2, and pouching $\{X_4, X_5\}$ and $\{X_6\}$ in m_2 gives m_3. In reverse, unpouching X_6 from $\{X_4, X_5, X_6\}$ in m_3 gives m_2, and unpouching X_5 from $\{X_4, X_5\}$ gives m_1.

one variable X' from a pouch node $\boldsymbol{X}_0$, resulting in two sibling pouch nodes $\boldsymbol{X}_1 = \boldsymbol{X}_0 \backslash \{X'\}$ and $\boldsymbol{X}_2 = \{X'\}$. Figure 3 shows some examples of the use of these two operators. The design of UP has a drawback that to split the pouch node $\boldsymbol{X}_0 = \boldsymbol{X}_1 \cup \boldsymbol{X}_2$ into two pouch nodes $\boldsymbol{X}_1$ and $\boldsymbol{X}_2$, it cannot be done in a single UP operation. Instead, it requires a series of UP operations to separate each variable of $\boldsymbol{X}_2$ from $\boldsymbol{X}_0$, and then a series of PO operations to pouch these variables into a single pouch node $\boldsymbol{X}_2$. The advantage of the current design of UP is that it does not have to consider so many candidate models by limiting the number of the variables for separation to one. This improves the efficiency of the UP operator.

The hill-climbing algorithm as outlined above is very inefficient and can handle only toy data sets. We have developed acceleration techniques that make the algorithm efficient enough for some real-world applications. These acceleration techniques, which are borrowed from Chen, Zhang, and Wang (2008), are described in the following two subsections. For brevity, we focus on how these techniques improve the efficiency of the algorithm, rather than give the tedious details for reproducing the algorithm exactly in this paper. The details of the algorithm will be given in the thesis of the first author.

7.2 EAST

In every search step, a search operator generates all possible candidate models for consideration. Let l and b be the numbers of latent nodes and pouch nodes, respectively, such that the total number of nodes is $n = l + b$. Moreover, let p, q, and r be the maximum number of variables in a pouch node, the maximum number of sibling pouch nodes, and the maximum number of neighbors of a latent node, respectively. Note that $q, r \leq b$ and $b, p \leq |\boldsymbol{X}|$. The numbers of candidate models that NI, ND, SI, SD, NR, PO, and UP generate are $O(lr(r-1)/2)$, $O(lr)$, $O(l)$, $O(l)$, $O(nl)$, $O(lq(q-1)/2)$, and $O(bp)$, respectively. If we consider all seven operators in each search step, many candidate models are generated but only one of them is chosen for the next step. In some sense, the time spent on estimating the suboptimal models are wasted, since these models are discarded and are not used for the next step.

A more efficient way is to consider fewer candidate models in each search step. This can be achieved by considering only a subset of search operators at a time. The search operators can be naturally partitioned into three categories. SI, NI, and PO belong to the expansion

category, since all of them create candidate models that are more complex than the current one. SD, ND, and UP, which simplify a model, belong to the simplification category. NR does not change the complexity considerably. It belongs to the adjustment category. We therefore perform search in three phases, each of which considers only operators of one category. The best model found in each phase is used to seed the next phase, and the search repeats these three phases until all these phases cannot find a better model. This search is called EAST, since it does Expansion, Adjustment, and Simplification until Termination.

7.3 Efficient Model Evaluation

Consider that m is the current model after a number of search steps. A candidate model m' can be obtained from m by applying a search operator. Very often, the search operator modifies only a small part of m, so that m and m' share a large number of parameters. For example, consider that m_2 is obtained from m_1 using the PO operator, as shown in Figure 3. m_1 and m_2 shares many parameters such as $P(x_1, x_2|y_2)$, $P(y_2|y_1)$, $P(y_3|y_1)$, and $P(x_6|y_3)$, etc. On the other hand, some parameters are not shared by m and m'. In this example, parameters $P(x_4|y_3)$ and $P(x_5|y_3)$ are specific to m_1, while parameter $P(x_4, x_5|y_3)$ is specific to m_2. We can divide the parameters of m' into two groups and write the parameters $\boldsymbol{\theta}'$ as $(\boldsymbol{\theta}'_1, \boldsymbol{\theta}'_2)$. $\boldsymbol{\theta}'_1$ consists of parameters shared with m, while $\boldsymbol{\theta}'_2$ consists of parameters specific to m'. Similarly, we write the parameters of m as $\boldsymbol{\theta} = (\boldsymbol{\theta}_1, \boldsymbol{\theta}_2)$.

Suppose we have computed the MLE $\boldsymbol{\theta}^* = (\boldsymbol{\theta}^*_1, \boldsymbol{\theta}^*_2)$ of the parameters of m. $\boldsymbol{\theta}^*_1$ can be used as estimates for the shared parameters of m'. Consequently, we can obtain a likelihood function $P(\mathcal{D}|m', \boldsymbol{\theta}^*_1, \boldsymbol{\theta}'_2)$ that depends on only the unshared parameters $\boldsymbol{\theta}'_2$. This function is referred to as *restricted likelihood function* of m'. As previously described, we use the BIC score to evaluate a candidate model. This requires the maximum log-likelihood (ML) of m'. Instead of computing the ML over all parameters of m', we may approximate it by maximizing the restricted log-likelihood over only the subset of parameters $\boldsymbol{\theta}'_2$. This results in an approximate score given by

$$BIC_{RL}(m'|\mathcal{D}) = \max_{\boldsymbol{\theta}'_2} \log P(\mathcal{D}|m', \boldsymbol{\theta}^*_1, \boldsymbol{\theta}'_2) - \frac{d(m')}{2} \log N.$$

The advantage of using BIC_{RL} is that it allows a more efficient implementation of EM. This is due to two reasons. First, it involves a maximization of fewer parameters in the M-step. This also means fewer computations in the E-step, since only distributions relevant to $\boldsymbol{\theta}'_2$ need to be computed. Second, the E-step can exploit the fixed parameters $\boldsymbol{\theta}^*_1$ to allow sharing of computation among all iterations of EM. Specifically, the E-step requires inference on PLTM, which in turn requires passing messages in the clique tree. As the parameters $\boldsymbol{\theta}^*_1$ do not change in any iterations, the messages that depend on only these parameters remain the same in all iterations. Therefore, some messages can be cached for each data case and can be shared for the inference used in subsequent E-steps.

In our implementation, both BIC and BIC_{RL} are used to evaluate models, but at different situations. The approximation BIC_{RL} is used for quickly evaluating candidate models generated by the same search operator, while the real BIC is used for accurately evaluating the candidate model with the highest BIC_{RL} for each of the search operators. This improves the speed for the evaluation of models within search operators, but maintains the accuracy for the evaluation of models across search operators.

8. Empirical Results

Our empirical study is divided into two parts, the first part on labeled data and the second on unlabeled data. It is presented in two sections. The first part, which is presented in this section, is designed to compare PLTM analysis on labeled data with other related methods. These methods include single-clustering methods, with or without variable selection, and other multiple-clustering methods. We aim to show that PLTM analysis, as a multiple-clustering method that facilitates variable selection, is better than the other clustering methods that perform variable selection. Moreover, we aim to show that, among the multiple-clustering methods that we compare with, PLTM analysis performs best in general.

In the following two subsections, we describe the algorithms that we compare PLTM with and how we compare these algorithms. Afterwards, we discuss the experimental results on different data sets.

8.1 Algorithms for Comparison

Four single-clustering method based on GMMs were chosen for comparison. The first method is plain GMM analysis. The second one is MCLUST (Fraley & Raftery, 2002, 2006)[3], which reduces the number of parameters by imposing constraints on the eigenvalue decomposition of the covariance matrices. The third method is CLUSTVARSEL (Raftery & Dean, 2006)[4], which is denoted as *CVS* for short. The last one is the method of Law et al. (2004), which we call *LFJ*, using the the first letters of the three author names. Among these four single clustering methods, the last two perform variable selection while the first two do not. We are mainly concerned with the two variable selection methods, while the other two are included for reference.

The multiple-clustering methods we selected for comparison included the model-based method proposed by Galimberti and Soffritti (2006). We denote it as *MCS* (Multiple Clustering Structures) due to the title of the paper. Since there are few model-based multiple-clustering methods, we included two other distance-based methods for comparison. The first one is the orthogonal projection method (*OP*) by Cui et al. (2007), and the second one is decorrelated k-means (*DKM*) by Jain et al. (2008). These two methods represent the two different approaches to find alternative clusterings. OP finds them sequentially, while DKM finds them simultaneously. For OP, we followed the authors' experimental setting for finding the first clustering. We used k-means after reducing the dimensionality of data with PCA with at least 90% of original variance retained.

In our experiments, the parameters of GMMs and PLTMs were estimated using the EM algorithm. The same settings were used for both cases. EM was terminated when it failed to improve the log-likelihood by 0.01 in one iteration or when the number of iterations reached 500. We used a variant of multiple-restart approach (Chickering & Heckerman, 1997) with 64 starting points to avoid local maxima. For the scheme to avoid spurious local maxima as described in Section 6, we set the constant γ at 20. For GMM, since there is no standard method to determine the number of clusters, we gave the true numbers of classes as input. For MCLUST, CVS, LFJ, and MCS, the maximum number of mixture components was

3. http://cran.r-project.org/web/packages/mclust
4. http://cran.r-project.org/web/packages/clustvarsel

set at 20. For OP and DKM, the number of clusterings is set to 2 and the number of clusters in each clustering is given according to the true number of classes. DKM requires setting a parameter λ that weights the significance of the decorrelation between clusterings. The authors suggest to use $\lambda \in [100, 10000]$, but we found they did not lead to very good performance in our experiments. Therefore, we used another scheme to determine λ. We first ran the algorithm on one data set using different values of $\lambda \in [10^{-7}, 10^{7}]$, and then re-ran the algorithm using the λ that gave the best results in the first run. This should have given an advantage to DKM, since in real situations the class information is not available for finding an appropriate λ. The k-means used by OP and DKM used 10 random starts.

8.2 Method of Comparison

All experiments in this section started with labeled data. We removed the class labels in the training phrase, and then evaluated the clusterings based on the class labels in the testing phrase. The objective was to see which method recovered the class variable the best.

A model produced by a GMM-based method contains a single latent variable Y. This variable represents one way to partition the data. We follow Strehl and Ghosh (2002) and evaluate the partition using normalized mutual information $NMI(C;Y)$ between Y and the class variable C. The NMI ranges from 0 to 1, where a higher value means a closer match between C and Y. It is given by

$$NMI(C;Y) = \frac{I(C;Y)}{\sqrt{H(C)H(Y)}},$$

where $I(C;Y)$ is the mutual information between C and Y and $H(V)$ is the entropy of a variable V. These quantities can be computed from $P(C,Y)$, which in turn is estimated by $P(C,Y) = \frac{1}{N}\sum_{k=1}^{N} P(C|\boldsymbol{d}_k)P(Y|\boldsymbol{d}_k)$, where $\boldsymbol{d}_1, \ldots, \boldsymbol{d}_N$ are the data samples.

The multiple-clustering methods produce more than one partitions. The user may find several of these partitions interesting and use them all in his work. However, in this section, we are talking about comparing different clustering algorithms in terms of the ability to recover the original class partition. So, the user needs to choose one of the partitions as the final result. The question becomes whether these multiple-clustering methods provide the possibility for the user to recover the original class partitions. Consequently, we assume that the user chooses the partitions closest to the class partitions among all the partitions produced by these multiple-clusterings methods and we evaluate the performance of a multiple-clustering method against a class partition C as

$$\max_{Y \in \boldsymbol{Y}} NMI(C;Y),$$

where $\boldsymbol{Y}$ represents the set of partitions produced by a multiple-clustering method. To account for the randomness in the algorithms PLTM, GMM, LFJ, OP, and DKM, we report the averages and the standard deviations of NMI over 10 runs.

8.3 Synthetic Data with Two Clusterings

In the first experiment, we aim to investigate how different methods perform when the data is known to have two clusterings with varying degrees of dependency. We generated

	Y_2		
Y_1	s_1	s_2	s_3
s_1	$(1+2\kappa)/3$	$(1-\kappa)/3$	$(1-\kappa)/3$
s_2	$(1-\kappa)/3$	$(1+2\kappa)/3$	$(1-\kappa)/3$
s_3	$(1-\kappa)/3$	$(1-\kappa)/3$	$(1+2\kappa)/3$

Table 1: Conditional distribution $P(y_2|y_1)$ that was used to generate the synthetic data.

data from the model specified in Example 1 (shown in Figure 1c), except that we used the conditional distribution $P(y_2|y_1)$, which is parameterized by κ, as shown in Table 1 instead. The parameter κ controls the dependence between the two latent variables. The two latent variables are independent when $\kappa = 0$, while they are identical when $\kappa = 1$. Example 1 shows the values of $P(y_2|y_1)$ when $\kappa = 0.4$. The data were obtained by sampling 1,000 data cases from this model for each different κ. In this model, Y_1 is connected to more manifest variables. While Y_1 can therefore be considered as the dominant clustering on this data, we are interested in whether both clusterings Y_1 and Y_2 can be recovered.

The clustering performances of PLTM and other single-clustering methods are shown in Table 2a. We can see that PLTM was better than the other methods on this data. While PLTM could reliably recover both clusterings in data and thus outperformed the others, these other methods generally could not recover any one of the clusterings particularly well due to the influence of the other clustering.

For the variable selection methods, the attributes of one of the clusterings can be considered as noise to the other clustering. However, both variable selection methods failed to select the correct subset of attributes in most cases. LFJ estimated the saliency values of all attributes to be close to 1, meaning that it essentially selected all attributes for clustering. This method assumes that the noise comes from a single component of Gaussian distribution, but the noise in this data is a mixture of Gaussian distributions. This difference may explain the failure of LFJ to exclude the noise attributes. CVS successfully identified the subset of attributes of Y_1 when the two clusterings were independent ($\kappa = 0$), and therefore could recover the clustering Y_1 well. However, the clustering Y_2 was completely lost in this case. For other κ's, CVS selected attributes relevant to both clusterings, and hence did not perform well.

The results of these single-clustering methods indicate that they are not suitable for multifaceted data. They either fail to select attributes from only one clustering, or lose the information on the other possibly meaningful clusterings in data. Now we turn to the multiple-clustering methods. Their results are shown in Table 2b.

PLTM outperformed the other multiple clustering methods in general. The performances of MCS and DKM were close to that of PLTM when the two clusterings were relatively independent (e.g. $\kappa = 0, 0.2$) and these two methods could recover both clusterings well. However, the performances of both methods dropped when the clusterings in data became more dependent (e.g. $\kappa = 0.6, 0.8$), possibly because this dependency among clusterings violates the assumption of these methods that clusterings are independent. In fact, MCS partitioned the attributes properly into two subsets when $\kappa = 0, 0.2, 0.4$, but improperly into one subset when $\kappa = 0.6, 0.8$, meaning that the assumption independent

		Facilitate VS	Perform VS		No VS	
		PLTM	**CVS**	**LFJ**	**GMM**	**MCLUST**
$\kappa = 0$	Y_1	**.96 (.00)**	.95 (.00)	.61 (.01)	.61 (.24)	.68 (.00)
	Y_2	**.80 (.00)**	.00 (.00)	.51 (.01)	.16 (.14)	.56 (.00)
$\kappa = 0.2$	Y_1	**.97 (.00)**	.69 (.00)	.63 (.01)	.68 (.15)	.70 (.00)
	Y_2	**.80 (.00)**	.53 (.00)	.51 (.01)	.14 (.06)	.57 (.00)
$\kappa = 0.4$	Y_1	**.95 (.00)**	.68 (.00)	.63 (.01)	.64 (.21)	.71 (.00)
	Y_2	**.82 (.00)**	.57 (.00)	.55 (.01)	.33 (.13)	.61 (.00)
$\kappa = 0.6$	Y_1	**.97 (.00)**	.75 (.00)	.66 (.01)	.62 (.00)	.77 (.00)
	Y_2	**.85 (.00)**	.64 (.00)	.56 (.02)	.56 (.00)	.64 (.00)
$\kappa = 0.8$	Y_1	**.97 (.00)**	.83 (.00)	.68 (.02)	.78 (.00)	.89 (.00)
	Y_2	**.87 (.00)**	.70 (.00)	.58 (.03)	.71 (.00)	.70 (.00)

(a) PLTM and single-clustering methods

		Model-Based		Distance-Based	
		PLTM	**MCS**	**OP**	**DKM**
$\kappa = 0$	Y_1	**.96 (.00)**	**.96 (.00)**	.49 (.00)	.94 (.00)
	Y_2	**.80 (.00)**	**.80 (.00)**	.28 (.00)	.83 (.00)
$\kappa = 0.2$	Y_1	**.97 (.00)**	**.97 (.00)**	.55 (.00)	.94 (.01)
	Y_2	**.80 (.00)**	78 (.00)	.29 (.00)	.71 (.00)
$\kappa = 0.4$	Y_1	**.95 (.00)**	**.95 (.00)**	.72 (.00)	.76 (.03)
	Y_2	**.82 (.00)**	.80 (.00)	.30 (.00)	.54 (.02)
$\kappa = 0.6$	Y_1	**.97 (.00)**	.77 (.00)	.77 (.00)	.70 (.03)
	Y_2	**.85 (.00)**	.64 (.00)	.41 (.00)	.45 (.00)
$\kappa = 0.8$	Y_1	**.97 (.00)**	.89 (.00)	.84 (.00)	.83 (.00)
	Y_2	**.87 (.00)**	.70 (.00)	.66 (.00)	.66 (.00)

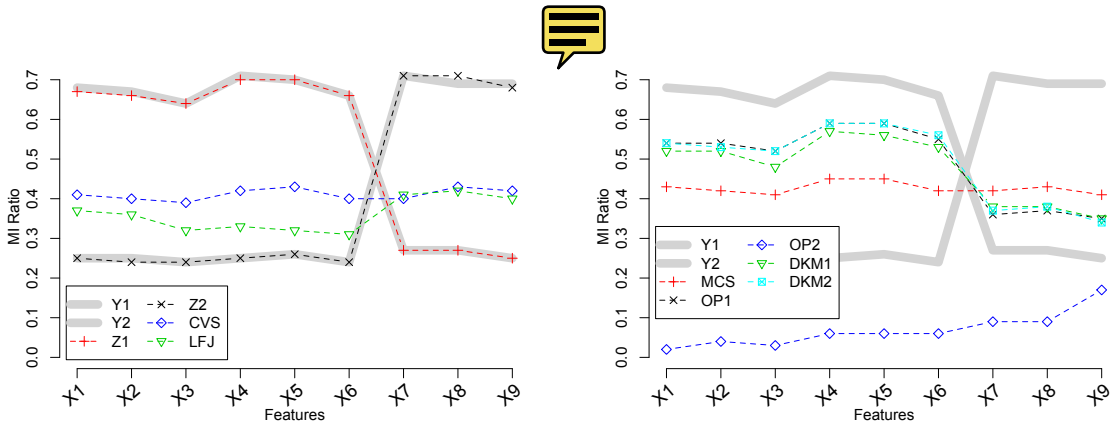
(b) PLTM and other multiple-clustering methods

Table 2: Clustering performances as measured by NMI on synthetic data. Best results are highlighted in bold. The first column indicates the κ used to generate the synthetic data.

clusterings did not hold in the latter situation. The method OP did poorly on this data set for all κ. Based on our observation, the first clustering found by OP was based on a combination of Y_1 and Y_2. This is different from the ideal case that one clustering is found in the first iteration while the other clustering is found in the subsequent iteration. It indicates that OP relies on the assumption that the first iteration focuses on only one clustering, and fails otherwise.

8.4 Feature Curves

Clustering involves learning a latent concept to partition data. The latent concept may not be obvious at a first sight. In this subsection, we introduce one way to visualize the latent concept using *feature curves*. We use the clusterings learned from synthetic data as examples. The feature curves are particularly useful for multiple clusterings methods, since these methods may produce numerous clusterings. The feature curves help the interpretation and selection of clusterings.



(a) PLTM and variable selection methods

(b) Other multiple-clustering methods

Figure 4: Features curves of the partitions obtained by various methods and those of the original class partitions (Y_1 and Y_2) on synthetic data ($\kappa = 0.6$).

The feature curve of one clustering is formed by plotting the dependency of Y between each of the attributes X. We quantify this dependency by

$$MI_{ratio}(X;Y) = I(X;Y)/H(Y)^5,$$

which we call *mutual information ratio*, since it is the ratio of the mutual information $I(X;Y)$ to the entropy of $H(Y)$. Roughly speaking, it shows the proportion of information about Y that can be told by the attribute X.

Figure 4a shows the feature curves of the two original class partitions Y_1 and Y_2 on synthetic data when $\kappa = 0.6$. These curves show that Y_1 is more correlated to attributes X_1–X_6 but less correlated to attributes X_7–X_9, and vice versa for Y_2. This can be expected from the structure of its generative model (Figure 1c), and shows the two facets of the data. The feature curve of Y_1 also shows it is not completely independent to X_7–X_9, which is reasonable because Y_1 and Y_2 are relatively correlated when $\kappa = 0.6$. The feature curves of the two latent variables Z_1 and Z_2, which were found by PLTM[6], are also shown in Figure 4a. They match those of the class partitions well, meaning that PLTM had successfully recovered the two facets of the synthetic data. This explains the good performance on PLTM. On the other hand, the feature curves for CVS and LFJ show that these two methods produce clusterings that depend similarly on the two facets. This indicates that they partitioned data based on both facets, and hence had mediocre performance.

Figure 4b shows the feature curves of the other multiple-clustering methods. MCS found only one clustering on this data set. Its feature curve and performance are similar to those of CVS and LFJ. The feature curves for DKM show that the two clusterings found (DKM1 and DKM2) were both based on facet X_1–X_6 but were also affected by the other facet. This suggests that DKM failed to find dissimilar clusterings on this data and also explains

5. To compute $I(X;Y)$ between a continuous variable X and a latent variable Y, we discretized X into 10 equal-width bins, so that $P(X,Y)$ could be estimated as a discrete distribution.
6. In the following, for any algorithm that had multiple runs, we draw the feature curves using an arbitrary run that had performance close to the algorithm average.

Data Set	Attributes	Classes	Samples
glass	9	6	214
image	18[7]	7	2310
ionosphere	33[7]	2	351
iris	4	3	150
vehicle	18	4	846
wdbc	30	2	569
wine	13	3	178
yeast	8	10	1484
zernike	47	10	2000

Table 3: Descriptions of UCI data sets.

its average performance on recovering Y_1 and poor performance on recovering Y_2. The first clustering OP1 found by OP has a feature curve similar to those for DKM. Since OP1 were correlated with all attributes, the second clustering OP2 found after orthogonal projection became relatively independent to all attributes, as shown by its feature curve. As a result, both iterations of OP could not find a feature curve that matched well with that of the class partition, and led to poor performance.

8.5 UCI Data

In the following experiment, we evaluated the clustering methods using some real-world data from the UCI machine learning repository (Frank & Asuncion, 2010). We chose 9 labeled data sets that contain only continuous attributes and have been commonly used in the literature. Table 3 shows the basic information of these data sets.

Tables 4a shows the clustering performances of PLTM and the other single-clustering methods. PLTM was clearly superior to the other variable selection methods. Specifically, it outperformed CVS on all of the nine data sets except that it was beaten by CVS on iris data. It outperformed LFJ on all the data sets. In most cases, PLTM outperformed CVS and LFJ by large margins.

PLTM also had clear advantages over GMM and MCLUST, the two methods that do not do variable selection. PLTM outperformed GMM on all the data sets except for ionosphere, iris, and wdbc data. It outperformed MCLUST on all the data sets except for iris and wdbc data. In most cases, PLTM again outperformed GMM and MCLUST by large margins.

The performances of the multiple-clustering methods are shown in Table 4b. PLTM outperformed MCS considerably on image, ionosphere, wine, and zernike data, and was comparable to MCS on the remaining data sets. PLTM was also generally better than the two distance-based methods. It outperformed OP and DKM markedly on 6 data sets, was comparable to them on zernike data, and was beaten by them on wdbc and yeast data. In fact, OP and DKM did not perform particularly better than k-means, on which they are based. This can be observed from Table 4b, which shows the performances of k-means for reference. OP was markedly better than k-means only on ionosphere data, while DKM showed at most a slight improvement on k-means on all these data. This is in

7. Attributes having single values had been removed.

Data Set	Facilitate VS	Perform VS		No VS	
	PLTM	CVS	LFJ	GMM	MCLUST
glass	**.43 (.03)**	.33 (.00)	.35 (.03)	.28 (.03)	.33 (.00)
image	**.71 (.03)**	.60 (.00)	.51 (.03)	.52 (.04)	.66 (.00)
vehicle	**.40 (.04)**	.21 (.00)	.27 (.01)	.25 (.08)	.36 (.00)
wine	**.97 (.00)**	.71 (.00)	.70 (.19)	.50 (.06)	.69 (.00)
yeast	**.18 (.00)**	.09 (.00)	.11 (.04)	.16 (.01)	.11 (.00)
zernike	**.50 (.02)**	.40 (.00)	.45 (.01)	.44 (.03)	.41 (.00)
ionosphere	.37 (.01)	.30 (.00)	.13 (.07)	**.57 (.04)**	.32 (.00)
iris	.76 (.00)	**.87 (.00)**	.68 (.02)	.73 (.08)	.76 (.00)
wdbc	.45 (.03)	.09 (.00)	.41 (.02)	.44 (.08)	**.68 (.00)**

(a) PLTM and single-clustering methods

Data Set	Model-based		Distance-based		
	PLTM	MCS	OP	DKM	KMEANS
image	**.71 (.03)**	.58 (.00)	.60 (.02)	.62 (.02)	.60 (.03)
ionosphere	**.37 (.01)**	.27 (.00)	.20 (.01)	.13 (.03)	.12 (.00)
wine	**.97 (.00)**	.70 (.00)	.88 (.00)	.88 (.00)	.88 (.00)
glass	**.43 (.03)**	.41 (.00)	.32 (.00)	.36 (.02)	.32 (.00)
iris	**.76 (.00)**	**.76 (.00)**	.66 (.00)	.66 (.00)	.66 (.00)
vehicle	.40 (.04)	**.42 (.00)**	.11 (.01)	.13 (.02)	.10 (.01)
zernike	.50 (.02)	.31 (.00)	.46 (.01)	**.51 (.01)**	.48 (.01)
wdbc	.45 (.03)	.43 (.00)	.59 (.00)	**.61 (.03)**	.59 (.00)
yeast	.18 (.00)	.15 (.00)	**.30 (.00)**	**.30 (.00)**	.30 (.00)

(b) PLTM and other multiple-clustering methods

Table 4: Clustering performances as measured by NMI on UCI data. Best results are highlighted in bold. Performances of k-means are included in Table (b) for reference. The horizontal lines divide the data sets into groups in which the performance of PLTM was better, comparable, or worse than those of the other methods.

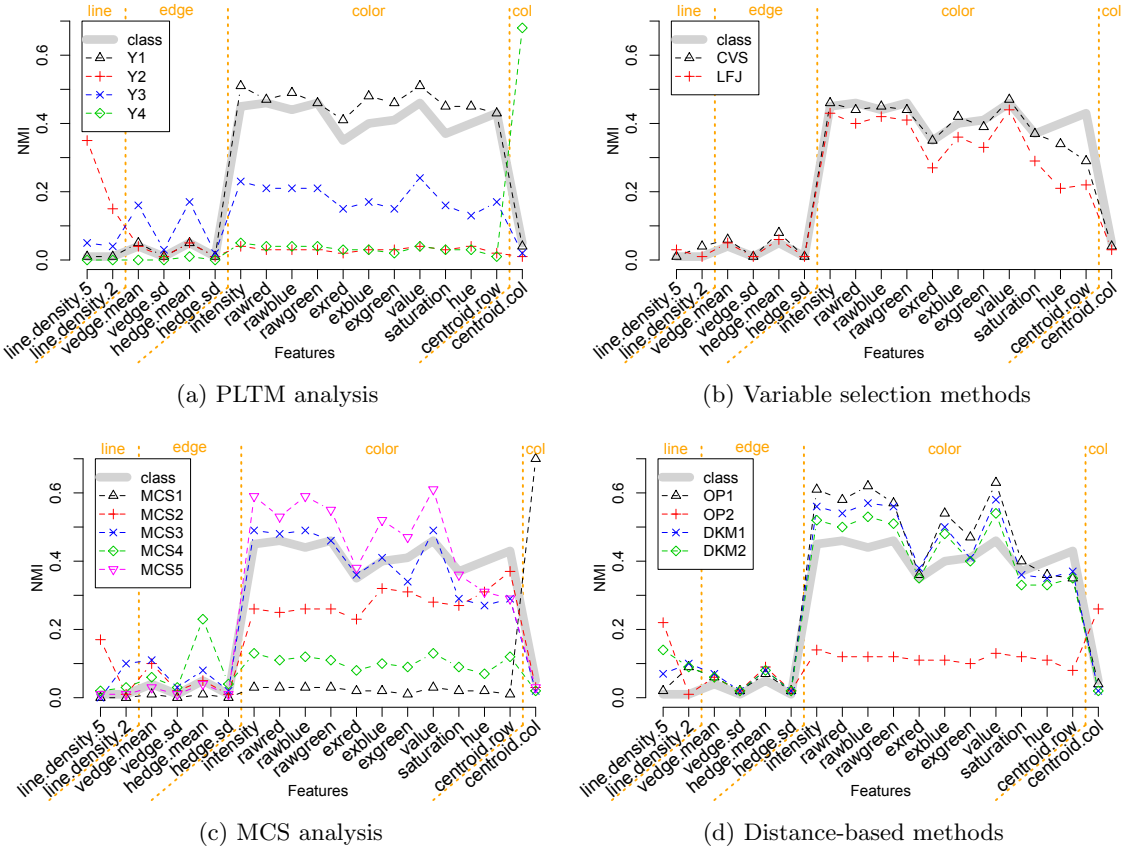
contrast to the dramatic improvement shown by PLTM over the GMM-based methods. This performance difference can possibly be explained by the different assumptions on multiple clusterings. MCS, OP, and DKM all assume independent clusterings, but PLTM assumes a tree structure of correlated clusterings. The results suggest that the assumption of PLTM is better.

After comparing the numerical results of different methods, we examine the models found by PLTM on three data sets to gain more insights into its superior performance.

8.5.1 Model from Image Data

In image data, each instance represents a 3×3 region of an image. It is described by 18 attributes. The feature curve of the original class partition is given in Figure 5. We see that it is a partition based on 10 color-related attributes from `intensity` to `hue` and the attribute `centroid.row`.

The structure of the model produced by PLTM analysis is shown in Figure 6. It contains four latent variables Y_1–Y_4. Their feature curves are shown in Figure 5a. We see that the feature curve of Y_1 matches that of the class partition beautifully. If the user chooses



(a) PLTM analysis

(b) Variable selection methods

(c) MCS analysis

(d) Distance-based methods

Figure 5: Features curves of the partitions obtained by various methods and that of the original class partition on image data.

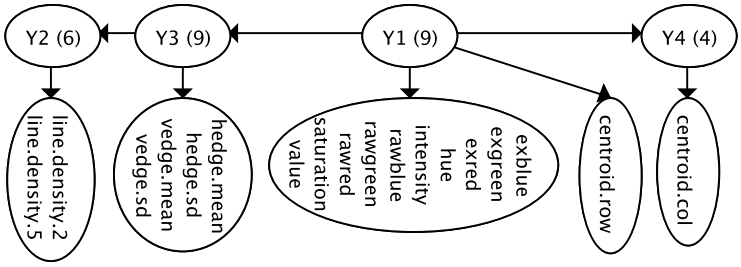


Figure 6: Structure of the PLTM learned from image data.

the partition represented by Y_1 as the final result, then the original class partition is well recovered. This explains the good performance of PLTM (NMI=0.71).

In addition to the good performance, PLTM has two other interesting findings on this data. First, the 10 color-related attributes semantically form a facet of the data. PLTM analysis has identified the facet in the pouch below Y_1. Moreover, it obtained a partition

based on not only the color attributes, but also the attribute `centroid.row`, the vertical location of a region in an image. This is interesting. It is because `centroid.row` is closely related to the color facet. Intuitively, the vertical location of a region should be correlated with the color of the region. For example, the color of the sky occurs more frequently at the top of an image and that of grass more frequently at the bottom.

Second, the latent variable Y_2 is strongly correlated with the two line density attributes. This is another facet of the data that PLTM analysis has identified. PLTM analysis has also identified the edge-related facet in the pouch below Y_3. However, it did not obtained a partition along the facet. The partition represented by Y_3 depends on attributes not only from the edge facet but also from the color facet as well. The two coordinate attributes `centroid.row` and `centroid.col` semantically form one facet. The facet has not been identified probably because the two attributes are not correlated.

We now look at the clusterings obtained by the other methods. The feature curves of the partitions obtained by LFJ and CVS are shown in Figure 5b. The LFJ curve matches that of the class partition quite well, but not as well as the feature curve of Y_1, especially on the attributes `exred`, `saturation`, `hue` and `centroid.row`. Consequently, the performance of LFJ (NMI=0.51) is not as good as PLTM. The feature curve of CVS may appear to match better to that of the class partition than that of PLTM. However, it shows a subtle difference on the attribute `centroid.row`. This difference shows that the partition by CVS obtains less information from `centroid.row` than it should be. In fact, CVS did not select this attribute for clustering. This missing information explains why the performance of CVS (NMI=0.60) is not as good as PLTM.

Figure 5c shows the feature curves of the clusterings produced by MCS. The feature curves show that MCS3 and MCS5 both cluster mainly based on the color facets. But both of their feature curves do not match that of the class partition very well, and therefore MCS did not do very well for image data (NMI=0.58). When we look at the the other clusterings obtained by MCS, they are not as meaningful as those obtained by PLTM. The clusterings generally do not respect the facets of data. For example, MCS2 is correlated with the color attributes, but is also correlated to one of the line density attributes. MCS4 is mainly correlated with one of the edge attribute, but not as much on the other edge attribute.

Figure 5d shows the feature curves of the distance-based methods. As we have suspected, the clusterings obtained by DKM are not very different from k-means. This is indicated by the fact that their feature curves are very close to that of OP1, which was obtained basically by k-means. The second iteration of OP produced a different clustering in OP2, which is mainly correlated with `line.density.5` and `centroid.col`. However, its meaning is not obvious and its usefulness is doubtful.

8.5.2 Model From Wdbc Data

The second data we look at is the wdbc data. This data were obtained from 10 computed features of 569 digitalized images of cell nuclei aspirated from breast masses. Each instance contains, for each feature, the mean (m), standard error (s), and worst value (w) of the cell nuclei in one image. Each instance is labeled as either benign or malignant.

Figure 7 shows the structure of PLTM learned from this data. We can see that this model identify some meaningful facets. The pouch below Y_1 identifies a facet related to

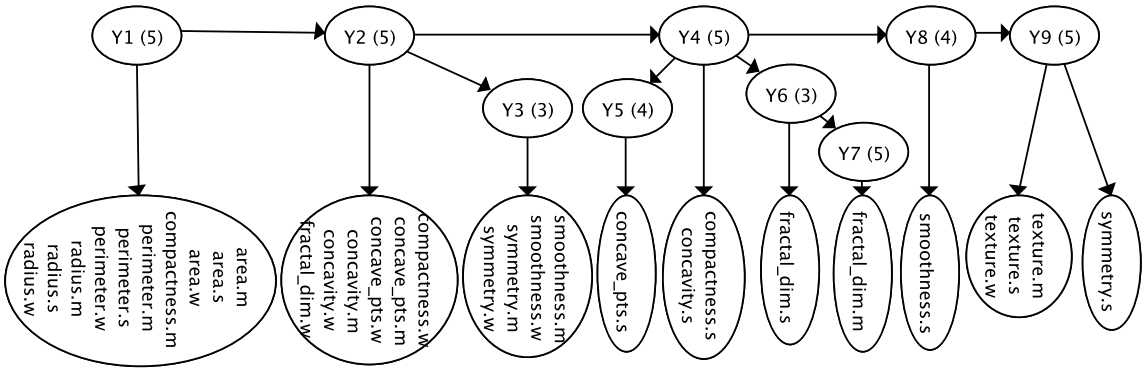


Figure 7: Structure of the PLTM learned from wdbc data.

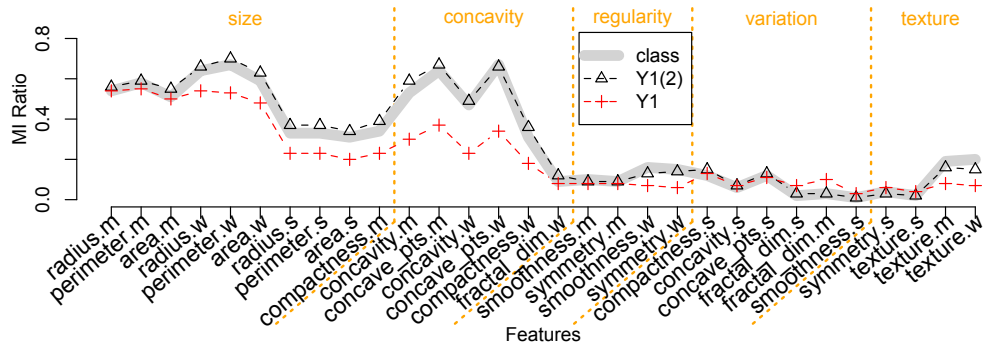


Figure 8: Features curves of the partition Y_1, obtained by PLTM, and that of the original class partition on wdbc data. $Y_1(2)$ is obtained by setting the cardinality of Y_1 to 2.

size of nuclei. It includes attributes mainly related to area, perimeter, and radius. The second facet is identified by the pouch below Y_2. It is related to the concavity and includes mainly the mean and worst values of the two features related to concavity. The third facet identified by the pouch below Y_3 is related to the regularity. It includes mean and worst values of smoothness and symmetry. The pouch below Y_9 identified a facet related to texture. Another attribute `symmetry.s` is also connected to Y_9. It appears that Y_9 obtains a clustering based on this attribute and the texture facet. The remaining attributes are all standard errors of some features and may considered as variation in the features. They are connected to the rest of the model through Y_4 and Y_8.

Although the model structure looks reasonable, the NMI achieved by PLTM is not good on this data (NMI=0.45). To understand the problem, we compare the feature curves of the class partition with the closest partition (Y_1) obtained by PLTM in Figure 8. The two feature curves indeed have similar shape. They indicate that both clusterings are mainly based on the size and concavity facets. However, the mutual information ratio between Y_1 and the attributes are lower in general. This is because the clustering Y_1 has 5 clusters but the class partition has only 2. The larger cardinality of Y_1 led to a larger entropy

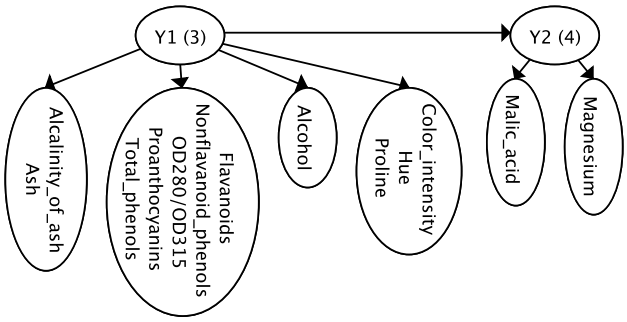


Figure 9: Structure of the PLTM learned from wine data.

$H(Y_1)$ and thus smaller values of mutual information ratio. To verify this explanation, we set the cardinality of Y_1 to 2. The feature curve of this adjusted latent variable ($Y_1(2)$) is shown in Figure 8. It now matches the feature curve of the class partition very well. This adjustment also improved the clustering performance of PLTM to 0.67 (sd=0.03) and made it comparable to the best score on this data set.

This data illustrates a limitation of using the class labels as gold standard. Even the data has only two classes, it may be more reasonable to group the marginal cases into a separate cluster and indicate that they require careful inspection. In this case, it will result in a lower NMI due to a higher number of clusters. Thus, the NMI may not reflect the whole truth of whether the clustering is reasonable. However, lacking of a better alternative, we still evaluate the clusterings based on the NMI with the class partitions.

8.5.3 Model from Wine Data

The PLTM learned from wine data is shown in Figure 9. The model structure also appears to be interesting. While we are not experts on wine, it seems natural to have `Ash` and `Alcalinity_of_ash` in one pouch as both are related to ash. Similarly, `Flavanoids`, `Nonflavanoid_phenols`, and `Total_phenols` are related to phenolic compounds. These compounds affect the color of wine, so it is reasonable to have them in one pouch along with the opacity attribute `OD280/OD315`. Moreover, both `Magnesium` and `Malic_acid` play a role in the production of ATP (adenosine triphosphate), the most essential chemical in energy production. So, it is not a surprise to find them connected to a second latent variable.

8.6 Discussion

In this section, we tested on two types of data. The synthetic data are known to have multiple clusterings, while the UCI data are labeled and known to have one meaningful partition. The variable selection methods, which can produce only single clusterings, are insufficient for the synthetic data with multiple clusterings. The multiple clustering methods are designed to handle this type of data. However, our experimental results show that they cannot handle data with correlated clusterings well. PLTM is shown to perform well on this data regardless of whether the clusterings are independent or correlated.

In contrast to synthetic data, each UCI data set has only one class partition. Variable selection methods are supposed to be designed for this kind of data. However, our experiments showed that PLTM could recover the class partitions better than the variable selection methods. In fact, the models obtained by PLTM analysis that are presented previously suggest that there may exist multiple meaningful clusterings even on this type of data. That may be the reason why variable selection methods did not work well, and it shows that it is better to facilitate variable selection with PLTM than to do variable selection.

On the other hand, the other multiple-clustering methods did not work as well as PLTM. They assume clusterings to be independent but those clustering found by PLTM are usually correlated. This invalid assumption explains why they perform worse than PLTM. Besides, PLTM could identify some meaningful facets from data and produce clusterings based on them. This is another advantage of PLTM over the other multiple-clustering methods.

9. Exploratory Analysis on Seasonal Statistics of NBA Players

In the second part of the empirical study, we perform an exploratory analysis on some seasonal statistics of National Basketball Association (NBA) players using PLTM. Unlike the previous data used, this data does not contain any labels. The objective here is not to recover any target partition. Rather, our aim is to see whether PLTM can obtain some interesting clusterings. We interpret the clusterings based on some basic basketball knowledge.

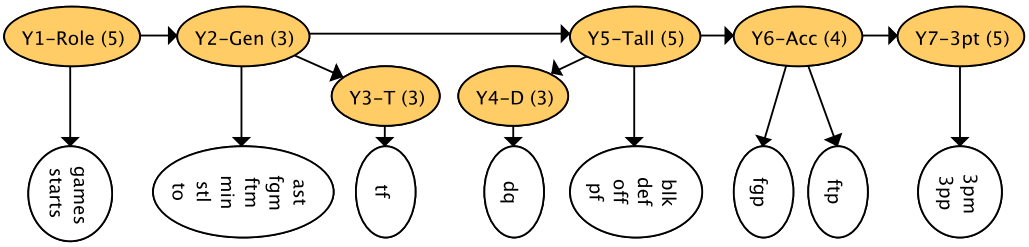
The data were collected from 441 players who played in at least one game in the 2009/10 season[8]. Each record corresponds to one player. It includes the numbers of games played (`games`) and started (`starts`) of a player in that season. It also contains 16 other per game averages, including minutes (`min`), field goals made (`fgm`), field goal percentage (`fgp`), three pointers made (`3gm`), three pointer percentage (`3gp`), free throws made (`ftm`), free throw percentage (`ftp`), offensive rebounds (`off`), defensive rebounds (`def`), assists (`ast`), steals (`stl`), blocks (`blk`), turnovers (`to`), personal fouls (`pf`), technical fouls (`tf`), and disqualifications (`dq`). This resulted in a data set with 18 attributes and 441 samples.

9.1 Clusterings Obtained

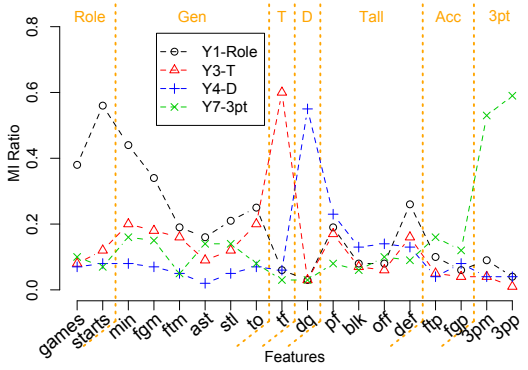
The structure of the model obtained from the PLTM analysis is shown in Figure 10a. The model contains 7 latent variables, each of which identifies a different facet of data. The first facet consists of attributes `games` and `starts`, which are related to role of a player. The second one consists of attributes `min`, `fgm`, `ftm`, `ast`, `stl`, and `to`, which are related to some general performance of a player. The third and fourth facets each contains only one attribute. They consist of `tf` and `dq`, respectively. The fifth facet contains attributes `blk`, `off`, `def`, and `pf`. This is related to one aspect of performance which taller players usually have an advantage in. The sixth facet consists of two attributes `ftp` and `fgp`, which are related to the shooting accuracies. The last facet contains `3pm` and `3pp`, which are related to three pointers.

We now try to interpret the meanings of the clusterings by inspecting the feature curves to see which identified facets the clusterings are based on. Figure 10b shows the feature

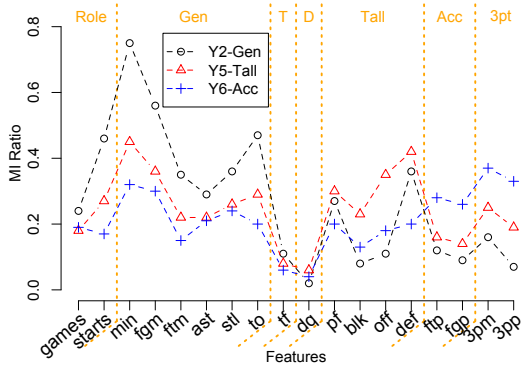
8. http://www.dougstats.com/09-10RD.txt



(a) Structure of model. The latent variables are shown in shaded nodes.



(b) Feature curves of Y_1-`Role`, Y_3-`T`, Y_4-`D`, and Y_7-`3pt`.



(c) Feature curves of Y_1-`Gen`, Y_5-`Tall`, and Y_6-`Acc`.

Figure 10: PLTM analysis on NBA data.

curves of Y_1, Y_3, Y_4, and Y_7. The feature curves show that Y_3, Y_4, and Y_7 partition data based on their corresponding facets. Therefore, we renamed the latent variables Y_3, Y_4, and Y_7 as `T` (technical fouls), `Q` (disqualifications), and `3pt` (three-pointer) , respectively. The feature curve of Y_1 shows that it is mainly correlated with `games` and `starts`, but also correlated with `min` and `fgm`. It is possible that the last two attributes only reflect the role of a player, therefore, we consider Y_1 to be a clustering on the role of player and renamed it accordingly.

Figure 10c shows the feature curves of the remaining latent variables. It shows that Y_2 clusters mainly on its corresponding facet with much emphasis on `min`. We hence renamed Y_2 as `Gen` (general). The other two clusterings are less definite. Their feature curves show that both Y_5 and Y_6 are correlated with the role facet and the general facet. However, we can see that `ftp`, `fgp`, `3pm`, and `3pp` have higher correlation with Y_6 than with other latent variables except Y_7 (`3pt`). Similarly, `blk` and `off` have relatively high correlation with Y_5. This suggests that what distinguish the clusterings Y_6 and Y_5 from the rest are attributes on the two shooting related facets and the facet related to tall man abilities, respectively. We therefore labeled Y_6 as `Acc` (shooting accuracy) and Y_5 as `Tall` (tall man abilities).

9.2 Cluster Means

The feature curves show what attributes are relevant to a clustering, but does not tell much about the clusters in a clustering. To understand the clusters, we may examine the mean

Role	P(Role)	games	starts
occasional	0.32	29.1	2.0
irreg_starter	0.11	46.1	31.7
reg_sub	0.19	68.2	5.4
reg	0.13	75.8	32.8
reg_starter	0.25	76.0	73.7
overall	1.00	56.3	27.9

(a) Role

3pt	P(3pt)	3pm	3pp
never	0.29	0.00	0.00
seldom	0.12	0.08	0.26
fair	0.17	0.33	0.28
good	0.40	1.19	0.36
extreme	0.02	0.73	0.64
overall	1.00	0.55	0.23

(b) 3pt

Acc	P(Acc)	fgp	ftp
low_ftp	0.10	0.44	0.37
low_fgp	0.16	0.39	0.72
high_ftp	0.47	0.44	0.79
high_fgp	0.28	0.52	0.67
overall	1.00	0.45	0.71

(c) Acc

Table 5: Means of attributes conditional on the specified latent variables on NBA data. The second columns show the marginal distributions of those latent variables, and the last rows show the unconditional means of the attributes.

values of attributes of the clusters. We use the clusterings Role, 3pt, and Acc as examples to illustrate that PLTM can find some interesting clusters.

Table 5a shows the means of games and starts conditional on the clustering Role. We see that players belonging to first two clusters did not play regularly. We labeled the first cluster as "occasional", since players belonging to it played occasionally. The second group of players also played less often than average, but they usually started in a game when they played. This cluster probably refers to those players who had the calibre of starters but had missed part of the season due to injuries or other reasons. The third group of players played often, but usually played as a substitute. We named this state as "reg_sub" (regular substitute). The fourth group of players played regularly and sometimes even started in a game. We just called this group "reg" (regular). The last group contains players who played and started regularly, so we called them "reg_starter" (regular starter).

Table 5b shows the means of 3pm and 3pp conditional on 3pt. 3pt partitions players into 5 clusters. The first two clusters ("never" and "seldom") were named based on how often the players belonging to them had made a three-pointer. The next two clusters ("fair" and "good") were named based on the three-pointer accuracies of the players. The last group was named as an extreme, as it contains players shooting with surprisingly high accuracy. As indicated by the marginal distribution, it consists of only a very small proportion of players. This is possibly a group of players who had made some three pointers during the sporadic games that they had played. The accuracy remained very high since they did not play often.

Table 5c shows the means of fgp and ftp conditional on Acc. The first group of players had particularly poor free throw percentage, so we named it "low_ftp". The second

Role	Gen		
	poor	fair	good
occasional	0.81	0.19	0.00
irreg_starter	0.00	0.69	0.31
reg_sub	0.22	0.78	0.00
regular	0.00	0.81	0.19
reg_starter	0.00	0.06	0.94

(a) $P(\texttt{Gen}|\texttt{Role})$

Tall	Acc			
	low_ftp	low_fgp	high_ftp	high_fgp
poor	0.28	0.53	0.00	0.18
fair	0.00	0.02	0.95	0.03
good	0.00	0.00	1.00	0.00
good_big	0.11	0.04	0.00	0.86
v_good	0.00	0.00	0.14	0.86

(b) $P(\texttt{Acc}|\texttt{Tall})$

Table 6: Conditional distributions of `Gen` and `Acc` on NBA data.

group of players was labeled as "low_fgp", since they had low field goal percentage and average free throw percentage. The third group of players had particularly high free throw percentage, while the fourth group of players had particularly high field goal percentage. We named them as "high_ftp" and "high_fgp", respectively. One may expect that since both `ftp` and `fgp` are related to the shooting accuracies, these two attributes should be positively correlated and the last two groups may look counter-intuitive. However, it is indeed reasonable due to one observation in NBA. Taller players usually stay closer to basket in games. They therefore take high-percentage shots more often and has higher field goal percentage. On the other hand, taller players are usually poorer in making free throws and have lower free throw percentage. One typical example is Dwight Howard. He had $\texttt{fgp} = 0.61$ but $\texttt{ftp} = 0.59$. He was classified appropriately as "high_fgp" by PLTM.

9.3 Relationships Between Clusterings

In addition to the distributions of individual clusterings, PLTM analysis models the probabilistic relationships between the clusterings. These relationships may also be interesting to users, as we demonstrate using the following two examples.

Table 6a shows the conditional distribution $P(\texttt{Gen}|\texttt{Role})$. The clusters represented by `Gen` were named according to the cluster means similarly as what we do in previous subsection. We observe that those players playing occasionally were mostly poor in general, and those starters almost always played well in general. While the other three groups of players usually played fairly, more of the irregular starters played well than those played regularly, and none of the regular substitutes played well. This relationship is reasonable because a player's general performance should be correlated with the role of the player.

Table 6b shows the conditional distribution $P(\texttt{Acc}|\texttt{Tall})$. It is consistent with our observation that taller men usually shoot free throws more poorly. Most players who played very well ("v_good") or well specifically ("good_big") as tall men belong to the group that has high field goal percentage but low free throw percentage ("high_fgp"). On the other hand, those who do not play well specifically as big men ("fair" and "good") usually have average field goal percentage and higher free throw percentage ("high_ftp"). For those who played poorly as a tall man, we cannot tell much about them.

Subset	Attributes	$\lvert Y\rvert$
1	games, starts, ftm, ftp	7
2	min, off, def, ast, to blk, pf	4
3	fgm, tf	18
4	fgp, 3pm, 3pp	10
5	stl, dq	16

Table 7: Partitions of attribute by MCS on NBA data. The last column lists the number of components that a GMM has on each attribute subset.

9.4 Comparison with MCS

After presenting the result from PLTM analysis, we now compare it with other methods. Both OP and DKM require the knowledge of number of clusterings and numbers of clusters beforehand. However, we do not have such knowledge in this exploratory analysis setting. Therefore, they are not applicable in this analysis.

We also analyzed this data with MCS. It has three disadvantages compared with PLTM. First, the partition of attributes are less natural than those facets identified by PLTM. This can be observed from Table 7. For example, fgm and tf in Subset 3 look unrelated, but they were grouped into a separate subset. Similarly, the apparently unrelated stl and dq were grouped together in Subset 5. While grouping games and starts together leads to a meaningful clustering, as PLTM previously shows, MCS added two more seemingly unrelated attributes ftp and ftm along with them in Subset 1. The second problem of the clusterings found by MCS is that they all have large number of clusters, only except for the Subset 2, as shown in the last column of Table 7. These two disadvantages make it more difficult to comprehend the clusterings found by MCS than those found by PLTM. The third limitation of MCS is that it has independent clusterings. Therefore, it cannot show the possibly meaningful relationships between the clusterings as PLTM does.

10. Concluding Remarks

In this paper, we propose PLTMs as a generalization of GMMs for multiple clusterings. These multiple clusterings can be used to facilitate variable selection since each of them is usually primarily related to only a subset of variables. Our empirical results on synthetic data and UCI data show that it is more reasonable to facilitate than to do variable selection.

When compared with other multiple-clustering methods, our empirical results show that PLTM analysis generally perform better than those methods on the synthetic data and UCI data. Our exploratory analysis on NBA data also shows that PLTM can produce some meaningful clusterings and model the probabilistic relationships between them. This demonstrate what PLTM analysis can do but the other multiple-clustering methods cannot.

One drawback of PLTM analysis is that the training is slow. For example, learning a PLTM took around 5 hours on data sets of moderate size (e.g., image and wdbc data) and around 2.5 days on zernike data, the largest data set used in our experiments. One main reason for the slow training time is that learning PLTMs involves model selection. A PLTM analysis does not need to specify the number of clusterings or numbers of clusters, which

is desirable since it is often unknown in applications such as exploratory analysis. Rather than requiring this information beforehand, the training process automatically searches for the optimal model structure that provides this information. This process requires estimation and evaluation of many candidate models before the optimal model can be obtained. Consequently, the training time of PLTM should not be compared directly with methods that do not include model selection.

Although the slow training limits the application of PLTM on data with more than hundreds of attributes, one should note what kind of data is appropriate to PLTM analysis. We may broadly distinguish data into two kinds according to the nature of attributes. The first kind of data contains primitive attributes. The values of those attributes are usually generated by machine and they do not contain much information about the objects when considered individually. Some data with very high-dimensions such as image data and text data fall into this category. For example, in some image pixels data, a pixel attribute only tells how bright one pixel is, but does not tell much else about an object. The second kind of data contains descriptive attributes. An attribute in this kind of data usually describes something meaningful about the objects. The UCI data that we focus on and the NBA data belong to this kind. PLTM is good at data with descriptive attributes, since this kind of attributes should lead to more meaningful facets, that PLTM can adeptly identify. And even though PLTM cannot handle very large data, the exploratory analysis on NBA data shows that we find good application of PLTM on this kind of data with as few as 18 attributes.

On the other hand, more research should be done to speed up the learning of PTLMs, so that we can obtain results more quickly and apply PLTM on larger data. This can possibly be achieved by parallelization or a better heuristic search. In particular, the numbers of candidate models generated by some search operators are quadratic to the number of variables. This limits the dimension of data that PLTM can be applied to and need further improvement. We plan to work on this direction in the future.

Acknowledgements

Research on this work was supported by Hong Kong Research Grants Council GRF Grant #622408 and The National Basic Research Program of China (aka the 973 Program) under project No. 2003CB517106. The first author was on leave at HKUST Fok Ying Tung Graduate School when the work was done.

References

Bae, E., & Bailey, J. (2006). COALA: A novel approach for the extraction of an alternative clustering of high quality and high dissimilarity. In *Proceedings of the Sixth IEEE International Conference on Data Mining*.

Caruana, R., Elhawary, M., Nguyen, N., & Smith, C. (2006). Meta clustering. In *Proceedings of the Sixth International Conference on Data Mining*.

Chen, T., Zhang, N. L., & Wang, Y. (2008). Efficient model evaluation in the search-based approach to latent structure discovery. In *Proceedings of the Fourth European Workshop on Probabilistic Graphical Models*, pp. 57–64.

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

Cowell, R. G., Dawid, A. P., Lauritzen, S. L., & Spiegelhalter, D. J. (1999). *Probabilistic Networks and Expert Systems*. Springer.

Cui, Y., Fern, X. Z., & Dy, J. G. (2007). Non-reduntant multi-view clustering via orthogonalization. In *Proceedings of the Seventh IEEE International Conferene on Data Mining*.

Dasgupta, S., & Ng, V. (2010). Mining clustering dimensions. In *Proceedings of the 27th International Conference on Machine Learning*.

Dempster, A. P., Laird, N. M., & Rubin, D. B. (1977). Maximum likelihood from incomplete data via the EM algorithm. *Journal of the Royal Statistical Society. Series B (Methodological)*, *39*(1), 1–38.

Dy, J. G., & Brodley, C. E. (2004). Feature selection for unsupervised learning. *Journal of Machine Learning Research*, *5*, 845–889.

Fonseca, J. R., & Cardoso, M. G. (2007). Mixture-model cluster analysis using information theoretical criteria. *Intelligent Data Analysis*, *11*, 155–173.

Fraley, C., & Raftery, A. E. (2002). Model-based clustering, discriminant analysis, and density estimation. *Journal of American Statistical Association*, *97*(458), 611–631.

Fraley, C., & Raftery, A. E. (2006). MCLUST version 3 for R: Normal mixture modeling and model-based clustering. Tech. rep. 504, Department of Statistics, University of Washington. (revised 2009).

Frank, A., & Asuncion, A. (2010). UCI machine learning repository. University of California, Irvine, School of Information and Computer Sciences. http://archive.ics.uci.edu/ml.

Galimberti, G., & Soffritti, G. (2006). Identifying multiple cluster structures through latent class models. In *From Data and Information Analysis to Knowledge Engineering*. Spring Berlin Heidelberg.

Galimberti, G., & Soffritti, G. (2007). Model-based methods to identify multiple cluster structures in a data set. *Computational Statistics and Data Analysis*, *52*, 520–536.

Geiger, D., & Heckerman, D. (1994). Learning Gaussian networks. Tech. rep. MSR-TR-94-10, Microsoft Research.

Gondek, D., & Hofmann, T. (2004). Non-redundant data clustering. In *Proceedings of the Fourth IEEE International Conference on Data Mining*.

Hoff, P. D. (2006). Model-based subspace clustering. *Bayesian Analysis*, *1*(2), 321–344.

Ingrassia, S. (2004). A likelihood-based constrained algorithm for multivariate normal mixture models. *Statistical Methods and Applications*, *13*(2), 151–166.

Jain, P., Meka, R., & Dhillon, I. S. (2008). Simultaneous unsupervised learning of disparate clusterings. In *Proceedings of the Seventh SIAM International Converence on Data Mining*, pp. 858–869.

Kriegel, H.-P., Kröger, P., & Zimek, A. (2009). Clustering high dimensional data: A survey on subspace clustering, pattern-based clustering, and correlation clustering. *ACM Transactions on Knowledge Discovery from Data*, *3*(1), 1–58.

Lauritzen, S. L., & Jensen, F. (2001). Stable local computation with conditional Gaussian distributions. *Statistics and Computing*, *11*, 191–203.

Law, M. H. C., Figueiredo, M. A. T., & Jain, A. K. (2004). Simultaneous feature selection and clustering using mixture models. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *26*(9), 1154–1166.

Li, Y., Dong, M., & Hua, J. (2009). Simultaneous localized feature selection and model detection for gaussian mixtures. *IEEE Transaction on Pattern Analysis and Machine Intelligence*, *31*(5), 953–960.

Liu, J. S., Zhang, J. L., Palumbo, M. J., & Lawrence, C. E. (2003). Bayesian clustering with variable and transformation selections (with discussion). *Bayesian Statistics*, *7*, 249–275.

Madeira, S. C., & Oliveria, A. L. (2004). Biclustering algorithms for biological data analysis: A survey. *IEEE Transactions on Computational Biology and Bioinformatics*, *1*(1), 24–45.

Maugis, C., Celeux, G., & Martin-Magniette, M.-L. (2009a). Variable selection for clustering with Gaussian mixture models. *Biometrics*, *65*, 701–709.

Maugis, C., Celeux, G., & Martin-Magniette, M.-L. (2009b). Variable selection in model-based clustering: A general variable role modeling. *Computational Statistics and Data Analysis*, *53*, 3872–3882.

McLachlan, G. J., & Peel, D. (2000). *Finite Mixture Models*. Wiley, New York.

Niu, D., Dy, J. G., & Jordan, M. I. (2010). Multiple non-redundant spectral clustering views. In *Proceedings of the 27th International Conference on Machine Learning*.

Pan, W., & Shen, X. (2007). Penalized model-based clustering with application to variable selection. *Journal of Machine Learning Research*, *8*, 1145–1164.

Qi, Z., & Davidson, I. (2009). A principled and flexible framework for finding alternative clusterings. In *Proceedings of the 15th ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*.

Raftery, A. E., & Dean, N. (2006). Variable selection for model-based clustering. *Journal of American Statistical Association*, *101*(473), 168–178.

Schwarz, G. (1978). Estimating the dimension of a model. *The Annals of Statistics*, *6*(2), 461–464.

Steinley, D., & Brusco, M. J. (2008). Selection of variables in cluster analysis: An empirical comparison of eight procedures. *Psychometrika*, *73*(1), 125–144.

Strehl, A., & Ghosh, J. (2002). Cluster ensembles — a knowledge reuse framework for combining multiple partitions. *Journal of Machine Learning Research*, *3*, 583–617.

Zeng, H., & Cheung, Y.-M. (2009). A new feature selection method for gaussian mixture clustering. *Pattern Recognition*, *42*, 243–250.

Zhang, N. L. (2002). Hierarchical latent class models for cluster analysis. In *Proceedings of the Eighteenth National Conference on Artificial Intelligence*.

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

Zhang, N. L., & Kočka, T. (2004). Efficient learning of hierarchical latent class models. In *Proceedings of the Sixteenth IEEE International Conference on Tools with Artificial Intelligence*, pp. 585–593.