Variable Selection in Model-Based Clustering: To Do or To Facilitate

Leonard K. M. Poon[†] LKMPOON@CSE.UST.HK
Nevin L. Zhang[†] LZHANG@CSE.UST.HK
Tao Chen[‡] CT.PKU.HKUST@GMAIL.COM
Yi Wang[†] WANGYI@CSE.UST.HK

[†]Department of Computer Science and Engineering, 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 is 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, 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 generalization of the Gaussian mixture model, show its ability to cluster data along multiple facets, and demonstrate it is often more reasonable to facilitate variable selection than to perform it.

1. Introduction

Variable selection is an important issue for cluster analysis of high-dimensional data. 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. Recently there is growing interest in the issue (Dy & Brodley, 2004; Steinley & Brusco, 2008; Zeng & Cheung, 2009). This paper is concerned with variable selection for model-based clustering.

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. Several 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 & Dean (2006) consider a variation of the Gaussian mixture model (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 search step, two nested models are compared using the Bayes factor and the better one is chosen to seed the next search step. Law et al. (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 et al. (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 et al., 2003; Hoff,

Appearing in *Proceedings of the 27th International Conference on Machine Learning*, Haifa, Israel, 2010.

2006) and regularization methods (Pan & Shen, 2007) have been developed based on this idea.

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.

Instead of performing variable selection, we advocate to facilitate variable selection by domain experts. 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 one of the clusterings is found interesting.

To realize the idea, we generalize 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.

The rest of the paper is organized as follows. We first introduce PLTMs in Section 2. In Sections 3 and 4, we discuss how to learn PLTMs from data. In Section 5, we present the empirical results. Related works are reviewed in Section 6 while conclusions are given in Section 7. Our discussion of the learning algorithm will be brief so that there is enough room to cover all the interesting empirical findings. More details of the learning algorithm will be given in an extended version of the paper.

2. Pouch Latent Tree Models

A *pouch latent tree model* (PLTM) is a rooted tree, where each internal node represents a latent (or unobserved) variable, and each leaf node represents a set of manifest (or observed) 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 1(a) 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. And we use the terms 'variable' and 'node' interchangeably. In a PLTM, the dependency of a latent variable Y on its parent Y' is characterized by a conditional discrete distribution $P(y|y')$.[1] Let $\boldsymbol{X}$ be the set of continuous manifest variables in a pouch node with parent node Y. We assume that, given a value y of Y, $\boldsymbol{X}$ follow the conditional Gaussian distribution $P(\boldsymbol{x}|y) = \mathcal{N}(\boldsymbol{x}|\boldsymbol{\mu}_y, \boldsymbol{\Sigma}_y)$ 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.

PLTMs generalize GMMs. As a matter of fact, a GMM is a PLTM that has only one latent variable and a single pouch node containing all manifest variables. Figure 1(b) depicts a GMM as a PLTM, where Y_1 is the latent variable and X_1–X_9 are the continuous manifest variables. PLTMs also resemble hierarchical latent class models (Zhang, 2004), except that the latter consists of only discrete variables and do not have pouch nodes.

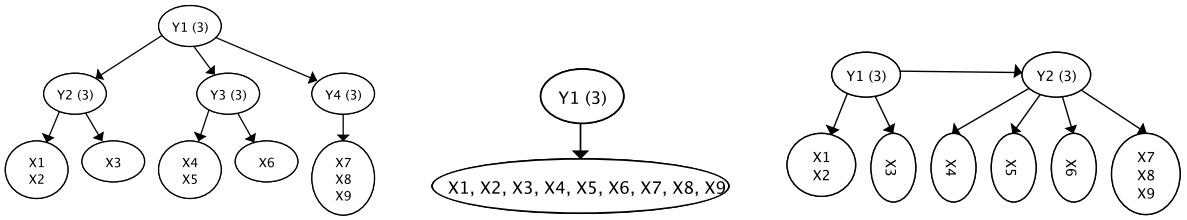
Cluster analysis based on PLTM requires learning a PLTM from data. In the next section, we first discuss parameter estimation. In Section 4, we discuss learning both model structure and parameters from data.

3. 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 (Dempster et al., 1977). 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-

[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.



(a) (b) (c)

Figure 1. (a) An example of pouch latent tree model. 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.

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 using, with minor adaptions, the exact inference algorithm for mixed Bayesian networks by Lauritzen & Jensen (2001). For each i, let $\boldsymbol{x}_i$ be the values of $\boldsymbol{X}$ in the sample $\boldsymbol{d}_i$. In the M-step, the new estimate $\boldsymbol{\theta}^{(t)}$ is obtained as follows:

$$P(y|y', \boldsymbol{\theta}^{(t)}) \propto \sum_{i=1}^{N} P\left(y, y'|\boldsymbol{d}_i, \boldsymbol{\theta}^{(t-1)}\right)$$

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

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

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 $\boldsymbol{\mu}_{\mathcal{D},\boldsymbol{X}}$ and $\boldsymbol{\Sigma}_{\mathcal{D},\boldsymbol{X}}$ be the sample mean and covariance of $\mathcal{D}$ projected on variables $\boldsymbol{X}$. For the conditional Gaussian distribution of $\boldsymbol{X}$, the means $\boldsymbol{\mu}_y^{(0)}$ are generated from $\mathcal{N}(\boldsymbol{\mu}_{\mathcal{D},\boldsymbol{X}}, \boldsymbol{\Sigma}_{\mathcal{D},\boldsymbol{X}})$, and the covariances $\boldsymbol{\Sigma}_y^{(0)}$ are initialized to be $\boldsymbol{\Sigma}_{\mathcal{D},\boldsymbol{X}}$.

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 $\boldsymbol{X}$. We impose the following constraints on the eigenvalues $\lambda^{(t)}$ of the matrix $\boldsymbol{\Sigma}_y^{(t)}$: $\sigma^2_{min}/\gamma \leq \lambda_i^{(t)} \leq \gamma \times \sigma^2_{max}$, where σ^2_{min} and σ^2_{max} are the minimum and maximum of the sample variances of the variables $\boldsymbol{X}$ on $\mathcal{D}$ (i.e. the diagonal values of the matrix $\boldsymbol{\Sigma}_{\mathcal{D},\boldsymbol{X}}$) and γ is a parameter for our method.

4. 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. 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 complexity.

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

There are four aspects of the structure m, namely, the number of latent variables, the cardinalities of 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 7 search operators. A node introduction operator and a node deletion operator are used to add and remove a latent variable, respectively. A state introduction operator and a state deletion operator are used to add and remove a state to and from a latent variable, respectively. A node relocation operator is used

to modify connections among nodes. Finally, a pouching operator is used to merge two pouch nodes, and an unpouching operator is used to split a pouch node into two pouch nodes.

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. They include a way to approximately evaluate candidate models without running complete EM on them, and a way to control search such that only a subset of search operators are used at each search step. Due to space limit, we do not describe those techniques. Instead, we focus on the empirical results and show that PLTMs provide us with an interesting new way to analyze unlabeled data.

5. Empirical Results

Our empirical investigation is designed to compare two types of analyses that can be applied to unlabeled data: PLTM analysis and GMM analysis. PLTM analysis yields a model with multiple latent variables. Each of the latent variables represents a partition of data and may depend only on a subset of attributes. GMM analysis produces a model with a single latent variable. It can be done with or without variable selection. This paper is primarily concerned with GMM analysis with variable selection. GMM analysis without variable selection is included for reference. When variable selection is performed, the latent variable may depend only on a subset of attributes.

5.1. Data Sets and Algorithms

We used both synthetic and real-world data sets in our experiments. The synthetic data were generated from the model shown in Figure 1(c). This model has 9 continuous manifest variables X_1–X_9 and 2 discrete latent variables Y_1–Y_2. Each latent variable has 3 possible values. The root variable Y_1 is uniformly distributed. Given a value of Y_1, Y_2 takes the same value with probability 0.5 and each of the other two values with probability 0.25. The conditional distributions of the manifest variables are all Gaussian. Their means are determined by the values of their parents and can be 0, 2.5, 5. All the manifest variables have variance 1 and the covariance between any pair of these variables in the same pouch is 0.5. The data were obtained by sampling 1,000 data cases from this model. The variable Y_1 was designated as the class variable, and the other latent variable Y_2 was excluded from the data.

The real-world data sets were borrowed from the UCI machine learning repository[2]. We chose 9 labeled data sets that are commonly used in the literature and that contain only continuous attributes. Table 1 shows the basic information of these data sets.

Table 1. Descriptions of data sets used in our experiments. The last column shows the numbers of latent variables obtained by PTLM analysis.

Data Set	Attr.	Classes	Samples	Latents
SYNTHETIC	9	3	1000	2
IMAGE	18[4]	7	2310	4
IONOSPHERE	33[4]	2	351	10
IRIS	4	3	150	1
SONAR	60	2	208	11
VEHICLE	18	4	846	2
WDBC	30	2	569	8
WINE	13	3	178	2
YEAST	8	10	1484	5
ZERNIKE	47	10	2000	7

We compare PLTM analysis with four methods based on GMMs. The first method is plain GMM analysis. The second one is MCLUST (Fraley & Raftery, 2006), which reduces the number of parameters by imposing constraints on the eigenvalue decomposition of the covariance matrices. The third one is CLUSTVARSEL (Raftery & Dean, 2006)[3], 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 methods, the last two perform variable selection while the first two do not.

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 3, we set the constant γ at 20. For GMM and CVS, the true numbers of classes were given as input. For MCLUST and LFJ, the maximum number of mixture components was set at 20.

5.2. Method of Comparison

Our experiments started with labeled data. We split the data into training and testing sets using 5-fold stratified cross-validation. In the training phase, mod-

[2] http://www.ics.uci.edu/∼mlearn/MLRepository.html
[3] http://cran.r-project.org/web/packages/clustvarsel
[4] Attributes having single values had been removed.

els were learned from training sets with the class labels removed. In the testing phase, the clusterings contained in models were evaluated and compared based on the testing sets. The objective is to see which method recovers the class variable the best.

A model produced by a GMM-based method contains a single latent variable Y. It represents one way to partition the data. We follow Strehl & Ghosh (2002) and evaluate the partition using normalized mutual information $NMI(C;Y)$ between Y and the class variable C. The NMI 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 $P(C,Y) = \frac{1}{N}\sum_{i=1}^{N} P(C|\boldsymbol{d}_i)P(Y|\boldsymbol{d}_i)$, where $\boldsymbol{d}_1,\ldots,\boldsymbol{d}_N$ are the samples in testing data.

A model resulting from PLTM analysis contains a set $\boldsymbol{Y}$ of latent variables. Each of the latent variables represents a partition of the data. Each combination of the latent variables also represents a partition of the data. In practice, the user may find several of the partitions interesting and use them all in his work. In this section, however, 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 PLTM analysis provides the possibility for the user to recover the original class partition. Consequently, we assume that the user chooses, among all the partitions produced by PLTM analysis, the one closest to the class partition and we evaluate the performance of PLTM using this quantity:

$$\max_{\boldsymbol{W}\subseteq\boldsymbol{Y}} NMI(C;\boldsymbol{W}).$$

This quantity is difficult to compute when the number of latent variables is large. We resort to a greedy search that adds one latent variable Y to the set $\boldsymbol{W}$ at a time.

Among the four GMM-based methods, CVS and LFJ make explicit efforts to perform variable selection, while GMM and MCLUST do not. PLTM does not make explicit effort to perform variable selection either. However, it produces multiple partitions of data and some of the partitions may depend only on subsets of attributes. Consequently, it allows the user to examine the clustering results based on various subsets of attributes and choose the ones he deems interesting. In this sense, we can view PLTM as a method to facilitate variable selection. So, our empirical work can be viewed as a comparison between two different approaches to variable selection: to do (CVS and LFJ) or to facilitate (PLTM). This explains the title of the paper.

5.3. Results

The results of our experiments are given in Table 2. Let us first compare the results of PLTM with those of the two variable selection methods, CVS and LFJ. The performances of PLTM are clearly superior. Specifically, it outperformed CVS on all the 10 data sets except that it was beaten by CVS on iris data. It outperformed LFJ on all the data sets except that it tied with LFJ on wine data. In most cases, PLTM outperformed CVS and LFJ by large margins.

PLTM also has 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 and wdbc data. It outperformed MCLUST on all the data sets except for iris and wdbc data. In most cases, PLTM outperformed GMM and MCLUST by large margins.

5.4. Explaining the Results

We next examine models produced by the various methods to gain insights about the superior performance of PLTM. Evidently it is not possible to look at all models produced during cross validation. We examine the models learned from the entire synthetic data and those learned from the entire image data.

5.4.1. Models from Synthetic Data

Before examining models obtained from synthetic data, we first take a look at the data set itself. The data were sampled from the model shown in Figure 1(c), with information about the two latent variables Y_1 and Y_2 removed. Nonetheless, the latent variables represent two natural ways to partition the data. To see how the partitions are related to the attributes, we plot the NMI[5] between the latent variables and the attributes in Figure 2. We call the curve for a latent variable its *feature curve*. We see that Y_1 is strongly correlated with X_1–X_3, but not with the other attributes. Hence it represents a partition based on those three attributes. Similarly, Y_2 represents a partition of the data based on attributes X_4–X_9. So, we say that

[5]To compute $NMI(X;Y)$ 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.

Table 2. Clustering performances as measured by NMI. Results that are significantly better than all others are highlighted in bold. The first row is a categorization of the methods in terms of the approach to variable selection (VS).

	Facilitate VS	Perform VS		No VS	
Data Set	**PLTM**	**CVS**	**LFJ**	**GMM**	**MCLUST**
SYNTHETIC	**.81 (.04)**	.07 (.01)	.54 (.02)	.14 (.08)	.59 (.02)
IMAGE	**.71 (.04)**	.45 (.21)	.54 (.03)	.48 (.03)	.55 (.07)
SONAR	**.29 (.04)**	.02 (.02)	.03 (.03)	.00 (.00)	.03 (.07)
VEHICLE	**.39 (.04)**	.26 (.01)	.31 (.03)	.25 (.08)	.30 (.08)
YEAST	**.26 (.03)**	.10 (.07)	.17 (.04)	.17 (.02)	.12 (.02)
ZERNIKE	**.57 (.02)**	.34 (.06)	.48 (.03)	.48 (.06)	.40 (.07)
IONOSPHERE	.41 (.02)	.27 (.10)	.12 (.09)	.49 (.12)	.32 (.03)
WINE	.91 (.11)	.73 (.10)	.90 (.06)	.53 (.22)	.71 (.02)
IRIS	.76 (.00)	**.87 (.05)**	.68 (.04)	.73 (.09)	.76 (.00)
WDBC	.47 (.04)	.27 (.16)	.41 (.05)	.49 (.05)	**.60 (.07)**

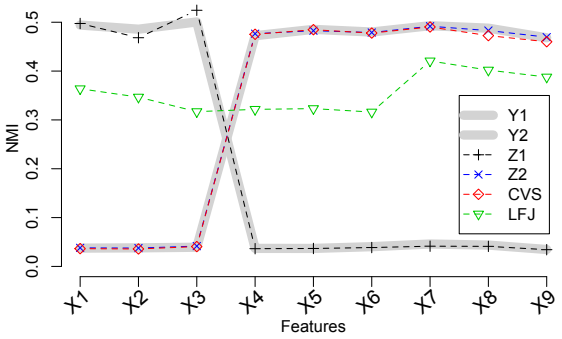


Figure 2. Feature curves of the partitions obtained by various methods and that of the original class partition on synthetic data.

the data has two facets, one represented by X_1–X_3 and another by X_4–X_9. The designated class partition Y_1 is a partition along the first facet.

The model produced by PLTM analysis have the same structure as the generative model. We name the two latent variables in the model Z_1 and Z_2 respectively. Their feature curves are also shown in Figure 2. We see that the feature curves of Z_1 and Z_2 match those of Y_1 and Y_2 well. This indicates that PLTM analysis has successfully recovered the two facets of the data. It has also produced a partition of the data along each of the facets. If the user chooses the partition Z_1 along the facet X_1–X_3 as the final result, then the original class partition is well recovered. This explains the good performance of PLTM (NMI=0.81).

The feature curves of the partitions obtained by LFJ and CVS are also shown in Figure 2. We see that the LFJ partition is not along any of the two natural facets of the data. Rather it is a partition based on a mixture of those two facets. Consequently, the performance of LFJ (NMI=0.54) is not as good as that of PLTM. CVS did identify the facet represented by X_4–X_9, but it is not the facet of the designated class partition. In other words, it picked the wrong facet. Consequently, the

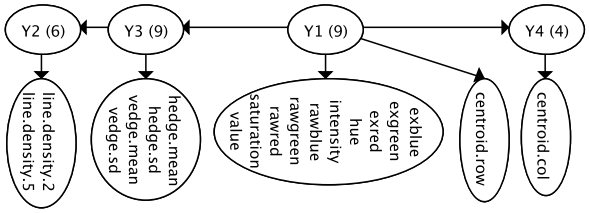


Figure 3. Structure of the PLTM learned from image data.

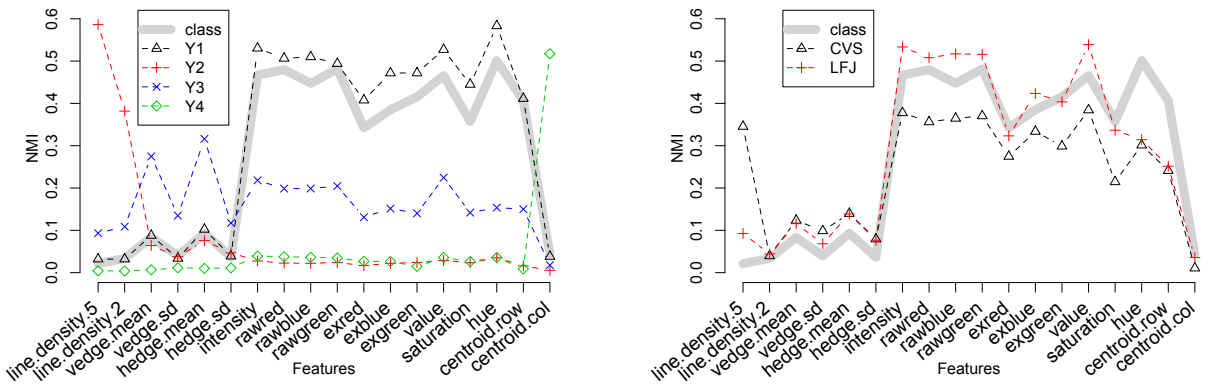
performance of CVS (NMI=0.07) is the worst among all the methods considered.

5.4.2. Models 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 4(a). 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 3. It contains 4 latent variables Y_1–Y_4. Their feature curves are shown in Figure 4(a). We see that the feature curve of Y_1 matches that of the class partition beautifully. If the user chooses 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).

The feature curves of the partitions obtained by LFJ and CVS are shown in Figure 4(b). 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 `line.density.5`, `hue` and `centroid.row`. Consequently, the performance of LFJ (NMI=0.54) is not as good as that of PLTM. Similar things can be said about the partition obtained by CVS. Its feature curve differs from the class feature curve even more than the LFJ curve on the attribute `line.density.5`, which is irrelevant to the class partition. Conse-



(a) PLTM analysis

(b) GMM-based methods

Figure 4. Feature curves of the partitions obtained by various methods and that of the original class partition on image data.

quently, the performance of CVS (NMI=0.45) is even worse than that of LFJ.

Two remarks are in order. 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 correlate 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 not only the edge attributes but others 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.

5.4.3. Discussions

We have also performed PLTM analysis on each of the the other data sets as a whole. The last column of Table 1 lists the numbers of latent variables obtained. We see that multiple latent variables have been identified except on iris data, which has only four attributes. Many of the latent variables represent partitions of data along natural facets of the data.

In general, PLTM analysis has the ability to identify natural facets of data and cluster data along those facets. In practice, a user may find several of the clusterings useful. In the setting where clustering algorithms are evaluated using labeled data, PLTM performs well if the original class partition is also along some of the natural facets, and poorly otherwise.

6. Related Works

There are some recent works that produce multiple clusterings. Caruana et al. (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. Cui et al. (2007) search for a collection of clustering solutions that are mutually orthogonal and collaboratively capture all the variance in data. Subspace clustering (e.g. Kriegel et al., 2009) tries to identify all clusters in all subspaces. 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.

7. Concluding Remarks

In this paper, we have proposed PLTMs as a generalization of GMMs and have empirically compared PLTM analysis with several GMM-based methods. Real-world high-dimensional data are usually multi-faceted and interesting clusterings in such data often are relevant only to subsets of attributes. One way to identify such clusterings is to perform variable selection. Another way is to perform PLTM analysis. Our work has shown that PLTM analysis is often more effective than the first approach.

One drawback of PLTM analysis is that the training is slow. For example, one fold of cross-validation took around 5 hours on data sets of moderate size (e.g., image, sonar, and wdbc data) and around 2.5 days on zernike data, the largest data set in our experiments. The learning of PLTMs can possibly be speeded up by parallelization or a better heuristic search. 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

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

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

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

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

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

Fraley, C. and Raftery, A. E. MCLUST version 3 for R: Normal mixture modeling and model-based clustering. Technical Report 504, Department of Statistics, University of Washington, 2006.

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

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

Kriegel, H.-P., Kröger, P., and Zimek, A. 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, 2009.

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

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

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

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

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

McLachlan, G. J. and Peel, D. *Finite Mixture Models*. Wiley, New York, 2000.

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

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

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

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

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

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

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