Mixed Deep Gaussian Mixture Model: a clustering model for mixed datasets

ADAC manuscript No.
(will be inserted by the editor)

Mixed Deep Gaussian Mixture Model: A clustering model for mixed datasets

Received: date / Accepted: date

Abstract Clustering mixed data presents numerous challenges inherent to the very heterogeneous nature of the variables. A clustering algorithm should be able, despite of this heterogeneity, to extract discriminant pieces of information from the variables in order to design groups. In this work we introduce a multi-layer architecture model-based clustering method called Mixed Deep Gaussian Mixture Model (MDGMM) that can be viewed as an automatic way to merge the clustering performed separately on continuous and non-continuous data. This architecture is flexible and can be adapted to mixed as well as to continuous or non-continuous data. In this sense we generalize Generalized Linear Latent Variable Models and Deep Gaussian Mixture Models. We also design a new initialisation strategy and a data driven method that selects the best specification of the model and the optimal number of clusters for a given dataset. Besides, our model provides continuous low-dimensional representations of the data which can be a useful tool to visualize mixed datasets. Finally, we validate the performance of our approach comparing its results with state-of-the-art mixed data clustering models over several commonly used datasets.

Keywords Binary and count data · Deep Gaussian Mixture Model · Generalized Linear Latent Variable Model · MCEM algorithm · Ordinal and categorical data · Two-heads architecture

1 Introduction

Mixed data consist of variables of heterogeneous nature that can be divided into two categories: the continuous data generated by real-valued random variables, and the non-continuous data which are composed of categorical data, ordinal data, binary data, and count data. By abuse of notation, these non-continuous variables will also be referred to as discrete variables in the following. Due to their different natures, mixed variables do not share common

scales and it is typically hard to measure the similarity between observations. There has been a significant and long interest in the statistical literature for mixed and continuous data clustering, which can be framed into four main categories, as described in Ahmad and Khan (2019): (i) partitional clustering minimizes the distance between observations and center groups by iterative optimization, as in K-modes or K-prototypes (Huang, 1997, 1998); (ii) hierarchical algorithms perform nested clusterings and merge them to create the final clustering (Philip and Ottaway, 1983; Chiu et al., 2001); (iii) model-based clustering (McLachlan and Peel, 2000; Fraley and Raftery, 2002; Melnykov et al., 2010), as their name suggests, rely on probability distributions; (iv) finally Neural Networks-based algorithms (Kohonen, 1990) design the clusters thanks to connected neurons that learn complex patterns contained in the data.

Within the framework of model-based clustering we propose a model for clustering mixed data, in which the different non-continuous variables are jointly modeled using a Generalized Linear Latent Variable Model (GLLVM) (Moustaki, 2003; Moustaki and Knott, 2000). GLLVMs assume that there exists a link function between the non-continuous observable space (composed of non-continuous variables) and a latent continuous data space, consisting of Gaussian latent variables. Recently, Cagnone and Viroli (2014) have extended this approach by considering latent variables that are no more Gaussian but follow some mixtures of Gaussians (Fraley and Raftery, 2002) so as the observations are naturally clustered into groups. Since the latent dimension is chosen to be strictly lower than the original dimension, the model also performs dimension reduction. For convenience, we will refer to this extended version when mentioning GLLVMs in the sequel.

Our work generalizes this idea by considering a Deep Gaussian Mixture Model (DGMM) (Viroli and McLachlan, 2019) in the latent space. DGMMs can be seen as a series of nested Mixtures of Factor Analyzers (MFA) (Ghahramani et al., 1996; McLachlan et al., 2003). As such, this approach performs clustering via subsequent dimensionally reduced latent spaces in a very flexible way. To adapt the GLLVM to mixed data we propose a multilayer architecture inspired by the idea that composing simple functions enables to capture complex patterns, as in supervised Neural Networks. We design two versions of our model. In the first one, denoted by M1DGMM, continuous and non-continuous data goes through the GLLVM model which acts as an embedding layer, *i.e.* projects the data into a simpler space before performing the clustering. The signal is then propagated to the following layers. In the second version, called M2DGMM, discrete data are still handled by the GLLVM model but continuous data are embedded separately by a DGMM head. The two signals are then merged by a "common tail". This second architecture is analogous to multi-inputs Supervised Deep Learning architectures used for instance when data are composed of both images and text.

Our model implementation relies on automatic differentiation (Baydin et al., 2017), as provided by the autograd package (Maclaurin et al., 2015), which

helps keeping an acceptable running time even when the number of layers increases.

To summarize, our work has three main contributions: it first extends the GLLVM and DGMM frameworks to deal with mixed data. Secondly, a new initialisation method is proposed to provide a suitable starting point for the MDGMM and more generally for GLLVM-based models. This initialization step combines Multiple Correspondence Analysis (MCA) or Factor Analysis of Mixed Data (FAMD) which generalizes it, GMM, MFA and the Partial Least Squares (PLS) algorithm. As mixed data are plunged into a multilayer continuous space we call this new initialisation Nested Spaces Embedding Procedure (NSEP). Thirdly, a model selection procedure is designed to identify the architecture of the model that best fits a given dataset.

Since the models are quite complex we propose to develop the method within the article and to reduce some mathematical developments by reporting them as Supplementary Material.

The paper is organized as follows: Section 2 provides a detailed description of the proposed model. In Section 3 the EM algorithms used for estimation are developed. Section 4 deals with the identifiability constraints of the model. Section 5 presents the initialization procedure NSEP and some practical considerations are given that can serve as a user manual. The performance of the model is compared to other competitor models in Section 6. In conclusion, Section 7 analyses the contributions of this work and highlights directions for future research.

2 Model presentation

2.1 The MDGMM as a generalization of existing models

In the sequel we assume that we observe n random variables $\mathbf{y}_1, \cdots, \mathbf{y}_n$, such that, $\forall i = 1, \cdots, n$, $\mathbf{y}_i = (\mathbf{y}_i^C, \mathbf{y}_i^D)$, where $\mathbf{y}_i^C$ is a p_C-dimensional vector of continuous random variables and $\mathbf{y}_i^D$ is a p_D-dimensional vector of non-continuous random variables. From what precedes, each $\mathbf{y}_i$ is hence a vector of mixed variables of dimension $p = p_C + p_D$.

The architecture of the MDGMM is based on two models. First, Mixtures of Factor Analyzers generalized by the Deep Gaussian Mixture Models are applied on continuous variables, and second, a Generalized Linear Latent Variable Model coupled with a DGMM is applied on non-continuous variables. Mixtures of Factor Analyzers represent the most elementary building block of our model and can be formulated as follows:

$$\mathbf{y}_i^C = \boldsymbol{\eta}_k + \boldsymbol{\Lambda}_k z_i + \mathbf{u}_{ik}, \text{ with probability } \pi_k,$$

where $k \in [1, K]$ identifies the group, $\boldsymbol{\eta}_k$ is a constant vector of dimension p_C, $\mathbf{z}_i \sim \mathrm{N}(0, \mathbf{I}_r)$ ($\mathbf{I}_r$ denoting the identity matrix), $\mathbf{u}_{ik} \sim \mathrm{N}(0, \boldsymbol{\Psi}_k)$ and $\boldsymbol{\Lambda}_k$ is the factor loading matrix of dimension $p_C \times r$, r being the dimension of the latent space. The underlying idea is to find a latent representation of the

data of lower dimension r, with $r < p_C$. For each group k, the loading matrix is then used to interpret the relationship existing between the data and their new representation.

The DGMM approach consists in extending the MFA model by assuming that $\mathbf{z}_i$ is no more drawn from a multivariate Gaussian but is itself a MFA. By repeating L times this hypothesis we obtain a L-layers DGMM defined by:

$$\begin{cases} \mathbf{y}_i^C = \boldsymbol{\eta}_{k_1}^{(1)} + \boldsymbol{\Lambda}_{k_1}^{(1)} \mathbf{z}_i^{(1)} + \mathbf{u}_{i,k_1}^{(1)}, \text{ with probability } \pi_{k_1}^{(1)} \\ \mathbf{z}_i^{(1)} = \boldsymbol{\eta}_{k_2}^{(2)} + \boldsymbol{\Lambda}_{k_2}^{(2)} \mathbf{z}_i^{(2)} + \mathbf{u}_{i,k_2}^{(2)}, \text{ with probability } \pi_{k_2}^{(2)} \\ \ldots \\ \mathbf{z}_i^{(L-1)} = \boldsymbol{\eta}_{k_L}^{(L)} + \boldsymbol{\Lambda}_{k_L}^{(L)} \mathbf{z}_i^{(L)} + \mathbf{u}_{i,k_L}^{(L)}, \text{ with probability } \pi_{k_L}^{(L)} \\ \mathbf{z}_i^{(L)} \sim \mathcal{N}(0, \mathbf{I}_{r_L}), \end{cases} \tag{1}$$

where, for $\ell = 1, \cdots, L$, $k_\ell \in [1, K_\ell]$, $\mathbf{u}_{ik_\ell}^{(\ell)} \sim N(0, \boldsymbol{\Psi}_{k_\ell}^{(\ell)})$, $\mathbf{z}_i^{(L)} \sim \mathrm{N}(0, \mathbf{I}_{r_L})$ and where the factor loading matrices $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)}$ have dimension $r_{\ell-1} \times r_\ell$, with the constraint $p > r_1 > r_2 > ... > r_L$. Identifiability constraints on the parameters $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)}$ and $\boldsymbol{\Psi}_{k_\ell}^{(\ell)}$ will be discussed in Section 4.

The DGMM described in (1) can only handle continuous data. In order to apply a DGMM to discrete data we propose to integrate a Generalized Linear Latent Variable Model (GLLVM) framework within (1). This new integrated model will be called Discrete DGMM (DDGMM).

A GLLVM assumes that, $\forall j \in [1, p_D]$, the discrete random variables $\mathbf{y}_j^D$ are associated to one (or more) continuous latent variable through an exponential family link (see the illustrations given in Cagnone and Viroli, 2014), under the so-called *conditional independence assumption*, according to which variables are mutually independent conditionally to the latent variables.

Hence, one can combine the previously introduced DGMM architecture and the GLLVM to deal with mixed data. In order to do so, we propose two specifications of the MDGMM: a one head version (the M1DGMM) and a two heads version (the M2DGMM). In the M1DGMM, the continuous variables pass through the GLLVM layer by defining a link function between $\mathbf{y}^C$ and $\mathbf{z}^{(1)}$ and one assumes that the *conditional independence assumption* evoked earlier holds. On the contrary, for the M2DGMM by specifying a second head to deal with the continuous data specifically, one can relax this assumption: the continuous variables are not assumed to be mutually independent with respect to the latent variables. Instead, each continuous variable is only conditionally independent from the discrete variables but not from the other continuous variables. The two-heads architecture is also more flexible than the one-head specification as its "link function" between the continuous data and the latent variables is a mixture of mixture rather than a probability distribution belonging to an exponential family. This flexibility comes at the price of additional model complexity and computational costs which has to be evaluated in regard of the performances of each specification.

2.2 Formal definition

Let $\mathbf{y}$ be the $n \times p$ matrix of the observed variables. We will denote by $i \in [1, n]$ the observation index and by $j \in [1, p]$ the variable index. We can decompose the data as $\mathbf{y} = (\mathbf{y}^C, \mathbf{y}^D)$ where $\mathbf{y}^C$ is the $n \times p_C$ matrix of continuous variables and $\mathbf{y}^D$ is the $n \times p_D$ matrix of discrete variables. The global architecture of the M2DGMM is analogous to (1) with an additional GLLVM step for the discrete head as follows:

$$
\begin{aligned}
\text{Discrete head}: & \begin{cases}
\mathbf{y}_i^D \to \mathbf{z}_i^{(1)D} \text{ through GLLVM link via } (\boldsymbol{\lambda}^{(0)}, \boldsymbol{\Lambda}^{(0)}) \\
\mathbf{z}_i^{(1)D} = \boldsymbol{\eta}_{k_1}^{(1)D} + \boldsymbol{\Lambda}_{k_1}^{(1)D} \mathbf{z}_i^{(2)D} + \mathbf{u}_{i,k_1}^{(1)D} \text{ with probability } \pi_{k_1}^{(1)D} \\
\ldots \\
\mathbf{z}_i^{(L_D)D} = \boldsymbol{\eta}_{k_{L_D}}^{(L_D)D} + \boldsymbol{\Lambda}_{k_{L_D}}^{(L_D)D} \mathbf{z}_i^{(L_D+1)} + \mathbf{u}_{i,k_{L_D}}^{(L_D)D}, \text{ with probability } \pi_{k_{L_D}}^{(L_D)D}
\end{cases} \\
\text{Continuous head}: & \begin{cases}
\mathbf{y}_i^C = \boldsymbol{\eta}_{k_1}^{(1)C} + \boldsymbol{\Lambda}_{k_1}^{(1)C} \mathbf{z}_i^{(1)C} + \mathbf{u}_{i,k_1}^{(1)C} \text{ with probability } \pi_{k_1}^{(1)C} \\
\mathbf{z}_i^{(1)C} = \boldsymbol{\eta}_{k_2}^{(2)C} + \boldsymbol{\Lambda}_{k_2}^{(2)C} \mathbf{z}_i^{(2)C} + \mathbf{u}_{i,k_2}^{(2)C} \text{ with probability } \pi_{k_2}^{(2)C} \\
\ldots \\
\mathbf{z}_i^{(L_C)C} = \boldsymbol{\eta}_{k_{L_C+1}}^{(L_C+1)C} + \boldsymbol{\Lambda}_{k_{L_C+1}}^{(L_C+1)C} \mathbf{z}_i^{(L_C+1)} + \mathbf{u}_{i,k_{L_C+1}}^{(L_C+1)C}, \text{ with probability } \pi_{k_{L_C+1}}^{(L_C+1)C}
\end{cases} \\
\text{Common tail}: & \begin{cases}
\mathbf{z}_i^{(L_0+1)} = \boldsymbol{\eta}_{k_{L_0+1}}^{(L_0+1)} + \boldsymbol{\Lambda}_{k_{L_0+1}}^{(L_0+1)} \mathbf{z}_i^{(L_0+2)} + \mathbf{u}_{i,k_{L_0+1}}^{(L_0+1)}, \text{ with probability } \pi_{k_{L_0+2}}^{(L_0+2)} \\
\ldots \\
\mathbf{z}_i^{(L-1)} = \boldsymbol{\eta}_{k_{L-1}}^{(L-1)} + \boldsymbol{\Lambda}_{k_{L-1}}^{(L-1)} \mathbf{z}_i^{(L)} + \mathbf{u}_{i,k_{L-1}}^{(L-1)} \text{ with probability } \pi_{k_L}^{(L)} \\
\mathbf{z}_i^{(L)} \sim \mathcal{N}(0, \boldsymbol{I}_{r_L}).
\end{cases}
\end{aligned}
\tag{2}
$$

It is assumed that the random variables $(u_{k_\ell}^{(\ell)})_{k_\ell, \ell}$ are all independent. The two heads only differ from each other by the fact that for the discrete head, a continuous representation of the data has first to be determined before information is fed through the layers. The GLLVM layer is parametrized by $(\boldsymbol{\lambda}_0, \boldsymbol{\Lambda}_0)$. $\boldsymbol{\lambda}_0 = (\lambda_{0bin}, \lambda_{0count}, \lambda_{0ord}, \lambda_{0categ})$ contains the intercept coefficients for each discrete data sub-type. $\boldsymbol{\Lambda}_0$ is a matrix of size $p_D \times r_1$, with r_1 the dimension of the first Discrete DGMM layer.

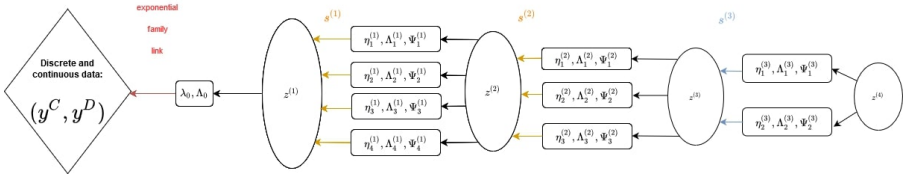
The notation remains the same as in the previous subsection and only a superscript is added to specify for each variable the head or tail to which it belongs. For instance $\mathbf{z}^C = (\mathbf{z}^{(1)C}, ..., \mathbf{z}^{(L_C)C})$ is the set of latent variables of the continuous head. This subscript is omitted for the common head. The ℓ-th layer of the head h contains K_ℓ^h components which is the number of components of the associated mixture. L_D and L_C are the number of layers of the discrete and continuous head, respectively.

Each path from one layer to the next is the realization of a mixture. In this sense we introduce, $s^{(\ell)h} \in [1, K_\ell^h]$ the latent variable associated with the index of the component k_ℓ^h of the layer ℓ of the head h. More generally, the latent variable associated with a path going from the first layer to the last layer of one head h is denoted by $s^h = (s^{(1)h}, ..., s^{(L_0)h})$. Similarly, the random variable associated to a path going through all the common tail layers is denoted by

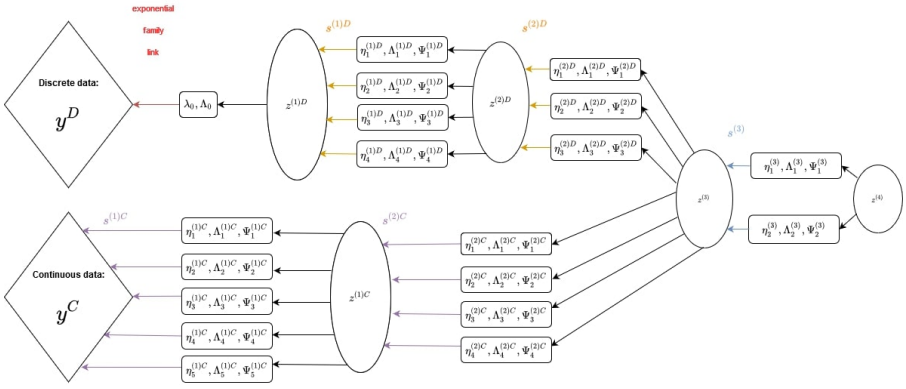
$s^{(L_0+1:)} = (s^{(L_0+1)}, ..., s^{(L)})$. By extension, the variable associated with a full path going from the beginning of head h to the end of the common tail is $s^{(1h:L)} = (s^h, s^{L_0+1:})$. $s^{(1h:L)}$ belongs to Ω^h the set of all possible paths starting from one head of cardinal $S^h = \prod_{\ell=1}^{L} K_\ell^h$. The variable associated with a path going from layer ℓ of head h to layer L will be denoted $s^{(\ell h:L)}$. Finally, by a slight abuse of notation a full path going through the component k_ℓ^h of the ℓ-th layer of head h will be denoted: $s^{(1:k_\ell^h:L)}$ or more simply $s^{(:k_\ell^h:)}$. In order to be as concise as possible, we group the parameters of the model by defining:

$$\boldsymbol{\Theta}_D = \left((\boldsymbol{\lambda}_0, \boldsymbol{\Lambda}_0), (\boldsymbol{\eta}_{k_\ell}^{(\ell)D}, \boldsymbol{\Lambda}_{k_\ell}^{(\ell)D}, \boldsymbol{\Psi}_{k_\ell}^{(\ell)D})_{k_\ell \in [1,K_\ell^D], \ell \in [1,L_0]}\right),$$

$$\boldsymbol{\Theta}_C = (\boldsymbol{\eta}_{k_\ell}^{(\ell)C}, \boldsymbol{\Lambda}_{k_\ell}^{(\ell)C}, \boldsymbol{\Psi}_{k_\ell}^{(\ell)C})_{k_\ell \in [1,K_\ell^C], \ell \in [1,L_0]}, \qquad \boldsymbol{\Theta}_{L_0+1:} = (\boldsymbol{\eta}_{k_\ell}^{(\ell)}, \boldsymbol{\Lambda}_{k_\ell}^{(\ell)}, \boldsymbol{\Psi}_{k_\ell}^{(\ell)})_{k_\ell \in [1,K_\ell], \ell \in [L_0+1,L]},$$



(a) M1DGMM



(b) M2DGMM

Fig. 1 Graphical model of: (a) M1DGMM, (b) M2DGMM

The architecture of the M1DGMM is the same except that there is no "continuous head" and that the y_i^C goes through the GLLVM link. Figure 1 presents the graphical models associated with both specifications. The M2DGMM illustrated here has $K_C = (5, 4)$, $K_D = (4, 3)$, $K = (2, 1)$, $L_C = L_D = L_0 = 2$, $S^C = 40$ and $S^D = 24$. The decreasing size of the $(\mathbf{z}^{(\ell)})_\ell$ illustrates the decreasing dimensions of the latent variables.
One can observe that $L_0 = \max(L_C, L_D)$, that is, the first layer of the common

tail is the L_0+1-th layers of the model. For simplicity of notation, we assume in the sequel that

$$L_C = L_D = L_0,$$

but all the results are easily obtained in the general case.
The intuition behind the M2DGMM architecture is simple. The two heads extract features from the data and pass them to the common tail. The tail reconciles both information sources, designs common features and performs the clustering. As such, any layer on the tail could in principle be used as clustering layer. As detailed in Section 5.2, one could even use several tail layers to perform several clustering procedures (with different latent dimensions or numbers of clusters) in the same model run. The same remarks applies for the hidden layers of the M1DGMM.

To summarize the different setups that can be handled by DGMM-based models:

- Use the M1DGMM or the M2DGMM when data are mixed,
- Use the DDGMM when data are non-continuous,
- Use the DGMM when data are continuous.

3 Model estimation

In this section we present the estimation details for the M2DGMM, the M1DGMM may be handled in much the same way. The complete density of the M2DGMM is given by:

$$\begin{aligned}
&L(\mathbf{y}^C,\mathbf{y}^D,\mathbf{z}^C,\mathbf{z}^D,\mathbf{z}^{(L_0+1:)},s^C,s^D,s^{(L_0+1:)}|\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:})\\
&\quad= L(y^C|\mathbf{z}^{(1)C},s^C,s^{(L_0+1:)},\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_{L_0+1:})L(\mathbf{z}^C|\mathbf{z}^{(L_0+1:)},s^C,s^{(L_0+1:)},\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_{L_0+1:})\\
&\quad\times L(y^D|\mathbf{z}^{(1)D},s^D,s^{(L_0+1:)},\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:})L(\mathbf{z}^D|\mathbf{z}^{(L_0+1:)},s^D,s^{(L_0+1:)},\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:})\\
&\quad\times L(\mathbf{z}^{(L_0+1:)}|s^C,s^D,s^{(L_0+1:)},\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:})L(s^C,s^D,s^{(L_0+1:)}|\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:}),
\end{aligned}$$

which comes from the fact that we assume the two heads of the model to be conditionally independent with respect to the tail layers. Moreover, the layers of both heads and tail share the Markov property derived from the graphical model: $(\mathbf{z}^{(\ell)h} \perp\!\!\!\perp \mathbf{z}^{(\ell+2)h},...,\mathbf{z}^{(L)h})\Big|\mathbf{z}^{(l+1)h}$, $\forall h \in \{C,D,(L_0+1:)\}$.

As the model involves latent variables, one cannot maximise directly the likelihood of the model to estimate the parameters of the model. Hence, the expected log-likelihood with respect to these latent variables and given the observed data is instead maximised using the EM-algorithm:

$$\mathbb{E}_{\mathbf{z}^C,\mathbf{z}^D,\mathbf{z}^{(L_0+1:)},s^C,s^D,s^{(L_0+1:)}|\mathbf{y}^C,\mathbf{y}^D,\hat{\boldsymbol{\Theta}}_C,\hat{\boldsymbol{\Theta}}_D,\hat{\boldsymbol{\Theta}}_{L_0+1:}}[\log L(\mathbf{y}^C,\mathbf{y}^D,\mathbf{z}^C,\mathbf{z}^D,\mathbf{z}^{(L_0+1:)},s^C,s^D,s^{(L_0+1:)}|\boldsymbol{\Theta}_C,\boldsymbol{\Theta}_D,\boldsymbol{\Theta}_{L_0+1:})],$$

As in deep mixture models (Viroli and McLachlan, 2019), some of the terms of the expected log-likelihood contain intractable expectations and a Monte Carlo extension of the EM algorithm (MCEM) (Wei and Tanner, 1990) is required

to overcome this issue. According to the decomposition of the complete log-likelihood into the sum of different terms, three types of layers are involved in the estimation process: the GLLVM layer, the regular DGMM layers and the common tail layers that join the two heads. In the following sections, we will present the detailed estimation steps for the three types of layers.

3.1 Generalized Linear Latent Variable Model Embedding Layer

In this section we present the canonical framework of GLLVMs for discrete data based on Moustaki (2003) and Moustaki and Knott (2000).
By the conditional independence assumption between discrete variables, the likelihood can be written as:

$$f(\mathbf{y}^D|\boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:}) = \int_{\mathbf{z}^{(1)D}} \prod_{j=1}^{p_D} f(y_j^D|\mathbf{z}^{(1)D}, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:}) f(\mathbf{z}^{(1)D}|\boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:}) d\mathbf{z}^{(1)D}, \tag{3}$$

where y_j^D is the jth component of $\mathbf{y}^D$. The density $f(y_j^D|\mathbf{z}^{(1)D}, \Theta_D, \Theta_{L_0+1:})$ belongs to an exponential family and in our empirical study we chose a Bernoulli distribution for binary variables, a binomial distribution for count variables, a multinomial distribution for categorical data and an ordered multinomial distribution for ordinal data. It is worth noting that the categorical variables are treated as such and not converted into binomial variables beforehand.
The whole expressions of the densities can be found in Cagnone and Viroli (2014). In order to train the GLLVM layer, we maximize

$$\begin{aligned}
&\mathbb{E}_{\mathbf{z}^{(1)D}, s^D, s^{(L_0+1:)}|\mathbf{y}^D, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:}}[\log L(\mathbf{y}^D|\mathbf{z}^{(1)D}, s^D, s^{L_0+1:}, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:})] \\
&= \mathbb{E}_{\mathbf{z}^{(1)D}|\mathbf{y}^D, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:}}[\log L(\mathbf{y}^D|\mathbf{z}^{(1)D}, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:})] \\
&= \int f(\mathbf{z}^{(1)D}|\mathbf{y}^D, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:}) \log L(\mathbf{y}^D|\mathbf{z}^{(1)D}, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:}) d\mathbf{z}^{(1)D},
\end{aligned} \tag{4}$$

the second equality being due to the fact that $\mathbf{y}^D$ is related to $(s^D, s^{(L_0+1:)})$ only through $\mathbf{z}^{(1)D}$.

3.1.1 MC and E Steps

Expectation (4) is not tractable as $f(\mathbf{z}^{(1)D}|s^D, s^{(L_0+1:)}, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})$ involves terms that are not directly computable. Hence, in order to compute and maximise (4), draws $f(\mathbf{z}^{(1)D}|s^D, s^{(L_0+1:)}, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})$ are performed using Monte Carlo methods. All details are given in the Supplementary Material.

3.1.2 M step

Even if the distribution of expectation (4) can be approximated, there are no closed-form solutions for the estimators of $(\boldsymbol{\lambda}_0, \boldsymbol{\Lambda}_0)$ that maximize

$$\mathbb{E}_{\mathbf{z}^{(1)D}|\mathbf{y}^D,\hat{\boldsymbol{\Theta}}_D,\hat{\boldsymbol{\Theta}}_{L_0+1:}}[\log L(\mathbf{y}^D|\mathbf{z}^{(1)D}, \boldsymbol{\Theta}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})].$$

Optimisation methods are therefore used to determine the optimal $(\boldsymbol{\lambda}_0, \boldsymbol{\Lambda}_0)$ as detailed in Supplementary Material.

3.2 Determining the parameters of the DGMM layers

In this section, we omit the subscript $h \in \{C, D\}$ on the $\mathbf{z}^h$, $\mathbf{y}^h$ and s^h variables because the reasoning is the same for both cases. For $\ell \in [1, L_0]$, we aim to maximize

$$\mathbb{E}_{\mathbf{z}^{(\ell)},\mathbf{z}^{(\ell+1)},s|\mathbf{y},\hat{\boldsymbol{\Theta}}}[\log L(\mathbf{z}^{(\ell)}|\mathbf{z}^{(\ell+1)}, s, \boldsymbol{\Theta})].$$

Here the conditional distribution under which the expectation is taken depends on variables located in 3 different layers.

3.2.1 MC Step

At each layer ℓ, $M^{(\ell)}$ pseudo-observations are drawn for each of the previously obtained $\prod_{j=1}^{\ell-1} M^{(j)}$ pseudo-observations. Hence, in order to draw from $f(\mathbf{z}^{(\ell)}, \mathbf{z}^{(\ell+1)}, s|\mathbf{y}, \hat{\boldsymbol{\Theta}})$ at layer ℓ:

- If $\ell = 1$, reuse the $M^{(1)}$ pseudo-observations drawn from $f(\mathbf{z}^{(1)}|s, \hat{\boldsymbol{\Theta}})$,
- otherwise reuse the $M^{(\ell-1)}$ pseudo-observations from $f(\mathbf{z}^{(\ell-1)}|\mathbf{y}, s, \hat{\boldsymbol{\Theta}})$ and the $M^{(\ell)}$ pseudo-observations from $f(\mathbf{z}^{(\ell)}|\mathbf{z}^{(\ell-1)}, s, \hat{\boldsymbol{\Theta}})$ computed for each pseudo-observation of the previous DGMM layer.
- Draw $M^{(\ell+1)}$ observations from $f(\mathbf{z}^{(\ell+1)}|\mathbf{z}^{(\ell)}, s, \hat{\boldsymbol{\Theta}})$ for each previously sampled $\mathbf{z}^{(\ell)}$.

3.2.2 E Step

The conditional expectation distribution has the following decomposition:

$$\begin{aligned} f(\mathbf{z}^{(\ell)}, \mathbf{z}^{(\ell+1)}, s|\mathbf{y}, \hat{\boldsymbol{\Theta}}) &= f(\mathbf{z}^{(\ell)}, s|\mathbf{y}, \hat{\boldsymbol{\Theta}}) f(\mathbf{z}^{(\ell+1)}|\mathbf{z}^{(\ell)}, s, \mathbf{y}, \hat{\boldsymbol{\Theta}}) \\ &= f(\mathbf{z}^{(\ell)}|\mathbf{y}, s, \hat{\boldsymbol{\Theta}}) f(s|\mathbf{y}, \hat{\boldsymbol{\Theta}}) f(\mathbf{z}^{(\ell+1)}|\mathbf{z}^{(\ell)}, s, \hat{\boldsymbol{\Theta}}), \end{aligned} \tag{5}$$

and is developed in the Supplementary Material.

3.2.3 M step

The estimators of the DGMM layer parameters $\forall \ell \in [1, L_0]$ are given by:

$$\begin{cases} \hat{\boldsymbol{\eta}}_{k_\ell}^{(\ell)} = \dfrac{\sum_{i=1}^{n} \sum_{\tilde{s}_i^{(:k_\ell:)}} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}, \hat{\boldsymbol{\Theta}}) \left[E[\mathbf{z}_i^{(\ell)} | s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)}, \mathbf{y}_i, \hat{\boldsymbol{\Theta}}] - \boldsymbol{\Lambda}_{k_\ell}^{(\ell)} E[\mathbf{z}_i^{(\ell+1)} | \tilde{s}_i^{(:k_\ell:)}, \mathbf{y}_i, \hat{\boldsymbol{\Theta}}] \right]}{\sum_{i=1}^{n} \sum_{\tilde{s}_i^{(:k_\ell:)}} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}_i, \hat{\boldsymbol{\Theta}})} \\ \hat{\boldsymbol{\Lambda}}_{k_\ell}^{(\ell)} = \dfrac{\sum_{i=1}^{n} \sum_{\tilde{s}_i^{(:k_\ell:)}} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}_i, \hat{\boldsymbol{\Theta}}) \left[E[(\mathbf{z}_i^{(\ell)} - \hat{\boldsymbol{\eta}}_{k_\ell}^{(\ell)}) \mathbf{z}_i^{(\ell+1)T} | s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)}, \mathbf{y}_i, \hat{\boldsymbol{\Theta}}] \right]}{\sum_{i=1}^{n} \sum_{\tilde{s}_i^{(:k_\ell:)}} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}_i, \hat{\boldsymbol{\Theta}})} E[\mathbf{z}_i^{(\ell+1)} \mathbf{z}_i^{(\ell+1)T} | \tilde{s}_i^{(:k_\ell:)}, \mathbf{y}_i, \hat{\boldsymbol{\Theta}}]^{-1} \\ \hat{\boldsymbol{\Psi}}_{k_\ell}^{(\ell)} = \dfrac{\sum_{i=1}^{n} \sum_{\tilde{s}_i} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}_i, \hat{\boldsymbol{\Theta}}) E\left[\left(\mathbf{z}_i^{(\ell)} - (\eta_{k_l}^{(\ell)} + \boldsymbol{\Lambda}_{k_l}^{(\ell)} \mathbf{z}_i^{(\ell+1)}) \right) \left(\mathbf{z}_i^{(\ell)} - (\boldsymbol{\eta}_{k_l}^{(\ell)} + \boldsymbol{\Lambda}_{k_l}^{(\ell)} \mathbf{z}_i^{(\ell+1)}) \right)^T | \tilde{s}_i^{(:k_\ell:)}, \mathbf{y}_i, \hat{\boldsymbol{\Theta}} \right]}{\sum_{i=1}^{n} \sum_{\tilde{s}_i^{(:k_\ell:)}} f(s_i^{(:k_\ell:)} = \tilde{s}_i^{(:k_\ell:)} | \mathbf{y}_i, \hat{\boldsymbol{\Theta}})}, \end{cases}$$

with $\tilde{s}_i^{(:k_\ell:)} = (\tilde{k}_1, ..., \tilde{k}_{\ell-1}, k_\ell, \tilde{k}_{\ell+1}, ..., \tilde{k}_L)$, a path going through the network and reaching the component k_ℓ. The details of the computation are given in the Supplementary Material.

3.3 Training of the common tail layers

In this section we aim to maximise $\forall \ell \in [L_0 + 1, L]$, the following expression:

$$\mathbb{E}_{\mathbf{z}^{(\ell)}, \mathbf{z}^{(\ell+1)}, s^C, s^D, s^{(L_0+1:)} | \mathbf{y}^C, \mathbf{y}^D, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:}} [\log L(\mathbf{z}^{(\ell)} | \mathbf{z}^{(\ell+1)}, s^C, s^D, s^{(L_0+1:)}, \boldsymbol{\Theta}_C, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:})].$$

3.3.1 MC Step

The MC step remains the same as for regular DGMM layers except that the conditioning concerns both types of data ($\mathbf{y}^C$ and $\mathbf{y}^D$) and not only discrete or continuous data as in the heads layers.

3.3.2 E Step and M step

The distribution of the conditional expectation is $f(\mathbf{z}^{(\ell)}, \mathbf{z}^{(\ell+1)}, s^C, s^D, s^{(L_0+1:)} | \mathbf{y}^C, \mathbf{y}^D, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})$ that we can express as previously (see details in the Supplementary Material).

In addition, the estimators of the junction layers keep the same form as the regular DGMM layers except once again that the two types of data and paths exist in the conditional distributions of the expectations.

3.4 Determining the path probabilities

In this section, we determine the path probabilities by optimizing the parameters of the following expression derived from the expected log-likelihood:

$$\mathbb{E}_{s^C, s^D, s^{(L_0+1:)} | \mathbf{y}^C, \mathbf{y}^D, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:}} [\log L(s^C, s^D, s^{(L_0+1:)} | \boldsymbol{\Theta}_C, \boldsymbol{\Theta}_D, \boldsymbol{\Theta}_{L_0+1:})],$$

with respect to π_s^h, for $h \in \{C, D\}$ and $\pi_s^{(L_0+1:)}$.

3.4.1 E step and M step

By mutual independence of s^C, s^D and $s^{L_0+1:}$, estimating the distribution of the expectation boils down to computing three densities: $f(s^{(\ell)D} = k_\ell|\mathbf{y}^D, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})$, $f(s^{(\ell)C} = k_\ell|\mathbf{y}^C, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_{L_0+1:})$, and $f(s^{(\ell)} = k_\ell|\mathbf{y}^C, \mathbf{y}^D, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_D, \hat{\Theta}_{L_0+1:})$ (details are given in the Supplementary Material).

Finally, estimators for each head h and for the common tail are given respectively by:

$$\hat{\pi}_{k_\ell}^{(\ell)h} = \frac{\sum_{i=1}^n f(s_i^{(\ell)h} = k_\ell|\mathbf{y}_i^h, \hat{\boldsymbol{\Theta}}_h, \hat{\boldsymbol{\Theta}}_{L_0+1:})}{n} \quad \text{and} \quad \hat{\pi}_{k_\ell}^{(\ell)} = \frac{\sum_{i=1}^n f(s_i^{(\ell)} = k_\ell|\mathbf{y}_i^C, \mathbf{y}_i^D, \hat{\boldsymbol{\Theta}}_C, \hat{\boldsymbol{\Theta}}_D, \hat{\boldsymbol{\Theta}}_{L_0+1:})}{n}.$$

4 Identifiability

In this section, we combine both GLLVM and DGMM identifiability constraints proposed in Cagnone and Viroli (2014) and Viroli and McLachlan (2019), respectively, to make our model identifiable.

4.1 GLLVM identifiability constraints

Both the GLLVM model and the Factor Analysis model assume that the latent variables are centered and of unit variance. This can be obtained by rescaling iteratively all the latent layers parameters from the last common layer to the first head layers as follows:

$$\begin{cases} \boldsymbol{\eta}_{k_\ell}^{(\ell)hnew} = (\mathbf{A}^{(\ell)h})^{-1T}\left[\boldsymbol{\eta}_{k_\ell}^{(\ell)h} - \sum_{k'_\ell} \pi_{k'_\ell}^{(\ell)h} \boldsymbol{\eta}_{k'_\ell}^{(\ell)h}\right] \\ \boldsymbol{\Lambda}_{k_\ell}^{(\ell)hnew} = (\mathbf{A}^{(\ell)h})^{-1T} \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h} \\ \boldsymbol{\Psi}_{k_\ell}^{(\ell)new} = (\mathbf{A}^{(\ell)h})^{-1T} \boldsymbol{\Psi}_{k_\ell}^{(\ell)h} (\mathbf{A}^{(\ell)h})^{-1}, \end{cases}$$

where $\mathbf{A}^{(\ell)h} = Var(\mathbf{z}^{(\ell)h})\ \forall \ell \in [1, L], h \in \{C, D, L_0 + 1 :\}$ and the subscript "new" denotes the rescaled version of the parameters. In the same way, the coefficients of $\boldsymbol{\Lambda}^{(0)}$ of the discrete head are rescaled as follows: $\boldsymbol{\Lambda}^{(0)new} = \boldsymbol{\Lambda}^{(0)}\mathbf{A}^{-1T}$. As in Cagnone and Viroli (2014), some coefficients of $\boldsymbol{\Lambda}^{(0)}$ are constrained to be zero in order to reduce the total number of degrees of freedom of the model. The details of this section are given in the Supplementary Material.

4.2 DGMM identifiability constraints

We assume first that the latent dimension is decreasing through the layers of each head and tail *i.e.* $p_h > r_1^h > ... > r_L$. Secondly, we make the assumption that $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)hT} \boldsymbol{\Psi}_{k_\ell}^{(\ell)-1h} \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h}$ is diagonal with elements in decreasing order

$\forall \ell \in [1, L]$. Fruehwirth-Schnatter and Lopes (2018) obtained sufficient conditions for MFA identifiability, including the so-called *Anderson-Rubin* (AR) condition, which requires that $r_\ell \leq \frac{r_{\ell-1}-1}{2}$. Enforcing this condition would prevent from defining a MDGMM for all datasets that present less than 7 variables of each type which is far too restrictive. Then, we implement a transformation to ensure the diagonality and the ordering of the coefficients of $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)hT} \boldsymbol{\Psi}_{k_\ell}^{(\ell)-1h} \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h}$ as follows:
Once all parameters have been estimated by the MCEM algorithm, the following transformation is applied over $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)h}$:

- Compute $\mathbf{B} = \boldsymbol{\Lambda}_{k_\ell}^{(\ell)hT} \boldsymbol{\Psi}_{k_\ell}^{(\ell)-1h} \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h}$.
- Decompose $\mathbf{B}$ according to the eigendecomposition $\mathbf{B} = \mathbf{PDP}^{-1}$, with $\mathbf{D}$ the matrix of the eigenvalues and $\mathbf{P}$ the matrix of eigenvectors.
- Define $\boldsymbol{\Lambda}_{k_\ell}^{(\ell)hnew} = \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h} P$.

5 Practical considerations

5.1 Initialisation procedure

EM-based algorithms are known to be very sensitive to their initialisation values as shown for instance by Biernacki et al. (2003) for Gaussian Mixture models. In our case, using purely random initialization as in Cagnone and Viroli (2014) made the model diverge most of the time when the latent space was of high dimension. This can be explained by the fact that the clustering is performed in a projected continuous space of which one has no prior knowledge about. Initialising at random the latent variables $(\boldsymbol{\eta}_{k_\ell}^{(\ell)h}, \boldsymbol{\Lambda}_{k_\ell}^{(\ell)h}, \boldsymbol{\Psi}_{k_\ell}^{(\ell)h}, s^{(\ell)h}, \mathbf{z}^{(\ell)h})_{k_\ell,\ell,h}$ and the exponential family links parameters $(\boldsymbol{\lambda}^{(0)}, \boldsymbol{\Lambda}^{(0)})$ seems not to be a good practice. This problem gets even worse as the number of DGMM layers grows. To stabilize our algorithm we propose the NSEP approach which combines MCA, GMM, FA and PLS algorithm in the M2DGMM case as follows:

- For discrete head initialisation, we perform a Multiple Correspondence Analysis (MCA) (Nenadic and Greenacre, 2005) to determine a continuous low dimensional representation of the discrete data and use it as a first approximation of the latent variables $\mathbf{z}^{(1)D}$. The MCA considers all variables as categorical, thus the more the dataset actually contains this type of variables the better the initialisation should in theory be. Once this is done, a Gaussian Mixture Model is fitted in order to determine groups in the continuous space and to estimate $(\pi_{k_\ell}^{(\ell)})$. For each group a Factor Analysis Model (FA) is fitted to determine the parameters of the model $(\boldsymbol{\eta}_{k_\ell}^{(\ell)}, \boldsymbol{\Lambda}_{k_\ell}^{(\ell)}, \boldsymbol{\Psi}_{k_\ell}^{(\ell)})$ and the latent variable of the following layer $\mathbf{z}^{(\ell+1)}$. Concerning the GLLVM parameters, logistic regressions of $\mathbf{y}_j^D$ over $\mathbf{z}^{(1)D}$ are fitted for each original variable of the discrete head: ordered logistic

regressions for ordinal variables and unordered logistic regressions for binary, count and categorical variables.

– For the continuous head and the common tail, the same described GMM coupled with FA procedure can be applied to determine the coefficients of the layer. The difficulty concerns the initialisation of the first tail layer with latent variable $\mathbf{z}^{(L_0+1)}$. Indeed, $\mathbf{z}^{(L_0+1)}$ has to be the same for both discrete and continuous last layers. As Factor Models are unsupervised models, one cannot enforce such a constraint on the latent variable generated from each head. To overcome this difficulty, $\mathbf{z}^{(L_0+1)}$ has been determined by applying a PCA over the stacked variables $(\mathbf{z}^{(L_0)C}, \mathbf{z}^{(L_0)D})$. Then the DGMM coefficients $(\boldsymbol{\eta}_{k_{L_0}}^{(L_0)h}, \boldsymbol{\Lambda}_{k_{L_0}}^{(L_0)h}, \boldsymbol{\Psi}_{k_{L_0}}^{(L_0)}h)$ of each head have been separately determined using Partial Least Square (Wold et al., 2001) of each head last latent variable over $\mathbf{z}^{(L_0+1)}$.

The same ideas are used to initialize the M1DGMM. As the data going through the unique head of the M1DGMM are mixed, Factor Analysis of Mixed Data (FAMD) (Pagès, 2014) is employed instead of MCA as it can handle mixed data.

5.2 Model and number of clusters selection

The selection of the best MDGMM architecture is performed using the pruning methodology which is widely used in the field of supervised Neural Networks (Blalock et al., 2020) but also for tree-based methods (Patil et al., 2010).

Classical approaches to model specification based on information criteria, such as AIC (Akaike, 1998) or BIC (Schwarz et al., 1978), need the estimation of all the possible specifications of the model. In contrast, our approach needs only one model run to determine the best architecture which is far more computationally efficient.

In the following, we give a summary of our pruning strategy (extensive details are provided in the Supplementary Material). The idea is to determine the best number of components on each layer k_ℓ^h by deleting the components associated with very low probabilities $\pi_{k_\ell}^{(\ell)h}$ as they are the least likely to explain the data. The choice of the latent dimensions of each layer r_ℓ^h is performed by looking at the dimensions that carry the most important pieces of information about the previous layer. The goal is to ensure the circulation of relevant information through the layers without transmitting noise information. This selection is conducted differently for the GLLVM layer compared to the regular DGMM layers. For the GLLVM layer, we perform logistic regressions of $\mathbf{y}^C$ over $\mathbf{z}^{(1)C}$ and delete the dimensions that were associated with non-significant coefficients in a vast majority of paths. Concerning the regular DGMM layers, information carried by the current layer given the previous layer has been modeled using a Principal Component Analysis. We compute the contribution of each original dimension to the first principal component analysis and keep

only the dimensions that present a high correlation with this first principal component, so that to drop information of secondary importance carried out through the layers.

Finally, the choice of the total number of layers is guided by the selected r_ℓ. For instance, if a dimension of two is selected for a head layer (or a dimension of one for a tail layer), then according to the identifiability constraint $p_h > r_1^h > ... > r_\ell^h > ... > r_L$, the following head (or tail) layers are deleted.

Given that this procedure also selects the number of components on the tail layers, it can also be used to automatically find the optimal number of clusters in the data. The user specifies a high number of components on the clustering layer and let the automatic selection operate. The optimal number of clusters is then the number of components remaining on the clustering layer at the end of the run. This feature of the algorithm is referred to as the "autoclus mode" of the MDGMM in the following and in the code. An evaluation of the "autoclus mode" is given for the M1DGMM on simulated data in the Supplementary Material.

Alternatively, in case of doubt about the number of clusters in the data, the MDGMM could be used in "multi-clustering mode". For example, if the number of clusters in the data is assumed to be two or three, one can define a MDGMM with three components on the first tail layer and two on the second tail layer. The first layer will output a three groups clustering and the second layer a two groups clustering. The two partitions obtained can then be compared to chose the best one. This can be done with the silhouette coefficient (Rousseeuw, 1987) as implemented in our code. In the "multi-clustering mode", the architecture selection just described is conducted with the number of components of the tail layers remaining frozen (as it corresponds to the number of clusters to look for in the data).

For all the clustering modes of the MDGMM, the architecture selection procedure is performed at the end of some iterations chosen beforehand by the user. Note that once the optimal specification has been determined, it is better to refit the model using the determined specification rather than keeping the former output. Indeed, changing the architecture "on the fly" (*i.e* through the model training), seems to disturb the quality of the final clustering.

Finally, in EM-based algorithms, the iteration which presents the best likelihood (the last one in general) is returned as the final output of the model. The likelihood of the model informs about how good the model is at explaining the data. However, it does not give direct information about the clustering performance of the model itself. Therefore, in the MDGMM we retain the iteration presenting the best silhouette coefficient (Rousseeuw, 1987) among all iterations. To summarize: the likelihood criterion was used as a stopping criterion to determine the total number of iterations of the algorithm and the best silhouette score was used to select the iteration returned by the model.

6 Real Applications

In this section we illustrate the proposed models on real datasets. First, we will present the continuous low dimensional representations of the data generated by the Discrete DGMM (DDGMM) and the M2DGMM. Then, the performance will be properly evaluated by comparing them to state-of-the-art mixed data clustering algorithms, the one-head version of the MDGMM (M1DGMM) provided with a Gaussian link function, the NSEP and the GLLVM. As some of the clustering models can deal with discrete data only (GLLVM, DDGMM) and other with mixed data (M1DGMM, MDGMM) we consider both types of data sets. The code of the introduced models is available on Github under the name MDGMM_suite. The associated DOI is 10.5281/zenodo.4382321.

6.1 Data description

For the discrete data specification, we studied the following three datasets:

- The Breast cancer dataset: a dataset of 286 observations and 9 discrete variables. Most of the variables are ordinal.
- The Tic Tac Toe dataset: composed of 9 variables corresponding to each cell of a 3×3 grid of tic-tac-toe. The dataset presents the grids content at the end of 958 games. Each cell can be filled with one of the player symbol (x or o), or left blanked (b) if the play has ended before all cells were filled in. Hence all the variables are categorical in contrast with the Breast cancer data. The goal is here to retrieve which game has led to victory of player 1 or of player 2 (no even games are considered here).
- The Mushrooms dataset: a two-class dataset with 22 attributes and 5644 observations once the missing data have been removed. The majority of the variables are categorical ones.

For mixed datasets, we studied the following three datasets:

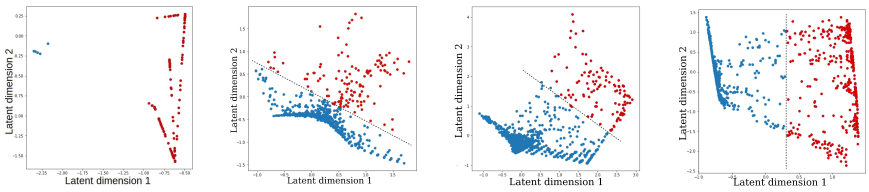
- The Heart (Stalog) dataset: composed of 270 observations, five continuous variables, three categorical variables, three binary variables and two ordinal variables.
- The Pima Indians Diabetes dataset: it presents several physiological variables (e.g. the blood pressure, the insulin rate, the age) of 768 Indian individuals. 267 individuals suffer from diabetes and the goal of classification tasks over this dataset is to distinguish the sound people from the sick ones. This dataset counts two discrete variables considered here respectively as binomial and ordinal and seven continuous variables.
- The Australian credit (Stalog) dataset: a binary classification dataset concerning credit cards. It is composed of 690 observations, 8 discrete categorical variables and 6 continuous variables.

In the analysis, all the continuous variables have been centered and reduced to ensure the numeric stability of the algorithms. All the datasets are available in the UCI repository (Dua and Graff, 2017).

6.2 Clustering vizualisation

According to their multi-layer structures, the DDGMM and the M2DGMM perform several dimension reductions of information while the signal goes through their layers. As such, they provide low dimensional continuous representations of complex data than can be discrete, mixed or potentially highly dimensional. These representations are useful to understand how observations are clustered through the training process. They could also be reused to train other algorithms in the same spirit as for supervised Neural Networks (Jogin et al., 2018).

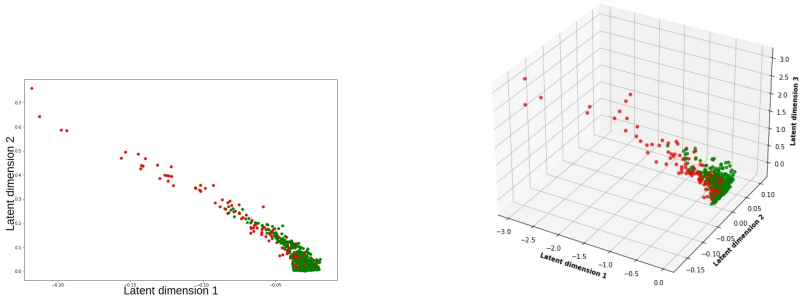
Figure 2 shows the evolution of the latent representation during the training of the clustering layer of a DDGMM for the Tic Tac Toe dataset. Four illustrative iterations have been chosen to highlight the training process. The clustering layer has a dimension of $r_\ell = 2$ and tries to distinguish $k_\ell = 2$ groups in the data. At the beginning of the training at $t1$, two sets of points are clearly separated but the latent space is very sparsely covered, denoting that the latent representation of the data is still preliminary. Through the next iterations, the latent space coverage is improved and two sets of points are pushed away from each other by the model. Moreover in $t2$ and $t3$ the frontier between the two clusters can be drawn as a straight line in a two dimensional space (represented by a dashed line on the figure). In $t4$ at the end of the training, the model seems to have found a simpler frontier to separate the groups as only a vertical line, i.e. a separation in a one dimensional space is needed. This highlight the information sorting process occurring through the layers in order to keep only the simplest and the more discriminating parts of the signal.



(a) Training at t1 (b) Training at t2 (c) Training at t3 (d) Training at t4

Fig. 2 Continuous 2D-latent representation of the Tic Tac Toe dataset through the training of a DDGMM. The clusters found are represented by different colors and the dashed line symbolizes the clusters separation during the last training iterations.

The next two figures illustrate graphical properties of the M2DGMM. Figure 3 presents two continuous representations of the Pima Diabetes data. These are obtained during the training of a M2DGMM with two hidden tail layers of respectively $r_{L_0+1} = 3$ and $r_{L_0+1} = 2$ during the same iteration. Two clusters are looked for in each case ($K_{L_0+1} = K_{L_0+2} = 2$) and are associated with green and red colors on the figure. On both layers the clusters are quite well

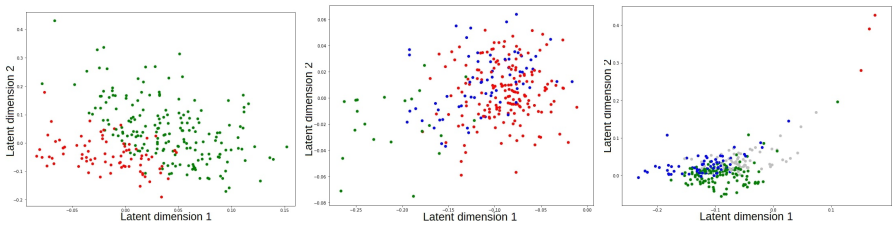


(a) 2D representation (b) 3D representation

Fig. 3 Continuous representations of the Pima Diabetes dataset provided by a M2DGMM

separated. The signal carried seems coherent between the two layers with a very similar structure. For the same computational cost, *i.e.* one run of the model, several latent representations of the data in different dimensions can therefore be obtained.

Finally, the graphical representations produced by the M2DGMM are useful tools to identify the right number of clusters in the data. Three M2DGMM have been run by setting $r_{L_0+1} = 2$ and with respectively $K_{L_0+1} = 2, K_{L_0+1} = 3$ and $K_{L_0+1} = 4$. The associated latent variables are presented in Figure 4 with a different color for each identified cluster. The representations with three



(a) 2 clusters (b) 3 clusters (c) 4 clusters

Fig. 4 Continuous representations of the Heart dataset at the end of the training of three M2DGMMs with different numbers of clusters specified

and four clusters present points that are intertwined, with no clear distinctions between clusters. On the contrary, when the number of clusters searched in the data is two this separation appears distinctly. Hence, this representation advocates for a two groups distinction in the data as it is suggested by the supervised labels of the dataset (absence or presence of heart disease). The four clusters representation also shows that the three points associated with the red cluster might be outliers potentially important to study.
As evoked in Subsection 5.2, this visual diagnostic can be completed by using the "autoclus mode" of the M2DGMM where the model automatically deter-

mines the best number of clusters in the data.

In addition to the graphical tools provided here, computing the contributions of the dataset variables to the latent dimensions of the model is also useful to interpret the clustering results, assess the impact of continuous and non-continuous variables on the clustering process and to perform variable selection. An example of such a representation is given for the M1DGMM on the Heart dataset in the Supplementary Material (see Section 9).

6.3 Performance comparison

Metrics	Silhouette	Micro	Macro
Algorithms	Breast Cancer		
GLLVM (random init)	0.215 (0.093)	0.673 (0.080)	0.570 (0.113)
GLLVM (with NSEP)	0.305 (0.023)	0.728 (0.025)	0.671 (0.018)
NSEP	0.303 (0.000)	0.722 (0.000)	0.664 (0.000)
DDGMM	0.268 (0.043)	0.696 (0.074)	0.648 (0.048)
k-Modes	0.174 (0.000)	0.592 (0.000)	0.534 (0.000)
k-Prototypes	0.293 (0.024)	0.729 (0.014)	0.666 (0.011)
Hierarchical	0.303 (0.000)	0.755 (0.000)	0.855 (0.000)
SOM	0.091 (0.088)	0.668 (0.060)	0.593 (0.011)
DBSCAN	0.264 (0.000)	0.726 (0.000)	0.860 (0.000)
	Tic Tac Toe dataset		
GLLVM (random init)	0.094 (0.031)	0.591 (0.052)	0.536 (0.100)
GLLVM (with NSEP)	0.110 (0.005)	0.550 (0.029)	0.545 (0.028)
NSEP	0.137 (0.000)	0.602 (0.021)	0.597 (0.019)
DDGMM	0.118 (0.016)	0.559 (0.028)	0.533 (0.036)
k-Modes	0.104 (0.002)	0.611 (0.000)	0.586 (0.000)
k-Prototypes	∅ (∅)	∅ (∅)	∅ (∅)
Hierarchical	0.078 (0.000)	0.654 (0.000)	0.827 (0.000)
SOM	0.082 (0.010)	0.650 (0.000)	0.560 (0.000)
DBSCAN	∅ (∅)	0.653 (0.000)	0.327 (0.000)
	Mushrooms dataset		
GLLVM (random init)	0.266 (0.103)	0.685 (0.107)	0.613 (0.255)
GLLVM (with NSEP)	0.351 (0.107)	0.803 (0.102)	0.854 (0.135)
NSEP	0.354 (0.064)	0.811 (0.101)	0.861 (0.074)
DDGMM	0.317 (0.078)	0.760 (0.131)	0.809 (0.116)
k-Modes	0.395 (0.000)	0.852 (0.000)	0.898 (0.000)
k-Prototypes	0.328 (0.081)	0.742 (0.136)	0.818 (0.086)
Hierarchical	0.395 (0.000)	0.854 (0.000)	0.904 (0.000)
SOM	0.155 (0.015)	0.710 (0.000)	0.814 (0.001)
DBSCAN	0.294 (0.000)	0.624 (0.000)	0.811 (0.000)

Table 1 Average results and standard errors over 30 runs of the best specification for each model over three discrete datasets

Algorithms	Silhouette	Micro	Macro
Metrics	Heart		
NSEP	0.165 (0.049)	0.738 (0.068)	0.739 (0.070)
M1DGMM	0.253 (0.003)	0.820 (0.012)	0.820 (0.012)
M2DGMM	0.146 (0.011)	0.710 (0.015)	0.712 (0.014)
k-Modes	0.247 (0.000)	0.811 (0.000)	0.813 (0.000)
k-Prototypes	0.044 (0.000)	0.593 (0.000)	0.585 (0.000)
Hierarchical	0.263 (0.000)	0.811 (0.000)	0.809 (0.000)
SOM	0.257 (0.000)	0.795 (0.000)	0.793 (0.000)
DBSCAN	0.177 (0.000)	0.556 (0.000)	0.724 (0.000)
	Pima		
NSEP	0.189 (0.013)	0.666 (0.056)	0.651 (0.051)
M1DGMM	0.227 (0.020)	0.633 (0.029)	0.607 (0.029)
M2DGMM	0.195 (0.079)	0.647 (0.019)	0.586 (0.068)
k-Modes	0.049 (0.033)	0.581 (0.000)	0.482 (0.000)
k-Prototypes	∅ (∅)	∅ (∅)	∅ (∅)
Hierarchical	0.391 (0.000)	0.656 (0.000)	0.826 (0.000)
SOM	0.232 (0.000)	0.644 (0.000)	0.610 (0.003)
DBSCAN	0.391 (0.000)	0.654 (0.000)	0.826 (0.000)
	Australian Credit		
NSEP	0.165 (0.034)	0.754 (0.098)	0.753 (0.110)
M1DGMM	0.170 (0.032)	0.707 (0.112)	0.806 (0.036)
M2DGMM	0.224 (0.080)	0.575 (0.040)	0.680 (0.104)
k-Modes	0.222 (0.007)	0.785 (0.008)	0.784 (0.007)
k-Prototypes	0.163 (0.000)	0.562 (0.000)	0.780 (0.000)
Hierarchical	0.399 (0.000)	0.849 (0.000)	0.847 (0.000)
SOM	0.127 (0.096)	0.649 (0.001)	0.676 (0.002)
DBSCAN	0.201 (0.000)	0.570 (0.000)	0.740 (0.000)

Table 2 Average results and standard errors over 30 runs of the best specification for each model over three mixed datasets

In order to benchmark the performance of the proposed strategy, we consider alternative algorithms coming from each family of approaches identified by Ahmad and Khan (2019), namely k-modes, k-Prototypes, Hierarchical Clustering, Self-Organising Maps (SOM), and DBSCAN (Ester et al., 1996).

For each dataset, we have set the number of unsupervised clusters to the "ground truth" classification number. In order to present a fair report, several specifications of the benchmark models have been run. For each specification, the models have been launched 30 times. The reported results correspond to the best specification of each benchmark model with respect to each metric on

average over the 30 runs. The set of specifications evaluated for each benchmark model is given in the Supplementary Material. Concerning our models, the architectures were automatically selected and then fitted 30 times on each dataset.
One unsupervised metric and two supervised metrics are used to assess the clustering quality: the silhouette coefficient, the micro precision and the macro precision. The silhouette coefficient measures how close on average a point is from the points of the same group with respect to the points of the other groups. The Euclidian distance cannot be used here due to the mixed feature space and hence the Gower distance (Gower, 1971) is used instead. The silhouette coefficient ranges between 1 (perfect clustering) and -1 (meaningless clustering). The micro precision corresponds to the overall accuracy, *i.e.* the proportion of correctly classified instances. The macro precision computes the proportion of correctly classified instances per class and then returns a non-weighted mean of those proportions. These two quantities tend to differ when the data are not balanced. The formal expressions of the metrics are given in the Supplementary Material. Note that we cannot use AIC or BIC criteria here since their values are not available for all methods.

Tables 1-2 present the best average results obtained by the algorithms and the associated standard error over the 30 runs in parenthesis. The best algorithm for a given dataset and metric is associated with a green cell and the worst with a red cell. An empty set symbol means that the metric was not defined for this algorithm on that dataset. For the special case of the k-prototypes algorithm, the empty set symbol means that the dataset contained only one type of discrete data which is a situation that the algorithm is not designed for.

6.3.1 Results on discrete data

The new initialisation (NSEP) enables the GLLVM to achieve better performances on the Mushrooms dataset and on the Breast dataset where the GLLVM attains the best silhouette score. It also stabilizes the GLLVM as the standard errors obtained are divided by at least a factor two for all metrics of the Breast Cancer and of the Tic Tac Toe datasets.
The NSEP in itself gives good results for all metrics and is often among the best two performing models. Finally, over the Tic Tac Toe dataset the DDGMM performs slightly better than the GLLVM, but less on the two other datasets. Hence, compared to the other methods, the models introduced in this work represent solid baseline models. On the contrary, some alternative methods appear to fit some datasets well and to poorly fit other ones. This is the case for instance of DBSCAN which performs well on the Breast cancer dataset, but much less on the Mushrooms and the Tic Tac Toe datasets (the algorithm could find only one group in the Tic Tac Toe data which explains that the silhouette score is not defined). Another example is k-Modes which obtains substantial results on the Mushrooms dataset but under-average results for

the two other datasets. Finally, among all methods, the hierarchical clustering is the algorithm that performs best on a majority of metrics and datasets.

6.3.2 Results on mixed data

As clear from results in Table 2, the NSEP seems again to be a good starting point for both algorithms and certainly also explains the fact that the M1DGMM reaches the best micro and macro scores on the Heart dataset.
The M1DGMM achieves better average results than the M2DGMM except for the silhouette score on the Australian Credit dataset and the micro precision on the Pima dataset. As in the discrete data results, the models introduced and especially the M1DGMM, give satisfactory performance on all datasets on average, whereas other models such as SOM, DBSCAN or k-modes perform well on some datasets only. Similarly, the hierarchical clustering method seems to provide the best results on a large set of metrics and datasets.

It is worth noting that the results presented in this section assumed that the number of clusters was known beforehand. The Supplementary Material gives additional information when it is not the case. Then, only DBSCAN and the hierarchical method can be used among the benchmark models and it appears that the M1DGMM seems to give better information than these two approaches about the number of classes supported by the data.

7 Conclusion

This work aimed to provide a reliable and flexible model for clustering mixed data by borrowing ingredients from the GLLVM and the DGMM recent approaches. Several sub-models have been introduced and could be used on their own:

- a new initialisation procedure called NSEP for GLLVM-based models,
- a Discrete DGMM (DDGMM) for discrete data,
- a one-head (M1DGMM) and a two-heads (M2DGMM) DGMM for mixed data.

This suite of models handles the usual clustering issues concerning architecture selection and the choice of the number of clusters in the data in an automated manner.
From the experiments carried out on real data, the MDGMM performances are in line with the other state-of-the-art models. It can be regarded as a baseline model over a general class of data. Its use of nested Mixtures of Factor Analyzers enables it to capture a very wide range of distributions and patterns.
Despite of its complexity, the MDGMM remains interpretable. From a practical viewpoint, the structure of the latent space can be observed through the model training with the help of the graphical utilities presented in section 6.2. Thus, they allow the user to perform visual diagnostics of the clustering

process. From a theoretical standpoint, the parameters of the model remain interpretable as the link between parameters and clustering results is proper thanks to the identifiability of the model. The set of identifiability constraints presented here could seem restrictive. However, it forces the model to stay in a quite well delimited parameter space and to avoid for instance a too significant explosion of the norm of the parameters values. The implementation of these constraints can nevertheless be improved by considering a Bayesian re-writing of our model on variational principles. Indeed, it should make identification requirements easier to meet, as one can keep only the posterior draws that meet the identifiability requirements. Niku et al. (2019) have rewritten the GLLVM model in a variational fashion using automatic differentiation and exhibit high running time and accuracy gains. Following their path, one could adapt the MDGMM to the variational framework.
Finally considering the training process, the choice of an EM-based algorithm was motivated by its extensive use in the Gaussian Mixture Model literature. The EM-related algorithms can however lead to significant variations of the partitions found from one run to another as pointed out by Selosse et al. (2020) in the DGMM case. The EM-related algorithms are also very sensitive to the initialisation, which was in our case particularly tricky given the size of the parameter space. Combining Multiple Correspondence Analysis with Gaussian Mixture Models, Factor Analysis and Partial Least Squares into NSEP has however enabled us to significantly stabilize the estimation process at least for moderate size architectures as shown in Supplementary Material. Yet, new initialisation and training processes could be designed to help the model to better rationalize latent structures in the data within its very highly dimensional space.

Acknowledgements

The authors thank the reviewers for their helpful comments which helped to improve the manuscript. This work benefited from the support of the Research Chair DIALog under the aegis of the Risk Foundation, a joint initiative by CNP Assurances and ISFA, Université Claude Bernard Lyon 1 (UCBL). This work also benefited from funds of the LIA LYSM (agreement between AMU, CNRS, ECM and INdAM).

Conflict of interest

The authors declare that they have no conflict of interest.

References

Ahmad A, Khan SS (2019) Survey of state-of-the-art mixed data clustering algorithms. IEEE Access 7:31883–31902

Akaike H (1998) Information theory and an extension of the maximum likelihood principle. In: Selected papers of hirotugu akaike, Springer, pp 199–213

Baydin AG, Pearlmutter BA, Radul AA, Siskind JM (2017) Automatic differentiation in machine learning: a survey. The Journal of Machine Learning Research 18(1):5595–5637

Biernacki C, Celeux G, Govaert G (2003) Choosing starting values for the em algorithm for getting the highest likelihood in multivariate gaussian mixture models. Computational Statistics & Data Analysis 41(3-4):561–575

Blalock D, Ortiz JJG, Frankle J, Guttag J (2020) What is the state of neural network pruning?, arXiv preprint arXiv:2003.03033

Cagnone S, Viroli C (2014) A factor mixture model for analyzing heterogeneity and cognitive structure of dementia. AStA Advances in Statistical Analysis 98(1):1–20

Chiu T, Fang D, Chen J, Wang Y, Jeris C (2001) A robust and scalable clustering algorithm for mixed type attributes in large database environment. In: Proceedings of the seventh ACM SIGKDD international conference on knowledge discovery and data mining, pp 263–268

Dua D, Graff C (2017) UCI machine learning repository. URL http://archive.ics.uci.edu/ml

Ester M, Kriegel HP, Sander J, Xu X, et al. (1996) A density-based algorithm for discovering clusters in large spatial databases with noise. In: Kdd, vol 96, pp 226–231

Fraley C, Raftery AE (2002) Model-based clustering, discriminant analysis, and density estimation. Journal of the American statistical Association 97(458):611–631

Fruehwirth-Schnatter S, Lopes HF (2018) Sparse bayesian factor analysis when the number of factors is unknown, arXiv preprint arXiv:1804.04231

Ghahramani Z, Hinton GE, et al. (1996) The em algorithm for mixtures of factor analyzers. Tech. rep., Technical Report CRG-TR-96-1, University of Toronto

Gower JC (1971) A general coefficient of similarity and some of its properties. Biometrics 27(4):857–871

Huang Z (1997) Clustering large data sets with mixed numeric and categorical values. In: Proceedings of the 1st pacific-asia conference on knowledge discovery and data mining,(PAKDD), Singapore, pp 21–34

Huang Z (1998) Extensions to the k-means algorithm for clustering large data sets with categorical values. Data mining and knowledge discovery 2(3):283–304

Jogin M, Madhulika M, Divya G, Meghana R, Apoorva S, et al. (2018) Feature extraction using convolution neural networks (cnn) and deep learning. In: 2018 3rd IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology (RTEICT), IEEE, pp 2319–2323

Kohonen T (1990) The self-organizing map. Proceedings of the IEEE 78(9):1464–1480

Maclaurin D, Duvenaud D, Adams RP (2015) Autograd: Effortless gradients in numpy. In: ICML 2015 AutoML Workshop, vol 238, p 5

McLachlan GJ, Peel D (2000) Finite mixture models, Probability and Statistics – Applied Probability and Statistics Section, vol 299. Wiley, New York

McLachlan GJ, Peel D, Bean RW (2003) Modelling High-Dimensional Data by Mixtures of Factor Analyzers. Computational Statistics and Data Analysis 41(3-4):379–388

Melnykov V, Maitra R, et al. (2010) Finite mixture models and model-based clustering. Statistics Surveys 4:80–116

Moustaki I (2003) A general class of latent variable models for ordinal manifest variables with covariate effects on the manifest and latent variables. British Journal of Mathematical and Statistical Psychology 56(2):337–357

Moustaki I, Knott M (2000) Generalized latent trait models. Psychometrika 65(3):391–411

Nenadic O, Greenacre M (2005) Computation of multiple correspondence analysis, with code in r. Tech. rep., Universitat Pompeu Fabra

Niku J, Brooks W, Herliansyah R, Hui FK, Taskinen S, Warton DI (2019) Efficient estimation of generalized linear latent variable models. PloS one 14(5):481–497

Pagès J (2014) Multiple factor analysis by example using R. CRC Press

Patil DD, Wadhai V, Gokhale J (2010) Evaluation of decision tree pruning algorithms for complexity and classification accuracy. International Journal of Computer Applications 11(2):23–30

Philip G, Ottaway B (1983) Mixed data cluster analysis: an illustration using cypriot hooked-tang weapons. Archaeometry 25(2):119–133

Rousseeuw PJ (1987) Silhouettes: a graphical aid to the interpretation and validation of cluster analysis. Journal of computational and applied mathematics 20:53–65

Schwarz G, et al. (1978) Estimating the dimension of a model. The annals of statistics 6(2):461–464

Selosse M, Gormley C, Jacques J, Biernacki C (2020) A bumpy journey: exploring deep gaussian mixture models. In: "I Can't Believe It's Not Better!"NeurIPS 2020 workshop

Viroli C, McLachlan GJ (2019) Deep gaussian mixture models. Statistics and Computing 29(1):43–51

Wei GC, Tanner MA (1990) A monte carlo implementation of the em algorithm and the poor man's data augmentation algorithms. Journal of the American statistical Association 85(411):699–704

Wold S, Sjöström M, Eriksson L (2001) Pls-regression: a basic tool of chemometrics. Chemometrics and intelligent laboratory systems 58(2):109–130