

# An Automatic Type-Inferential General Latent Feature Model

1<sup>st</sup> Neil Dhir

*The Alan Turing Institute*  
London, United Kingdom  
ndhir@turing.ac.uk

2<sup>th</sup> Tomasz Rudny

*Mind Foundry Ltd.*  
Oxford, United Kingdom  
tomasz.rudny@mindfoundry.ai

3<sup>rd</sup> Davide Zilli

*Mind Foundry Ltd.*  
Oxford, United Kingdom  
davide.zilli@mindfoundry.ai

4<sup>nd</sup> Alessandra Tosi




*Mind Foundry Ltd.*  
Oxford, United Kingdom  
alessandra.tosi@mindfoundry.ai

**Index Terms**—AutoML, statistical data-type, Bayesian non-parametrics, latent features, graphical models, MCMC

**Abstract**—With ever more data becoming available, there has been a recent drive to develop modelling tools for heterogeneous data sets such as electronic healthcare records. Therein appears both Bayesian non-parametric latent feature models, as well as methods for automatically determining the statistical data type (e.g. ordinal or categorical) of the attributes present in the data. We present a model which combines both of these attractive features in an end-to-end framework. By jointly learning the model complexity and statistical type of the data, we demonstrate that redundant information can be discarded while higher accuracy statistical type estimates are produced on real experimental data.

## I. INTRODUCTION

The research area of automatic machine learning (AutoML) targets the progressive automation of machine learning methods, aiming to make effective methods available to everyone. At its core AutoML seeks to progressively augment human machine learning experts, to enable them to focus on data analysis, whilst leaving more mundane tasks, such as data ingestion, to an automatic transformation and identification system. One such mundane task is the identification of statistical type i.e. if a sample of data is e.g. ordinal, categorical, nominal or real-valued (too mention but a few types).

Whilst it is trivial to construct simple heuristics to distinguish between major types of data (e.g. continuous vs. discrete), it is much harder to reason about sub-types. For example, suppose we are presented with this univariate data-sample: {, , }. Have we been presented with a sample from a traffic light or a data sample from a bag of M&M sweets? The former has order, the latter does not. Without further semantic knowledge, *heuristically*, this is an almost impossible inference task, but one which can be addressed by stochastic data modelling and which would have high utility in an AutoML framework.

### A. On the importance of type

One of the primary assumptions in data analysis is that we can describe a data set of objects (examples), using a common set of attributes. Typically though, for further analysis to proceed, the attributes need to be assigned a *type* (categorical, ordinal, real, Boolean etc.). The type informs the inference, hence its importance. [15] explains that prediction tasks differ depending on the kind of data we are considering. If it is an ordinal type, we employ ordinal regression. If it is a real type,

we may employ linear regression. The type instance also has implications for the resultant pre-processing. For example, one-hot-encoding transforms categorical attributes to a format that works better with many classification and regression algorithms. But many times we cannot encode attributes nominally, because this would impose a non-existent order on the data. Hence, for statistical machine learning, the statistical type is paramount. This amounts to an assumption of “known and fixed” [15] likelihood model – where the Gaussian is a particularly popular choice. But few methods exist for performing automatic model selection – commensurate with finding the statistical type of the columns  $\{\mathbf{x}^d \in \mathbf{X} \mid d = 1, \dots, D\}$  in data set  $\mathbf{X}$ . For high-dimensional heterogeneous data sets there exists a clear need for automating this process, particularly when the data sets in question are large (examples are shown in table I).

TABLE I: High-dimensional heterogeneous data sets. Unless otherwise specified, all can be found in the UCI repository [1]. The last two rows are electronic healthcare records (EHR).

Data set	$N$	$D$
Internet Advertisements	3, 279	1, 558
Arrhythmia	452	276
KDD Cup	191, 779	481
UJIIndoorLoc	21, 048	529
EHR A [6]	64, 819	1, 865
EHR B [8]	40, 736	4, 894

### B. Type is not all

Heterogeneous data sets make modelling more complex as discussed, hence the need to identify the statistical types (i.e. identify their likelihood model) of the attributes present. Supposing this can be accomplished, then one prominent way of in-painting missing data is to employ latent features, which are computed from the observations using matrix factorisation.

While a few select methods exist for heterogeneous latent feature modelling and type inference, to our knowledge, there exists no end-to-end model which can do both. This is a difficult task because we are asking the framework to model likelihoods which may change during run-time. But at the same time, by also jointly modelling the model complexity, we can quickly

infer good estimates of the column data types. And so, to add to the current coterie of heterogeneous data modelling, our contributions are as follows:

- 1) We combine Bayesian non-parametric heterogeneous data modelling with automatic type detection. The *general latent type and feature model* (GLTFM) overcomes the current limitation where the user has to pre-specify a set of column types for heterogeneous data modelling.
- 2) Our method in (1) is unsupervised and non-parametrically learns column types *conditioned* on the inferred model complexity from (1). Because we employ a latent feature model paradigm, redundant information present in the columns can be disregarded, and a compact representation used instead.

The paper is structured as follows: a technical problem description is presented in §II, the model is presented in §III, accompanying inference protocols in §IV and finally experimental results are presented in §V.

## II. PROBLEM STATEMENT

Posit that our observations (data) are stored in design matrix  $\mathbf{X}$  of size  $N \times D$ . We denote object  $n$  on the rows, as  $\mathbf{x}_n \triangleq [x_n^1, x_n^2, \dots, x_n^d, \dots, x_n^D]$  and the attribute/column location is dictated by index  $d$ . In probabilistic low-rank matrix factorisation we assume that  $\mathbf{X}$  can be approximated by the matrix product, denoted by  $\odot$ , between a binary feature matrix  $\mathbf{Z} \in \{0, 1\}^{N \times K}$  (*binary by choice*, it can also be e.g. Gaussian distributed) and a weight matrix  $\mathbf{B} \in \mathbb{R}^{K \times D}$ . Under this model, the rows in  $\mathbf{Z} \triangleq [\mathbf{z}_1, \mathbf{z}_2, \dots, \mathbf{z}_N]^\top$  are exchangeable. Consequently it follows that each column  $k$  of  $\mathbf{Z}$  corresponds to a latent feature, where  $z_{nk} = 1$  if feature  $k$  contributes to observation  $n$ . Should feature  $k$  not contribute, then  $z_{nk} = 0$ . The weighting vector  $\mathbf{b}^d$  weighs the contribution of the  $k^{\text{th}}$  latent feature, to the  $d^{\text{th}}$  dimension of  $\mathbf{X}$ , s.t.  $\cup_d \mathbf{b}^d = \mathbf{B}$ . Like the rows in  $\mathbf{Z}$ , note that  $\mathbf{b}^i \perp \mathbf{b}^j, \forall i \neq j \in \{1, \dots, D\}$ , yielding an approximation  $\mathbf{X} \approx \mathbf{Z} \odot \mathbf{B} + \epsilon$ . Where  $\epsilon$  is white Gaussian noise with distribution  $\epsilon \sim \mathcal{N}(\mathbf{0}, \Sigma)$ . Noise is independent of  $\mathbf{Z}$  and  $\mathbf{B}$  and is uncorrelated across observations.

Database modelling typically assumes that column likelihood models (types)  $l^d \forall d \in \{1, \dots, D\}$  are known *a priori*. For example in the linear-Gaussian latent feature model (LG-LFM) [3] the set of column types is given by  $\mathcal{L} \triangleq \{l^d \mid d = 1, \dots, D\}$  where all  $\{l^d\}_{d=1}^D$  are assumed to be Gaussian. This is a form of probabilistic low-rank matrix factorisation which can also be seen as latent feature modelling [3]. The Indian buffet process (IBP) operator in the LG-LFM, imposes binary latent features on  $\mathbf{X}$ . The LG-LFM, like many models, assumes that  $\mathbf{X}$  is type-homogeneous (Gaussian) across dimensions. Another LFM, infinite sparse factor analysis [5], makes the same assumptions about  $\mathbf{X}$ . Indeed, most LFMs assume homogeneity in  $\mathbf{X}$  – being either real [7], [11], [14] or categorical [9], [10], [14]. There are some studies which do consider heterogeneous data such as [4], [12]. In particular, [13] presented a novel general LFM (GLFM) capable of dealing with heterogeneous observations, but with the proviso that  $\{l^d\}_{d=1}^D$  are known *a priori*. This is

a prohibitive requirement since this information is not always available and as noted earlier, can be prohibitively expensive to ascertain.

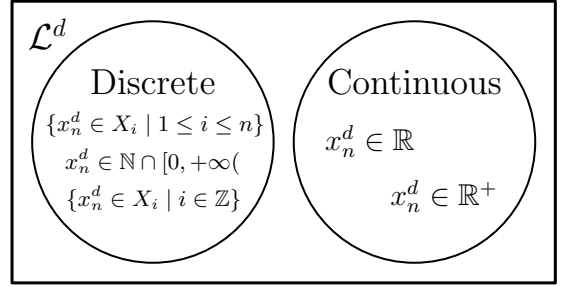


Fig. 1: Different data types. From top-to-bottom, *continuous*: real and positive real; *discrete*: categorical, ordinal and count.

Type inference typically employs heuristics to determine whether  $x_n^d$  is continuous or discrete. These are adequate to determine if variables are e.g. Boolean or integers. Finding the sub-types, as shown in fig. 1 is much harder. The reason being that, first, we only have access to finite data sample – hence we do not know what we have not observed. Second, we cannot distinguish between ordinal data and count data (defined on an infinite ordered space with equidistant values) – recall the example given in §I. But without complete information we do not know if samples are equidistant. In the context of information entropy, our measure of uncertainty of each variable is only approximate as our sample will always be finite, and we require infinite samples to exactly determine type. To address this problem, [15] proposed a Bayesian method which accurately discovers statistical data types (of the flavours shown in fig. 1), by imposing a likelihood-model prior, where the possible set of types is *a priori* unknown so that the type set becomes a set of sets  $\mathcal{L} = \{l \in \mathcal{L}^d \mid d = 1 \dots, D\}$  where  $\mathcal{L}^d \triangleq \{l_1^d, \dots, l_J^d\}$  specifies a set of potential likelihood models, of size  $|\mathcal{L}^d| = J$ , for column  $d$  (naturally we do not need specify exactly  $J$  candidates per column). [15]’s method concerns finding likelihood-weights  $\mathbf{w}^d \triangleq [w_1^d, \dots, w_J^d]$  for each  $d$ , which captures the probability of type  $l_j^d$ , in attribute  $\mathbf{x}^d \in \mathbf{X}$ . Consequently when the prior likelihood-models are unknown, then

$$\mathcal{L} = \left\{ \underbrace{\{l_1^1, \dots, l_J^1\}}_{=\mathcal{L}^1}, \dots, \underbrace{\{l_1^d, \dots, l_J^d\}}_{=\mathcal{L}^d}, \dots, \underbrace{\{l_1^D, \dots, l_J^D\}}_{=\mathcal{L}^D} \right\}$$

but when known, as in the case of the GLFM, then simply  $\mathcal{L} = \{l^1, \dots, l^D\}$ . In §III-C we present a general Bayesian non-parametric latent feature model suitable for heterogeneous data sets, which infers the model complexity and the statistical types from the data.

## III. METHODS

This section describes a model capable of *complete*<sup>1</sup> automatic heterogeneous latent feature modelling by virtue

<sup>1</sup>By ‘complete’ we mean: data infusion, latent variable modelling and type prediction.

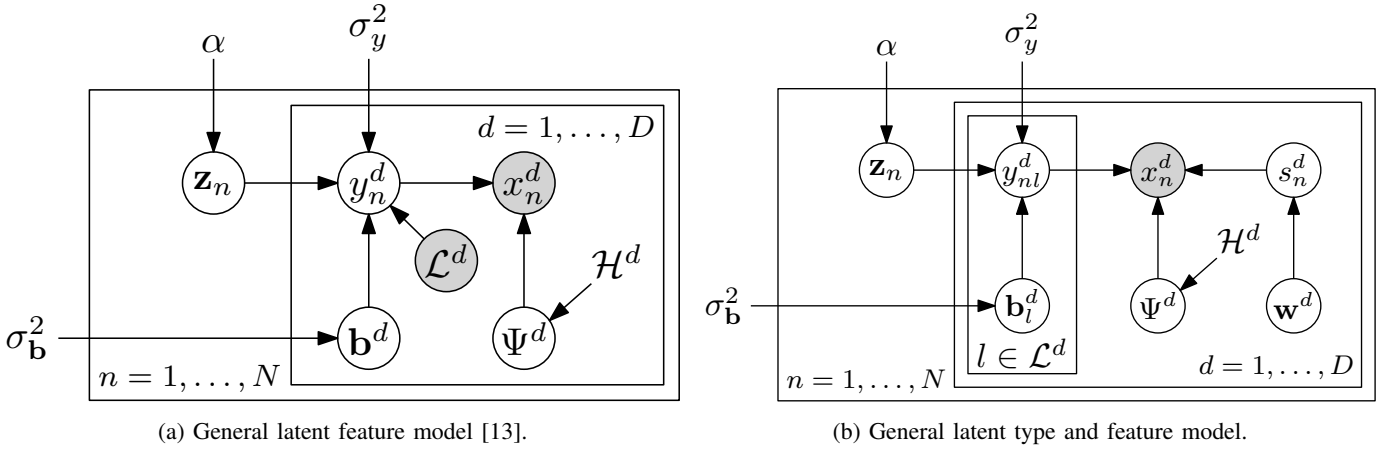


Fig. 2: Probabilistic graphical models. The above models describe the generative process behind the models under discussion. They are presented on a per-observation level to emphasise the full generative process, per data point. Note the extension inherent in fig. 2b: it provides for a mechanism to learn the likelihood model for each dimension of  $\mathbf{X}$ .

of automatic type inference. This is achieved, broadly, by wrapping a type inference mechanism (we refer to this as the ‘inner’ model), with latent feature modelling (‘outer’ model). By interleaving non-parametric inference over model complexity  $K$ , evaluating the inner model at this  $K$ , and then updating the posterior estimate of the types  $\mathcal{L}$ , we propose a form of global optimisation over  $K$  and  $\mathcal{L}$ .

#### A. Latent feature modelling

We begin by giving a brief synopsis of the GLFM (see fig. 2a), enough to lead us into the GLTFM. For more incisive commentary on the GLFM we refer the reader to [14]. And so, the *outer* model consists of a *non-parametric* LFM. Therein we seek to find the posterior distribution of  $\mathbf{Z}$  and  $\mathbf{B}$ , where independence is assumed *a priori* for  $\mathbf{Z}$  and  $\mathbf{B}$ , and where the likelihood model  $p(\mathbf{X} | \mathbf{Z}, \mathbf{B})$  is determined by the application [2] and so too the feature prior  $p(\mathbf{B})$ . Our interest is that of latent feature modelling so the likelihood can be factorised as

$$p(\mathbf{X} | \mathbf{Z}, \mathbf{B}, \mathcal{L}) = \prod_{d=1}^D \prod_{n=1}^N p(x_n^d | \mathbf{z}_n, \mathbf{b}^d, l^d). \quad (1)$$

The density  $p(\mathbf{Z})$  requires a flexible prior [3] and because  $\mathbf{Z}$  is an indicator matrix which tells us if a particular feature in  $\mathbf{B}$  is present in object  $n$ , we want to determine  $K$  at run-time but also leave it unbounded – achieved using an Indian buffet process (IBP) matrix prior. When we allow  $\mathbf{X}$  to include heterogeneous data types, we can capture the latent structure using the GLFM. Under this model the type set  $\mathcal{L}$  is assumed known. By that we mean that the user has to manually specify the type of each and every column in  $\mathbf{X}$  before the GLFM can be used. This might be feasible for a low-dimensional data sets, but becomes highly impractical once we start to consider high-dimensional data sets as those found in table I.

This is one of the drawbacks that we address by including automatic type inference, which allows  $\mathcal{L}$  to be determined

during run-time. We discuss in §III-C how posterior type estimates are produced. But first, let

$$\begin{aligned} p(\mathbf{b}^d | \sigma_{\mathbf{B}}) &\triangleq \mathcal{N}(\mathbf{0}, \sigma_{\mathbf{B}}^2 \mathbf{I}), & p(\mathbf{Z} | \alpha) &\triangleq \text{IBP}(\alpha), \\ p(\mathbf{X} | \mathbf{Z} \odot \mathbf{B}, \sigma_{\mathbf{X}}) &\triangleq \mathcal{N}(\mathbf{Z} \odot \mathbf{B}, \sigma_{\mathbf{X}}^2 \mathbf{I}) \end{aligned} \quad (2)$$

where  $\mathbf{Z}$  follows an IBP prior with concentration parameter  $\alpha$ , with hyperpriors (not shown in fig. 2 to avoid clutter):

$$\begin{aligned} p(\alpha) &\triangleq \mathcal{G}(a_\alpha, b_\alpha), & p(\sigma_{\mathbf{X}}) &\triangleq \mathcal{G}(a_y, b_y), \\ p(\sigma_{\mathbf{B}}) &\triangleq \mathcal{G}(a_B, b_B). \end{aligned} \quad (3)$$

The model in eq. (1) extends the LG-LFM in [3] and [13] extended one of the central tenets of that model, the IBP, to account for heterogeneous data while maintaining the model complexity of conjugate models. The IBP places a prior on binary matrices where the number of columns, corresponding to latent features  $K$ , is potentially infinite and its utility rests on two foundations. First, we can learn the model complexity from the data. Second, [13] argues that binary-valued latent features have been shown to provide more interpretable results in data exploration than standard real-valued latent feature models [9], [10]. As interpretability is a key feature of e.g. EHR data, this is something which we wish to promote. But, as far as the GLFM is concerned, it is silent on the inference of type. Alas, what we practically seek is to make the observed  $\{\mathcal{L}\}_{d=1}^D$  node in fig. 2a, inferable. This will result in an estimate of  $\mathcal{L}$  which means that our posterior estimate of  $\mathbf{X}$  incorporates our uncertainty about the type itself. This is important because it may provide further insight into the generative process of a particular column, e.g. regarding its ordinal (or not) nature. As we saw in the example in §I this is not a trivial exercise, and one which warrants automation for large data sets wherein data exploration is required or desired. Indeed, we can easily demonstrate how the LFM paradigm breaks down when the wrong statistical type is imposed.

### B. The wrong type and model complexity

One of the primary contentions of this paper is that there is utility in sharing the number of latent features  $K$  with a type-inference mechanism. Before we move onto experiments in §V, we will here empirically demonstrate the impact of inferring an appropriate  $K$  and  $\mathcal{L}$ .

We apply the GLFM to a simulated data set consisting of 100  $6 \times 6$ , each generated by randomly assigning a feature to each image to a class with probability of 20%, and taking a linear combination of the weights associated with features to which the images were assigned. We add isotropic Gaussian noise ( $\sigma^2 = 0.5$ ). The 100 images were generated as binary linear combinations of four ( $K = 4$ ) sets of class weights, shown in fig. 3.

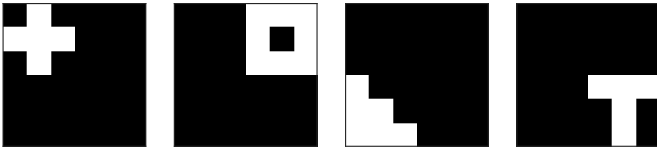


Fig. 3: Four ( $K = 4$ ) sets of binary class-weights, where each feature is of size  $6 \times 6$ .

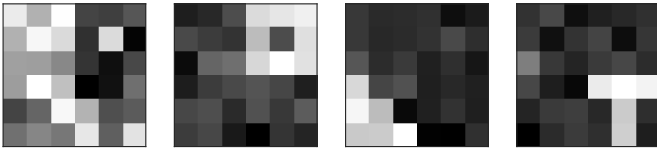


Fig. 4: Example of inferred weights  $\mathbf{B}$  after 1000 simulations using the GLFM. The order has been altered to match the true classes. Attributes in  $\mathbf{X}$  take values on the real line  $\mathbb{R}$ .

We compare our generating matrix  $\mathbf{Z}_0$  to the posterior estimate  $\mathbf{Z}$ , by measuring the structure error [17], which is equivalent to

$$\epsilon_{\text{SE}} = \sum_{i,j} \left| \mathbf{z}_0 \mathbf{z}_0^\top - \mathbb{E} \left[ \mathbf{z} \mathbf{z}^\top \right] \right|_{i,j} \quad (4)$$

where the difference is measured between the upper triangular portions of the matrix products. For four values of  $\alpha$  (higher means more exploration in  $K$ -space, lower the opposite), we ran ten independent simulations for each value. The results are summarised in table II. We truncated the model to a maximum of ten features ( $K_{\text{max}} = 10$ ) and initialised each experiment with  $K = 2$ . The resulting expected model complexity and structure error for  $\alpha$  are shown in table II.

TABLE II: 1000 MCMC simulations were used for each of the ten independent experiments. Attributes in  $\mathbf{X}$  take values on the real line  $\mathbb{R}$ .

$\alpha$	1	2	4	8
$\mathbb{E}[K]$	3.0	<b>4.4</b>	5.2	5.1
$\mathbb{E}[\epsilon_{\text{SE}}]$	3389.6	<b>3099.3</b>	4581.7	4007.3

These results trivially demonstrate that finding the correct number of features is important for posterior parameter estimation. This is intuitively obvious, however, we also show that using the wrong number of features has detrimental effect to the estimation task. We practically demonstrate the importance of this in §V, where it is shown that using the wrong number of latent features can result in erroneous type prediction.

Having weighed upon the importance of  $K$ , where we issued the model with the correct type, with manifested observations taking values on the real line i.e.  $\mathcal{L} = \{l^d = \mathbb{R} \mid d = 1, \dots, 36\}$ . We continue by prescribing the *wrong* type  $\mathbb{R}_+$  to measure the effect in the structure error. Results are shown in table III. The same experimental setup was employed.

TABLE III: 1000 MCMC simulations were used for each of the ten independent experiments. Values in  $\mathbf{X}$  take values on the positive real line.

$\alpha$	1	2	4	8
$\mathbb{E}[K]$	8.0	8.0	8.0	8.0
$\mathbb{E}[\epsilon_{\text{SE}}]$	8948.2	7922.9	8733.3	7195.6

As demonstrated in table III, using the wrong likelihood model has a large effect on the inference mechanism's ability to produce accurate posterior estimates of the model parameters.

### C. Latent feature and type modelling

In §III-B we showed two failure modes of the GLFM, which we now proceed to address by introducing the GL(T)FM – a GLFM with type inference. In [15] the authors demonstrate that their type-inferential model (from hereon referred to as the general latent type model (GLTM) for ease of discussion) requires the number of latent features  $K$  to be set *a priori*. However, type prediction is intimately linked to the model complexity  $K$  as shown in §III-B. By a similar token, model complexity in the GLFM is intimately linked to the statistical column types as shown in §III-B.

We have sought to maintain notation commonality between this work and [14], [15], to enable ease of comparison between all studies. Alas, in the inner model, our interest lies in capturing the generative distribution of each attribute  $\mathbf{x}^d \in \mathbf{X}$ . We can entertain this problem using the main idea from [15], where we assume that the likelihood model of  $\mathbf{x}^d \in \mathbf{X}$ , is a mixture of likelihood models

$$p(\mathbf{x}^d \mid \mathbf{Z}, \{\mathbf{b}_l^d\}_{l \in \mathcal{L}^d}, K) = \sum_{l \in \mathcal{L}^d} w_l^d \cdot p_l(\mathbf{x}^d \mid \mathbf{Z}, \mathbf{b}_l^d, K)$$

with likelihood-specific mixture weights given by  $w_l^d$ , with likelihood-specific density/mass function given by  $p_l(\cdot)$  and model complexity  $K$  passed from the outer model. By explicitly conditioning on the posterior estimate of the binary feature matrix from the outer model (via  $K$ ), we do *not* need to perform inference over the model complexity. Hence  $K$  is fixed from the outset *but*, crucially, it is an inferred parameter unlike in [15]. The usual provisos hold for this mixture:  $\sum_{l \in \mathcal{L}^d} w_l^d = 1$  and all densities  $p_l(\cdot)$  are normalised. Moreover, to avoid clutter, we

have omitted  $\Psi^d$  and  $\mathcal{H}^d$  from the density. Where the former denotes the set of random variables necessary to define the distribution of the  $d$ -th attribute and  $\mathcal{H}^d$  contains the hyper-parameters associated with the random variables in  $\Psi^d$ . Further, let for the inner model:

$$\begin{aligned} p(\mathbf{b}_l^d | \boldsymbol{\mu}_l^d, \boldsymbol{\Sigma}_b) &\triangleq \mathcal{N}(\boldsymbol{\mu}_l^d, \boldsymbol{\Sigma}_b), \\ p(\mathbf{z}_n | \boldsymbol{\mu}_Z^d, \boldsymbol{\Sigma}_Z) &\triangleq \mathcal{N}(\boldsymbol{\mu}_Z^d, \boldsymbol{\Sigma}_Z), \\ p(\mathbf{X} | \mathbf{Z} \odot \mathbf{B}, \sigma_X) &\triangleq \mathcal{N}(\mathbf{Z} \odot \mathbf{B}, \sigma_X^2 \mathbf{I}) \end{aligned} \quad (5)$$

In addition, as noted earlier, one of the goals with the GLTFM is automatic type inference. This means assigning one of the supported statistical types in fig. 1. But to do so, we first need to ascertain whether a column is continuous or discrete and then once that has been established, use MCMC inference to find the correct sub-type (e.g. categorical or ordinal). We do this using simple heuristics such as counting the number of unique values in a column, amongst other tests.

1) *Pseudo-observations*: At the core of heterogeneous database modelling is the innovation by [14] which enables  $\mathbf{X}$  to contain the multiple statistical types found in fig. 1, as well as efficient inference [15]. In that model representation, for each observation  $x_n^d$ , an auxiliary Gaussian variable  $y_n^d$  (a ‘pseudo-observation’) is introduced. In and of itself, this allows for the development of efficient inference algorithms [14]. For further details, see [13, §2]. Pseudo-observations are simulated as  $y_{nl}^d \sim \mathcal{N}(\mathbf{z}_n \mathbf{b}_l^d, \sigma_y^d)$ , and when the latent variables are conditioned on the pseudo-observations, the latent variables model behaves as a standard conjugate Gaussian model, with the associated efficient inference for the latent feature matrix and the weight vectors. Under this construction, the real line  $\mathbb{R}$  is mapped to the support  $\Omega_l$  of the  $d^{\text{th}}$  attribute in  $\mathbf{X}$ . The pseudo-observation paradigm holds for both the outer and inner model.

Following [15], we introduce latent variable  $s_n^d$  which indicates which likelihood model observation  $x_n^d$  belongs to, s.t.  $s_n^d \sim \text{Multinomial}(\mathbf{w}^d)$ . Hence, the transformation and pseudo-observation are found as

$$x_n^d = f_{s_n^d} \left( y_{n s_n^d}^d + u_n^d \right) \quad (6)$$

where  $u_n^d \sim \mathcal{N}(0, \sigma_u^2)$  is a noise variable s.t. that the support of  $f_{s_n^d}$  is  $\Omega_l$  for likelihood model  $l$ . For further information see [15, §3.1].

#### IV. INFERENCE

The GLTFM consists of a non-parametric LFM as the ‘outer’ model, responsible for model complexity, and a parametric LFM as the ‘inner’ model, responsible for the statistical type(s). This means we are interleaving two ways of describing  $\mathbf{X}$ , using two different inference mechanisms to do so. For details on these inference schemes see [13] and [15] respectively. The main parameter updates are outlined hereafter.

##### A. Outer model: binary latent features

Our approach is similar to that by [13] with the addendum that we condition on  $\mathcal{L}$ . The probability of each element in  $\mathbf{Z}$

is given by

$$\begin{aligned} p(z_{nk} = 1 | \{\mathbf{y}\}_{d=1}^D, \mathbf{Z}_{-nk}, \mathcal{L}) &\propto \\ \frac{m_{-nk}}{M} \prod_{d=1}^D \prod_{r=1}^{S_d} \int_{\mathbf{b}_r^d} p(y_{nr}^d | \mathbf{z}_n, \mathbf{b}_r^d) p(\mathbf{b}_r^d | \mathbf{y}_{-nr}^d, \mathbf{Z}_{-n}) d\mathbf{b}_r^d \end{aligned} \quad (7)$$

where  $S_d$  is the number of columns in matrices  $\mathbf{Y}$  and  $\mathbf{B}$  which contain categorical types (note also the explicit conditioning on  $\mathcal{L}$ ). All other types render  $S_d = 1$  i.e. when a column is not categorical. Further  $\mathbf{Z}_{-n}$  is  $\mathbf{Z}$  with the  $n^{\text{th}}$  row removed. Where  $\mathbf{y}_{-nr}^d$  is the  $r^{\text{th}}$  column of matrix  $\mathbf{Y}$  ( $r = 1$  if the type is not categorical), without element  $y_{nr}^d$ . Finally,  $p(\mathbf{b}_r^d | \mathbf{y}_{-nr}^d, \mathbf{Z}_{-n}) = \mathcal{N}(\mathbf{b}_r^d | \mathbf{P}_{-n}^{-1} \boldsymbol{\lambda}_{-nr}^d, \mathbf{P}_{-n}^{-1})$  is the posterior of the feature weight without taking the  $n^{\text{th}}$  observation into account [13]. Here  $\mathbf{P}_{-n} = \mathbf{Z}_{-n}^T \mathbf{Z} + \sigma_b^{-2} \mathbf{I}$  and  $\boldsymbol{\lambda}_{-nr}^d = \mathbf{Z}_{-nr}^T \mathbf{y}_{-nr}^d$  are the natural parameters of the Gaussian distribution.

##### B. Inner model: statistical types

The updates for the inner model (fig. 2b) are easier since there are no non-parametric mechanisms, and feature vectors can be efficiently sampled. Again we take our cue from [15] but note that in these equations our latent state number  $K$  has been inferred from the outer model – see §IV-A. Thus, feature  $\mathbf{z}_n$  of  $\mathbf{Z}$  is simulated from  $\mathbf{z}_n \sim \mathcal{N}(\boldsymbol{\mu}_Z^d, \boldsymbol{\Sigma}_Z)$ , with  $\boldsymbol{\mu}_Z^d = \boldsymbol{\Sigma}_Z \left( \sum_{l \in \mathcal{L}^d} \sum_{nl} \mathbf{b}_l^d y_{nl}^d \right)$  and  $\boldsymbol{\Sigma}_Z = \left( \sum_{d=1}^D \sum_{l \in \mathcal{L}^d} \mathbf{b}_l^d (\mathbf{b}_l^d)^\top + \sigma_Z^{-2} \mathbf{I} \right)$ . Feature weight  $\mathbf{b}_l^d \sim \mathcal{N}(\boldsymbol{\mu}_l^d, \boldsymbol{\Sigma}_b)$ , with  $\boldsymbol{\mu}_l^d = \boldsymbol{\Sigma}_b \left( \sum_n \mathbf{z}_n^\top y_{nl}^d \right)$  and  $\boldsymbol{\Sigma}_b = \left( \sigma_y^{-2} \mathbf{Z}^\top \mathbf{Z} + \sigma_b^{-2} \mathbf{I} \right)$ .

$$\begin{aligned} p(s_n^d = l | \mathbf{w}^d, \mathbf{Z}, \{\mathbf{b}_l^d\}, K) &= \\ \frac{w_l^d p_l(x_n^d | \mathbf{z}_n, \mathbf{a}_l^d)}{\sum_{l' \in \mathcal{L}^d} w_{l'}^d p_{l'}(x_n^d | \mathbf{z}_n, \mathbf{a}_{l'}^d)} \end{aligned} \quad (8)$$

Finally, we place a prior on the likelihood weight vector  $\mathbf{w}^d$  for dimension  $d$ , using a Dirichlet distribution parametrised by  $\{\alpha_l\}_{m \in \mathcal{L}^d}$ . Then using the likelihood assignments in eq. (8), we exploit conjugacy for updates to the parameters using the counts. A pseudo-inference algorithm is described in alg. 1 (outlining the main inference operations).

As one of our goals is interpretability, the output which we ultimately seek is the binary feature matrix of the outer model. Moreover, in alg. 1 we are implicitly passing all parameters to the GLTFM but have omitted them to emphasise the interleaving of (and inference over)  $K$  and  $\mathcal{L}$ .

#### V. EXPERIMENTS

In this section we compare the GLTFM to the GLTM on several real datasets described in table IV. To provide commonality and easy of comparison with [15], we have selected five data sets which can also be found in the aforementioned study. Further, the experimental parameters in table V were used. Finally, due to space constraints, we only show results for ‘interesting’ findings i.e. where the GLTFM and the GLTM deviate from ground-truth or where they differ in their outcome.

---

**Algorithm 1:** GLTFM pseudo-inference

---

**Input :** Raw observations  $\mathbf{X}_{N \times D}$ ,  $t, F, G, H$ 

1. Initialise all GLTFM parameters
2. Get canonical statistical column types
3. Map observations  $\mathbf{X}$  to pseudo-observations  $\mathbf{Y}$

**for**  $i = 1, \dots, H$  **do****for**  $j = 1, \dots, F$  **do** $\triangleright$  Update all outer GLTFM parametersInfer  $K$  given  $\mathcal{L}$ **if**  $i \bmod t = 0$  **then****for**  $m = 1, \dots, G$  **do** $\triangleright$  Update all inner GLTFM parametersInfer  $\mathcal{L}$  given  $K$ **Output:**  $\mathbf{Z}, \{\mathbf{b}\}_{d=1}^D, \{\Psi\}_{d=1}^D, \mathcal{L}$ 

---

TABLE IV: Heterogeneous real data used for experiments, where #Discrete is the number of discrete columns.

Dataset	$N$	$D$	#Discrete
German	1,000	21	16
Breast	681	10	9
Abalone	4,177	9	2
Wine	177	14	2
Dermatology	366	35	35

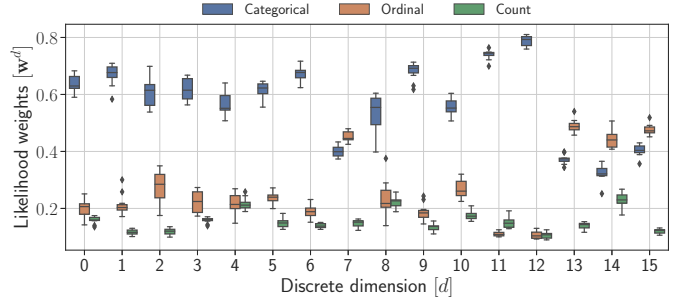
TABLE V: Experimental parameter sets (two in total) used for experiments. The first and second row denote how many simulations we used for the outer and inner models respectively, where we used an equivalent number for each repeated GLTM run (third row). The  $K$  row denotes the model complexity of the GLTM (one of several values used by [15]), where  $K_{\text{init}}$  is the start value for the GLTFM. Where the number of iterations ( $H$ ) notes how many independent runs we used for either model.

Parameter	#1	#2
$F$ (outer)	100	250
$G$ (inner)	250	250
GLTM	250	500
$K_{\text{init}}$ (GLTFM)	2	2
$K$ (GLTM)	10	15
$H$ (GLTFM)	20	20
$\alpha$	4	4

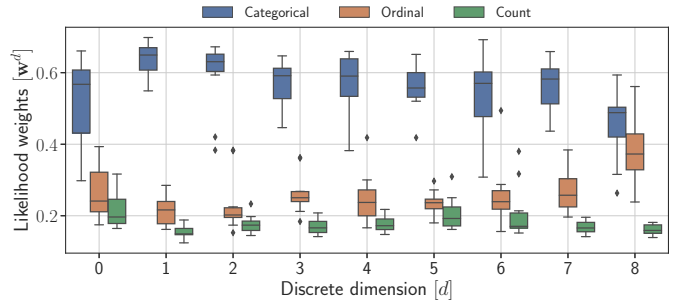
### A. Type inference on real data

To evaluate our model, we first consider a simple task, akin to the one tackled by [15, §4], wherein we seek to infer the statistical type of the columns in the **German** and **Breast** data sets described in table IV. In this exercise we are only considering the discrete columns with discrete statistical types as these are typically more difficult to infer. Specifically we seek to infer if an attribute takes values in a finite unordered set

(categorical) e.g. {‘white’, ‘black’, ‘yellow’}; takes values in a finite ordered set (ordinal) e.g. {‘small’, ‘average’, ‘large’} or if it takes values in the natural numbers (count) i.e.  $\{0, \dots, \infty\}$ . Furthermore, we use the same preprocessing steps as [16] and hence remove binary features (again because these are easy to infer). Results are shown in fig. 5.



(a) German.



(b) Breast.

Fig. 5: Distribution of the inferred likelihood weights  $w^d$  when the GLTFM is applied to two real data sets.

The GLTFM correctly identifies the statistical type of the attributes in the Breast data set. The results in fig. 5a are more interesting since the inferred discrete variables are a mix of ordinal and categorical types. For example  $\{d = i \mid i = 7, 13, 14, 15\}$  in fig. 5a are found to be of ordinal type. This is commensurate with what was found by [15, table 2].

### B. Comparing the GLTM to the GLTFM

We apply the GLTFM and the GLTM to the **Wine**, **Abalone** and **Dermatology** data sets. Herein, we demonstrate, over the course of the model iterations, how the GLTFM (typically) produces a better posterior estimate of the statistical type.

a) *Wine*: The red wine data set results from chemical analysis of wines grown in the same region in Italy but derived from three different cultivars. The data set has 14 attributes, mixed continuous and discrete. For our below experiments we used set 1 in table V, with results shown in fig. 7. The GLTFM is exploring the model complexity as shown in fig. 7a and fig. 7c. In the case of the ‘Cultivar’ attribute, both models infer the wrong type, where the true type is count. This can be explained by the small (finite) number of values that the attribute takes, are not enough for an accurate type inference. Clearly model complexity exploration does not help with

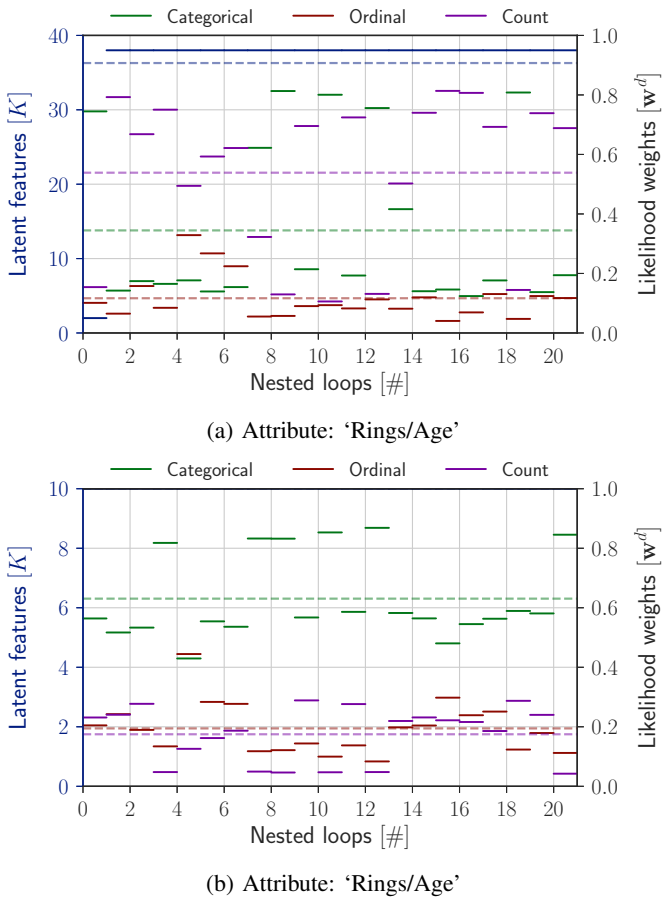


Fig. 6: Results of running the GLTFM (top) and the GLTM (bottom) on the **abalone** dataset using experimental set 1. Shown are results for one discrete variable.

this attribute. The GLTFM fares better with the ‘Magnesium’ attribute which it correctly identifies as a count type, but which the GLTM maintains holds a categorical statistical type.

b) *Dermatology*: In this data set we also compare the average test log-likelihood per observation evaluated on a held-out set containing 10% of the observations. This study was conducted both with experimental set 1 and 2 defined in table V. We again consider discrete variables. This time, both models are in broad agreement w.r.t. the inferred type of the attributes. For both experimental sets, the GLTFM settles down on a posterior estimate of  $K = 4$ , moreover the displays comparable performance (see fig. 8c and fig. 8f) to the baseline GLTM, whilst inferring both model complexity and statistical types. Broadly both models achieve the same held-out log-likelihood, but importantly the GLTFM does so whilst also maintaining a lower model-complexity.

c) *Abalone*: Herein the GLTM has a model complexity set to  $K = 10$  (inspired by the discussion in [15, §4.1]) but which renders an erroneous inference w.r.t. the ‘Rings/Age’ attribute – which, according to the dataset details, is expected to be count data. The failure of the GLTM, for this type of attribute, has previously been discussed in [15, §4.2]. The

GLTFM finds the correct type (albeit with a large likelihood weight spread) but at the expense of a large model complexity ( $K = 38$ , all experiments were truncated at  $K_{\max} = 40$ ).

## VI. CONCLUSION

In this paper we have presented a general model suitable for the analysis of heterogeneous data. The GLTFM combines the benefits of type inference and latent variable modelling, to enable as few inference resources to be used as possible. We overcome the limitations of the GLFM by learning column types directly from the data. We overcome the limitations of the GLTM by learning model complexity from the data, whilst also maintaining a binary feature matrix, to promote interpretability of the model.

## REFERENCES

- [1] D. Dheeru and E. Karra Taniskidou, “UCI machine learning repository,” 2017.
- [2] F. Doshi, K. Miller, J. Van Gael, and Y. W. Teh, “Variational inference for the indian buffet process,” in *Artificial Intelligence and Statistics*, 2009, pp. 137–144.
- [3] Z. Ghahramani and T. L. Griffiths, “Infinite latent feature models and the indian buffet process,” in *Advances in neural information processing systems*, 2006, pp. 475–482.
- [4] J. M. Hernández-Lobato, J. R. Lloyd, D. Hernández-Lobato, and Z. Ghahramani, “Learning the semantics of discrete random variables: Ordinal or categorical,” in *NIPS Workshop on Learning Semantics*, 2014.
- [5] D. Knowles and Z. Ghahramani, “Infinite sparse factor analysis and infinite independent components analysis,” in *International Conference on Independent Component Analysis and Signal Separation*. Springer, 2007, pp. 381–388.
- [6] M. Maurits, T. Huizinga, S. Raychaudhuri, M. Reinders, E. Karlson, E. van den Akker, and R. Knevel, “Ab1282 a big-data approach to electronic health record data—using dimensionality reduction and clustering techniques to study longitudinal relationships between diseases,” 2019.
- [7] A. Mnih and R. R. Salakhutdinov, “Probabilistic matrix factorization,” in *Advances in neural information processing systems*, 2008, pp. 1257–1264.
- [8] J. H. Park, H. E. Cho, J. H. Kim, M. Wall, Y. Stern, H. Lim, S. Yoo, H.-S. Kim, and J. Cha, “Electronic health records based prediction of future incidence of alzheimers disease using machine learning,” *bioRxiv*, p. 625582, 2019.
- [9] F. Ruiz, I. Valera, C. Blanco, and F. Perez-Cruz, “Bayesian nonparametric modeling of suicide attempts,” in *Advances in neural information processing systems*, 2012, pp. 1853–1861.
- [10] F. J. Ruiz, I. Valera, C. Blanco, and F. Perez-Cruz, “Bayesian nonparametric comorbidity analysis of psychiatric disorders,” *Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1215–1247, 2014.
- [11] R. Salakhutdinov and A. Mnih, “Bayesian probabilistic matrix factorization using markov chain monte carlo,” in *Proceedings of the 25th international conference on Machine learning*. ACM, 2008, pp. 880–887.
- [12] E. Salazar, M. Cain, E. Darling, S. Mitroff, and L. Carin, “Inferring latent structure from mixed real and categorical relational data,” *arXiv preprint arXiv:1206.6469*, 2012.
- [13] I. Valera, M. F. Pradier, M. Lomeli, and Z. Ghahramani, “General Latent Feature Models for Heterogeneous Datasets,” *ArXiv e-prints*, Jun. 2017.
- [14] I. Valera and Z. Ghahramani, “General table completion using a bayesian nonparametric model,” in *Advances in Neural Information Processing Systems*, 2014, pp. 981–989.
- [15] —, “Automatic discovery of the statistical types of variables in a dataset,” in *International Conference on Machine Learning (ICML)*, Sydney, 2017.
- [16] A. Vergari, A. Molina, R. Peharz, Z. Ghahramani, K. Kersting, and I. Valera, “Automatic bayesian density analysis,” in *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 33, 2019, pp. 5207–5215.
- [17] S. A. Williamson, S. N. MacEachern, and E. P. Xing, “Restricting exchangeable nonparametric distributions,” in *Advances in Neural Information Processing Systems*, 2013, pp. 2598–2606.

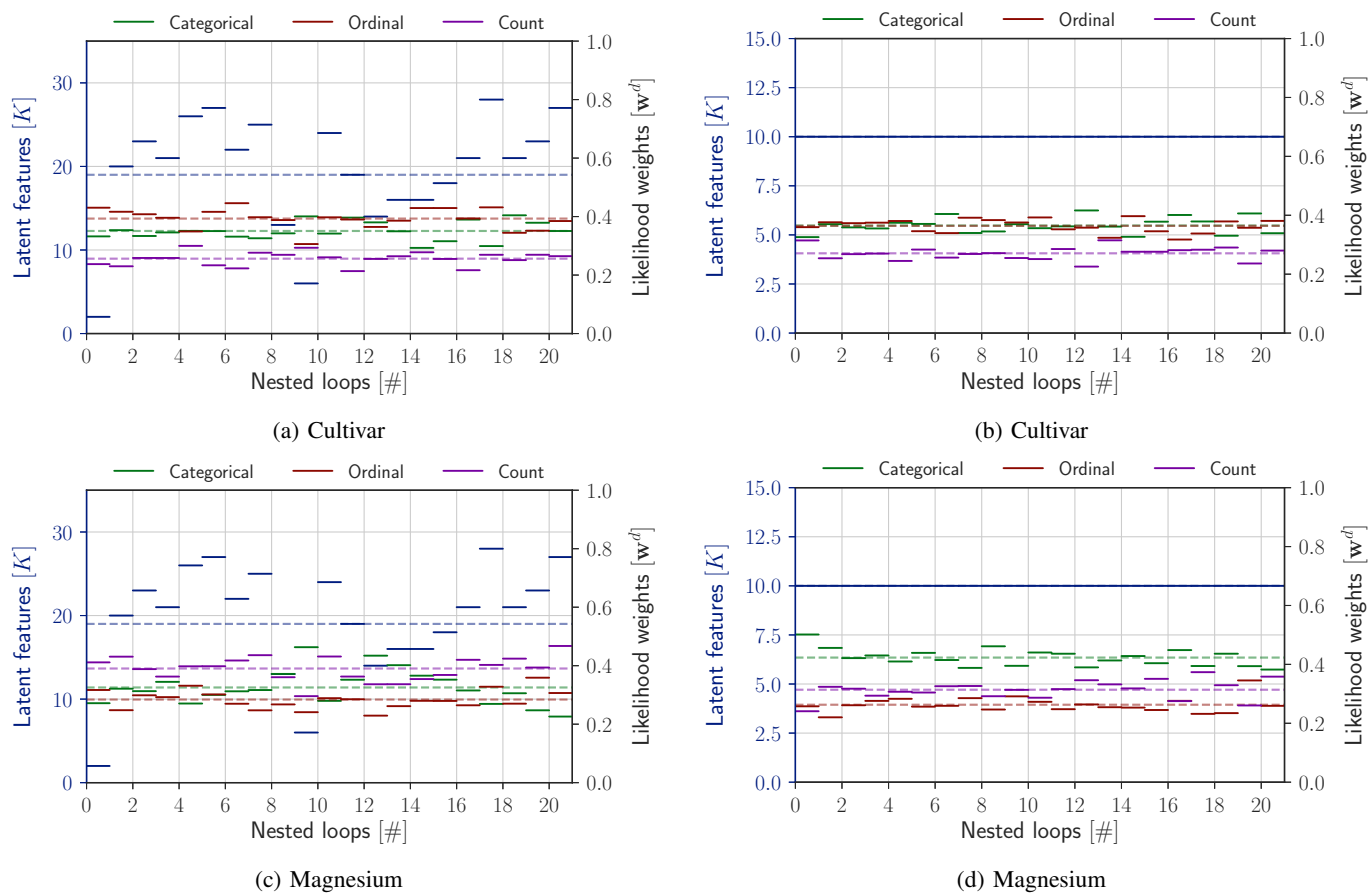


Fig. 7: The GLTFM (left) and the GLTM (right) applied to the **wine** dataset [1] using experimental set 1. Results regard both the discrete variables in the data set. Depicted is the expected values (--) for the latent number of features and the likelihood weights for both discrete variables in the dataset, over the number of iterations  $H$  (see eq. 1).

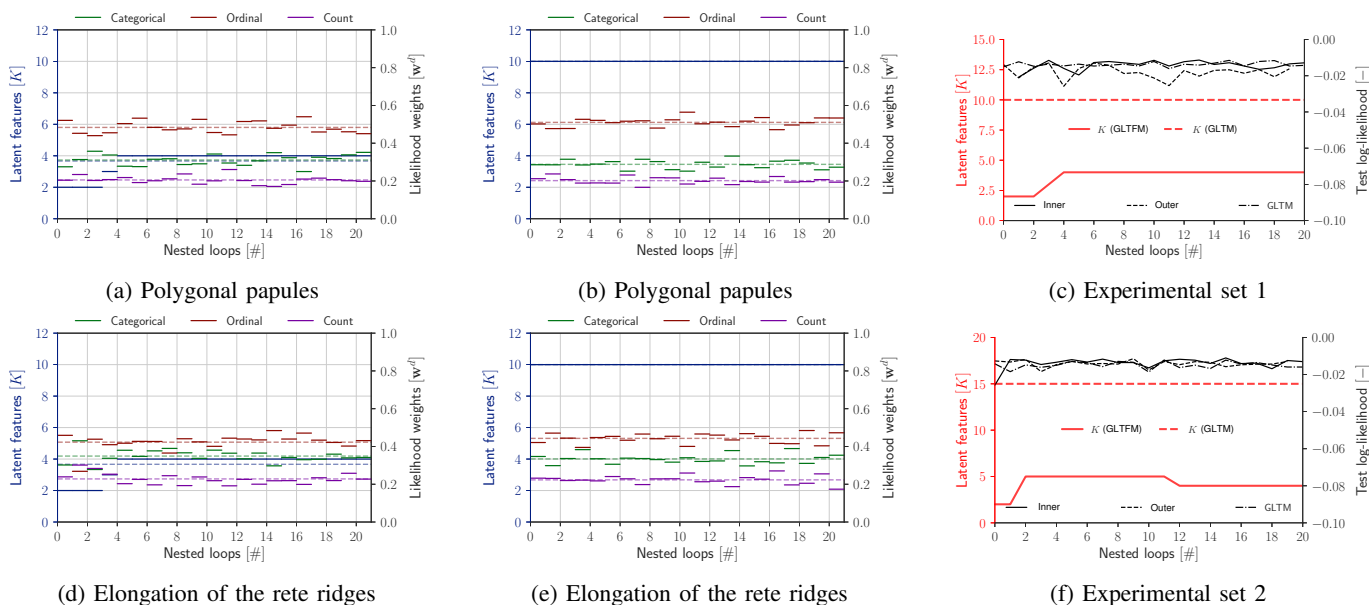


Fig. 8: The GLTFM (left) and the GLTM (centre) applied to the **Dermatology** dataset using experimental set 1. Shown are the results from inference over some of the discrete variables in the dataset. On the right we show a comparison of the average test log-likelihood per observation evaluated on a held-out set containing the 10% of the observations.