Latent Classification Models for Binary Data

Langseth, Helge; Nielsen, Thomas Dyhre

Published in:
Pattern Recognition

DOI (link to publication from Publisher):
10.1016/j.patcog.2009.05.002

Publication date:
2009

Document Version
Early version, also known as pre-print

Latent Classification Models for Binary Data

Helge Langseth [a] and Thomas D. Nielsen [b]

[a] *Department of Information and Computer Sciences, Norwegian University of Science and Technology, N-7491 Trondheim, Norway*

[b] *Department of Computer Science, Aalborg University, DK-9220 Aalborg, Denmark*

Abstract

One of the simplest, and yet most consistently well-performing set of classifiers is the naïve Bayes models (a special class of Bayesian network models). However, these models rely on the (naïve) assumption that all the attributes used to describe an instance are conditionally independent given the class of that instance. To relax this independence assumption, we have in previous work proposed a family of models, called *latent classification models* (LCMs). LCMs are defined for continuous domains and generalize the naïve Bayes model by using latent variables to model class-conditional dependencies between the attributes. In addition to providing good classification accuracy, the LCM model has several appealing properties, including a relatively small parameter space making it less susceptible to over-fitting. In this paper we take a first-step towards generalizing LCMs to hybrid domains, by proposing an LCM model for domains with binary attributes. We present algorithms for learning the proposed model, and we describe a variational approximation-based inference procedure. Finally, we empirically compare the accuracy of the proposed model to the accuracy of other classifiers for a number of different domains, including the problem of recognizing symbols in black and white images.

Key words: classification, binary images, Bayesian networks, variational inference
1991 MSC: 62H30, 68T37, 70G75, 62H35

Email addresses: helgel@idi.ntnu.no (Helge Langseth), tdn@cs.aau.dk (Thomas D. Nielsen).

1 Introduction

Classification is the task of predicting the class of an instance from a set of attributes describing that instance, i.e., to apply a mapping from the attribute space into a predefined set of classes. When learning a classifier we seek to find such a mapping based on a database of labelled instances. Classifier learning, which has been an active research field over the last decades, can therefore be seen as a model selection process where the task is to find the single model, from some set of models, with the highest classification accuracy.

One of the simplest and yet still well-performing set of classifiers is the naïve Bayes model [1,2]. Generally, in the naïve Bayes models all attributes are assumed to be conditionally independent given the class variable. This assumption is clearly violated in many real world domains, and it has inspired several extensions of the basic model. These extensions can roughly be characterized as either (*i*) using a set of models that admits a more general correlation structure or (*ii*) relying on a preprocessing of the data. As an example of the former, Friedman et al. [3] propose the tree augmented naïve Bayes (TAN) model; in the TAN framework, each attribute is allowed to have at most one parent besides the class variable. Another approach is to preprocess the data before learning the classifier s.t. the transformed data abides to the independence assumptions of the model class. This approach has been pursued by e.g. Bressan and Vitria [4], who consider applying a class conditional independent component analysis [5] and then using a naïve Bayes model on the transformed data.

Transforming the data to fit the independence assumptions needs not be performed as a filtering step (independent of the classifier), but can instead be integrated into the model structure. This is, for example, the approach implemented in the framework of *latent classification models* [6]. In a latent classification model (LCM), the conditional dependencies among the (continuous) attributes are encoded using latent variables, which allow the model to be interpreted as a combination of a naïve Bayes model and a mixture of factor analyzers [7]. Besides providing a high classification accuracy, the parameter space of LCMs is also relatively small making the model less susceptible to overfitting [8]. Moreover, the use of latent variables provides a well-defined semantics and a transparent model structure that admits analysis.

In this paper we propose an extension to our previous work on latent classification models [6]. Compared to LCMs, which target domains containing continuous attributes only, the present paper introduces *binary LCM* (bLCM), which takes a first step towards general hybrid domains by focusing on do-

mains with binary attributes. More specifically, the bLCM model shares the latent structure of the original LCM model, but instead of focusing on continuous attributes the bLCM model assumes all attributes to be binary. We describe algorithms for doing both learning and inference in bLCMs, and we present promising results from a comparison of the classification accuracy of the proposed classifier and the accuracy of other classifiers in these types of domains.

2 Notation

In the context of classification, we shall use $\{T_1, \ldots, T_n\}$ to denote the attributes describing instances to be classified; when considering continuous domains we let $\mathrm{sp}(T_i) = \mathbb{R}$ and when focusing on binary domains we let $\mathrm{sp}(T_i) = \{0, 1\}$, for all $1 \leq i \leq n$. Furthermore, we shall use Y to denote the (discrete) class variable, where $\mathrm{sp}(Y)$ is the set of possible classes (for notational convenience we also assume that $\mathrm{sp}(Y) = \{1, 2, \ldots, |\mathrm{sp}(Y)|\}$).

When doing classification in a probabilistic framework, a Bayes optimal classifier will classify a new instance $\boldsymbol{t} = (t_1, \ldots, t_n)$ to class y^* according to

$$y^* = \arg \min_{y \in \mathrm{sp}(Y)} \sum_{y' \in \mathrm{sp}(Y)} L(y, y') P(y' | \boldsymbol{t}), \tag{1}$$

where $L(\cdot, \cdot)$ is the loss-function, see e.g. [8–10]. An example of such a loss-function is the 0/1-loss, where $L(\cdot, \cdot)$ is defined s.t. $L(y, y') = 0$ if $y = y'$ and 1 otherwise. When learning a probabilistic classifier, the task is therefore to learn the probability distribution $P(Y = y | \boldsymbol{T} = \boldsymbol{t})$ from a set of N labeled training samples $\mathcal{D}_N = \{\boldsymbol{D}_1, \ldots, \boldsymbol{D}_N\}$, where $\boldsymbol{D}_i = (t_1^i, \ldots, t_n^i, y^i)$ is a configuration over the attributes together with a class label.

3 Binary Latent Classification Models

As mentioned in Section 1, one approach for handling the conditional dependencies between the attributes is to perform a data transformation within the classification model. For example, the LCM model [6] embeds a factor analysis (FA) model, which makes a dimensionality reduction based on the covariance structure of the data; the data relevant for classification is thus summarized by a collection of continuous latent variables. The model proposed in [6] is restricted to continuous domains, but in this paper we take a first step towards

a generalization to hybrid domains. Particularly, we shall consider the case where all attributes are binary, resulting in the so-called bLCM model.

In what follows we give a brief description of the LCM model, and after that the bLCM model is introduced.

3.1 The LCM Model

The LCM model can roughly be seen as combining an FA model with a naïve Bayes model. The FA model describes the attributes, $\boldsymbol{T}$, using a q-dimensional vector of *factor variables* $\boldsymbol{X}$ (with $q \leq n$) and by assuming the generative model

$$\boldsymbol{T} = \boldsymbol{W}\boldsymbol{X} + \boldsymbol{\epsilon},$$

where $\boldsymbol{W}$ is the regression matrix. In its most common setting, the FA assumes $\boldsymbol{X} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{I})$, and $\boldsymbol{\epsilon} \sim \mathcal{N}(\boldsymbol{0}, \boldsymbol{\Theta})$ is an n-dimensional random variable with diagonal covariance matrix $\boldsymbol{\Theta}$, leading to the assumption that $\boldsymbol{T}$ follows a Gaussian distribution as well. In this model, the factor variables model the dependencies among the attributes, and $\boldsymbol{\epsilon}$ is interpreted as the sensor noise associated with the attributes. In the LCM setting, the FA model was extended to suit classification by specifying a class-conditional prior distribution for the latent variables $\boldsymbol{X}$, i.e., $\boldsymbol{X} \,|\, \{Y = y\} \sim \mathcal{N}(\boldsymbol{\mu}_y, \boldsymbol{\Gamma}_y)$, where $\boldsymbol{\Gamma}_y$ is a diagonal matrix (see Fig. 1).

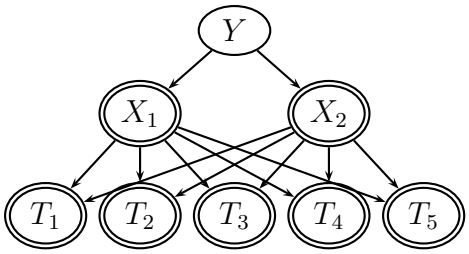

Fig. 1. The Bayesian network representation of an LCM with $n = 5$ attributes and $q = 2$ latent variables. Note that all latent variables as well as the attributes are continuous (indicated by double circles).

3.2 The bLCM Model

The factor analysis setup used in [6] has many desirable properties, including a relative small parameter space and a robust and simple parameter estimation procedure (based on the maximum likelihood principle [11]). When

generalizing the LCM model to discrete domains we still assume *continuous* latent variables, resulting in a so-called *latent trait* model [12]. [1] That is, we assume that there exists a vector $\boldsymbol{X}$ of latent variables and local probability distributions $P(t_i|\boldsymbol{X} = \boldsymbol{x})$ such that

$$P(\boldsymbol{t}, y) = P(y) \int_{\mathbb{R}^q} f(\boldsymbol{x} \,|\, y) \prod_{i=1}^{n} P(t_i|\boldsymbol{x}) \, d\boldsymbol{x}.$$

Analogously to LCMs, a bLCM can be seen as combining a latent trait model with a naïve Bayes model. Hence, the factor variables $\mathcal{X}$ appear as children of the class variable in the graphical representation of the model (see Fig. 2). More specifically, the variables can be partitioned into three disjoint subsets: $\{Y\}$ is the class variable, $\mathcal{T}$ is the set of binary attributes, and $\mathcal{X}$ is the set of latent variables ($\mathcal{X}$ has the same role as the factor variables in an FA). In a bLCM the class variable appear as root, $\mathcal{T}$ constitute the leaves with only latent variables as parents, and the latent variables are all internal having the class variable as parent and the attributes as children. Note that in a bLCM, the latent variables are conditionally independent given the class, but marginally dependent (see Fig. 2).

For the quantitative part of the bLCM, we assume that:

- The class variable, Y, follows a multinomial distribution, i.e., $P(Y = j) = p_j$, where $1 \leq j \leq |\mathrm{sp}\,(Y)|$, $p_j \geq 0$ and $\sum_{j=1}^{|\mathrm{sp}(Y)|} p_j = 1$.
- Conditionally on $Y = j$ the latent variables $\boldsymbol{X}$, follow a Gaussian distribution with $\mathbb{E}\,[\boldsymbol{X} \,|\, Y = j] = \boldsymbol{\mu}_j$ and $\mathrm{Cov}\,(\boldsymbol{X} \,|\, Y = j) = \boldsymbol{\Gamma}_j$. Moreover, it follows from the model structure that $\boldsymbol{\Gamma}_j$ has to be diagonal (meaning that $X_k \perp\!\!\!\perp X_l \;|\, \{Y = j\}$, for all $k \neq l$ and for all $j = 1, \ldots, |\mathrm{sp}\,(Y)|$).
- For each T_i there exists a vector $\boldsymbol{w}_i \in \mathbb{R}^q$ and a parameter $b_i \in \mathbb{R}$ that together take a vector of (unobservable) latent variables and maps it to the log-odds of the (observable) attribute:

$$\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{x} + b_i = \log \left(\frac{P(T_i = 1 \,|\, \boldsymbol{x})}{P(T_i = 0 \,|\, \boldsymbol{x})} \right).$$

  We define $g(v) = (1 + \exp(-v))^{-1}$, and have that

$$P(T_i = t_i \,|\, \boldsymbol{x}) = g\left((2t_i - 1)(\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{x} + b_i)\right),$$

  for $t_i \in \{0, 1\}$.

[1] Some researchers assume the latent variables to be discrete and thereby define a *discrete FA*, see e.g. [13].

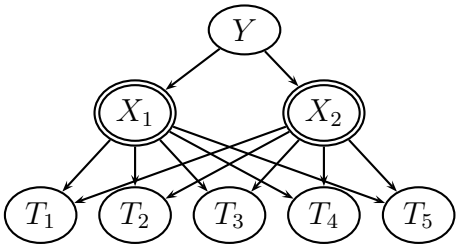

Fig. 2. A graphical representation of a bLCM with $d = 5$ attributes and $q = 2$ latent variables. Note that all latent variables are continuous, whereas the attributes are binary.

3.3 The Mixture Model

When we consider the bLCM model definition above, it is important to emphasize that the attributes are assumed conditionally independent of the class variable given the factor variables ($\boldsymbol{T} \perp\!\!\!\perp Y | \boldsymbol{X}$), and that the same mappings, $\boldsymbol{w}_i$, from the latent space to the attribute space is used for all classes. Thus, the relation between the class variable and the attributes is conveyed by the latent variables only, i.e., the latent variables summarize *all* the information from the attributes which is relevant for classification. Unfortunately, as we shall see in the following example, these independence assumptions may severely restrict the expressive power of the model. As described in Section 4, the marginal distribution of $\boldsymbol{X}$ cannot be expressed in closed form, thus making a formal analysis of model expressiblity difficult. Examples 1 and 2 are therefore used only as qualitative motivation for the forthcoming definition of the full model.

Example 1 *To illustrate the expressiveness of the bLCM model, we sample images of the digits* 0*,* 1*,* 6*, and* 7*. We do this by first training a bLCM structure on data consisting of binary images of these numbers, and afterwards we sample from the learned model (a method for learning bLCMs is described in Section 5). Consider the re-sampled* USPS *database [14], prepared by Rasmussen and Williams [15], which consists of* 16×16 *grey-scale images of handwritten digits. We have binarized this data, and used the images of the digits* 0*,* 1*,* 6 *and* 7 *(the rest were discarded) to learn a model with* $q = 35$ *latent variables and* $n = 256$ *binary attributes, one attribute for each pixel. To make the problem a bit difficult we specified two classes by grouping together images of digits* 0 *and* 1*, and* 6 *and* 7*, respectively. Examples of the samples generated from the learned model can be seen in the upper row of Fig. 3. From these samples we clearly see that the bLCM model has poor generative properties for the present example; several sample images are not readable by humans.*

In order to extend the expressibility of the bLCM model we propose a natural generalization, termed *mixture bLCMs* or mbLCMs. Intuitively, the mbLCM

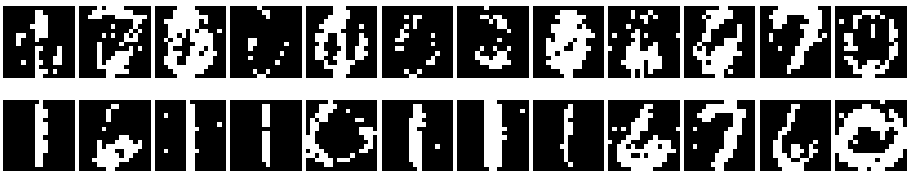

Fig. 3. Samples from a bLCM model with the two classes $\{0, 1\}$ and $\{6, 7\}$. The images in the upper line are generated from a standard bLCM, the bottom row shows images generated from a bLCM with two mixture components. The same random seed was used for both images in any given column, so comparing two images in a column gives an impression of the expressibility of the two models.

can be interpreted as integrating a naïve Bayes model with either *i*) a mixture of latent trait models, or *ii*) a combined latent trait and latent class model. More formally, in an mbLCM we have a mixture variable M so that for each mixture component $M = m$ and attribute T_i there exist a vector $\boldsymbol{w}_{i,m} \in \mathbb{R}^q$ and a parameter $b_{i,m} \in \mathbb{R}$ that together map from the latent variables to the log-odds of the attribute:

$$\boldsymbol{w}_{i,m}^{\mathrm{T}}\boldsymbol{x} + b_{i,m} = \log\left(\frac{P(T_i = 1 \,|\, \boldsymbol{x}, M = m)}{P(T_i = 0 \,|\, \boldsymbol{x}, M = m)}\right).$$

Based on the specification above, the mbLCM defines a partitioning of the variables into four disjoint subsets: $\{Y\}$ is the class variable, $\{M\}$ is the mixture variable, $\mathcal{X}$ is the set of attributes and $\mathcal{T}$ is a set of latent variables. The structure of a mbLCM is identical to the structure of the standard bLCM, except that we also have the mixture variable M as an internal node having Y as parent and with all variables in $\mathcal{X}$ as children (see Fig. 4). Moreover, we assume that the mixture variable, M, follows a multinomial distribution, i.e., $P(M = m|Y = j) = p_{m,j}$, where $1 \leq m \leq |\mathrm{sp}\,(M)|$, $p_{m,j} \geq 0$ and $\sum_{m=1}^{|\mathrm{sp}(M)|} p_{m,j} = 1$, for all $1 \leq j \leq |\mathrm{sp}\,(Y)|$.

Observe that the mbLCM model is a proper generalization in the sense that with $|\mathrm{sp}\,(M)| = 1$ an mbLCM reduces to a simple bLCM. Thus, in the remainder of this paper, when referring to a bLCM we mean the general mixture model that includes the simple bLCM model as a special case.

Example 2 *In order to illustrate the impact of introducing the mixture variable, consider again the sampling procedure described in Example 1. For this example we have learned a bLCM with two mixture components and* 35 *latent variables. Images from this model were sampled, and the results can be seen in the lower row of Fig. 3. The results suggests that mixture models are required if we want a sufficiently expressive class of generative models.*

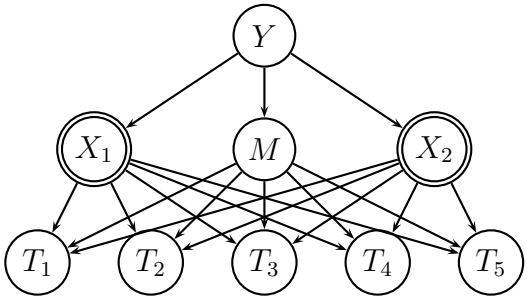

Fig. 4. A graphical representation of a mixture bLCM with $n = 6$ attributes and $q = 2$ latent variables. Note that if $|\mathrm{sp}\,(M)| = 1$, then the model simply corresponds to a standard bLCM.

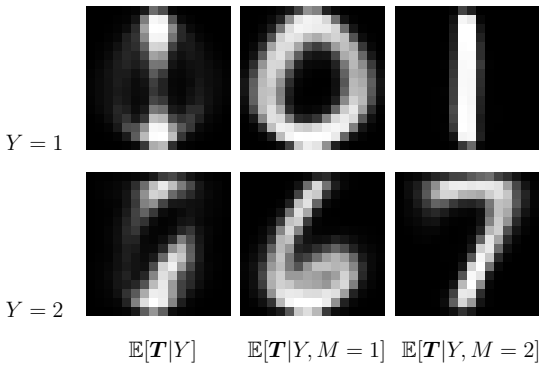

Fig. 5. The expected values for the attributes after learning a bLCM with 25 latent variables and 2 mixture components. The first row shows the results for the first class (digits 0 and 1), the second row gives results for the class containing digits 6 and 7. The first column gives the expected values for the attributes "overall", whereas the second and third column give the same results for each of the two mixture components. We can clearly see that the mixtures are used to model the separate digits making up each class. Note that this part of the learning is done unsupervised, as each image is only labelled by its class and not its digit.

We further examine the bLCM by calculating the expected values for the attributes conditioned on the class and the mixture variable. The results are shown in Fig. 5, which clearly illustrates that the mixture variable accounts for the different digits making up each class. This points towards a different view on mixture bLCMs corresponding to Ghahramani and Hinton's interpretation of mixtures of FAs [7]: A mixture of factor analyzers concurrently performs clustering (the mixture model) and, within each cluster, local dimensionality reduction (factor analysis). Analogously, we can interpret the bLCM model as concurrently performing clustering and, within each cluster, local classification.

With the introduction of the mixture component, we can now show that the mbLCM can approximate any distribution over $\{Y\} \cup \mathcal{T}$ arbitrarily well.

Proposition 1 *Assume that Y is distributed as $P(Y = i) = p_i$ for $i \in \{1, \ldots, |\mathrm{sp}(Y)|\}$, and let $P(T_1, \ldots, T_d = t_1, \ldots, t_d \,|\, Y = y)$ be given. Then the joint distribution for $(Y, \boldsymbol{T})$ can be approximated arbitrarily well by a bLCM model.*

The proof is constructive, i.e., we will show how to construct a bLCM, which can approximate any probability distribution $P(\boldsymbol{T} = \boldsymbol{t}|Y = y)$ arbitrarily well. First, however, we need some notation: The idea of the proof is to let the mixture variable have one state for each configuration of $\boldsymbol{T}$, i.e., $\mathrm{sp}(M) = \{0, 1, 2, \ldots, 2^d - 1\}$. Each state of M maps to a specific configuration over $\boldsymbol{T}$; specifically, we use the binary representation of the state of M as the configuration over $\boldsymbol{T}$. If, for instance, $d = 5$, then M has $2^5 = 32$ states. The configuration $\boldsymbol{t} = (1, 0, 0, 1, 0)$ is represented by the 18th state of M, as `10010` is the binary representation of 18. We use the notation $\boldsymbol{t} \leftrightarrow m$ to denote that a state m coincides with the configuration $\boldsymbol{t}$, so in our example we have that $\{M = 18\} \leftrightarrow \{\boldsymbol{t} = (1, 0, 0, 1, 0)\}$.

Proof 1 *Consider a bLCM, where:*

- *The graphical structure consist of one latent variable X, d binary attributes, and a mixture variable M.*
- *The size of the state space of the mixture variable is $|\mathrm{sp}(M)| = 2^d$*
- *Conditional on $Y = y$, X follows a Gaussian distribution with $\mu_{X\,|\,y} = 1$ and $\sigma^2_{X\,|\,y} = \epsilon$, for all $y \in \mathrm{sp}(Y)$.*
- *For a fixed m and $\boldsymbol{t} \leftrightarrow m$ we set $w_{i,m} = \eta > 0$ if $t_i = 1$ in $\boldsymbol{t}$ and $w_{i,m} = -\eta$ otherwise.*

By marginalizing out X from the distribution specified by the bLCM we get:

$$
\begin{aligned}
P(\boldsymbol{t}\,|\,y) &= \int_x P(\boldsymbol{t}\,|\,x, y)\, f(x\,|\,y)\, dx \\
&= \int_x \sum_m P(\boldsymbol{t}\,|\,x, y, m) P(m\,|\,y) f(x\,|\,y)\, dx \\
&= \int_x \sum_m P(\boldsymbol{t}\,|\,x, m) P(m\,|\,y) f(x\,|\,y)\, dx \\
&= \sum_m P(m\,|\,y) \int_x \prod_{i=1}^{n} P(t_i\,|\,x, m) f(x\,|\,y)\, dx \\
&= \sum_m P(m\,|\,y) P(\boldsymbol{t}\,|\,m, y) \qquad (2)
\end{aligned}
$$

Next, let $\eta \to \infty$ and $\epsilon \to 0$ in Equation (2), then $P(\boldsymbol{T} = \boldsymbol{t}\,|\,Y = y, M =$

$m) \to 1$ *if and only if* $\boldsymbol{t} \leftrightarrow m$ *and* 0 *otherwise. Thus, we have*

$$\begin{aligned} P(\boldsymbol{t} \,|\, Y = y) &= \sum_{m'} P(\boldsymbol{t} \,|\, Y = y, M = m') P(M = m' \,|\, Y = y) \\ &= P(M = m \,|\, Y = y, m \leftrightarrow \boldsymbol{t}) \end{aligned}$$

in the limit. The last step is to define $P(M = m \,|\, Y = y)$*, and since we have as many states of* M *as there are configurations over* $\boldsymbol{t}$*, we can choose* $P(M = m \,|\, Y = y) = P(\boldsymbol{t}|y, \boldsymbol{t} \leftrightarrow m)$*, and the result then follows.* □

4 Inference in bLCM models

Making classification in a bLCM amounts to calculating $P(y \,|\, \boldsymbol{t})$ (confer Equation (1)). As $P(y \,|\, \boldsymbol{t}) = P(y, \boldsymbol{t})/P(\boldsymbol{t})$, where $P(\boldsymbol{t})$ is independent of y (and therefore can be regarded as a normalization constant), we will in the following focus on calculating

$$P(\boldsymbol{t}, y) = P(y) \int_{\mathbb{R}^q} \left\{ \prod_{i=1}^{d} P(t_i \,|\, \boldsymbol{x}) \right\} f(\boldsymbol{x}|y) d\boldsymbol{x}, \tag{3}$$

for a bLCM with a single mixture component, i.e., having $|\mathrm{sp}\,(M)| = 1$.

It is, however, well known [16,17] that this integral cannot be calculated analytically. In the following we therefore derive a variational approximation [16–19] for this expression.

As a starting-point, consider the integral

$$\begin{aligned} P(\boldsymbol{t} \,|\, y) &= \int_{\mathbb{R}^q} \left\{ \prod_{i=1}^{d} P(t_i \,|\, \boldsymbol{x}) \right\} f(\boldsymbol{x} \,|\, y) d\boldsymbol{x} \\ &= \int_{\mathbb{R}^q} \left\{ \prod_{i=1}^{d} P(t_i \,|\, \boldsymbol{x}) \right\} \left\{ \prod_{j=1}^{q} \frac{1}{\sqrt{2\pi}\sigma_{j,y}} \exp\left(-\frac{(x_j - \mu_{j,y})^2}{2\sigma_{j,y}^2} \right) \right\} d\boldsymbol{x}, \end{aligned} \tag{4}$$

where the second equality follows when we assume that $X_j \,|\, \{Y = y\} \sim \mathcal{N}(\mu_{j,y}, \sigma_{j,y}^2)$ and that $X_k \perp\!\!\!\perp X_l \,|\, Y$ for $k \neq l$. This likelihood function cannot be calculated in closed form, but fortunately Tipping [17] showed how a similar model can be handled with a variational approximation. Following his

procedure, we introduce

$$\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i) = g(\xi_i) \exp((A_i - \xi_i)/2 + \lambda(\xi_i)(A_i^2 - \xi_i^2)), \tag{5}$$

where

$$A_i = (2t_i - 1)(\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{x} + b_i) \quad \text{and} \quad \lambda(\xi_i) = \frac{\exp(-\xi_i) - 1}{4\xi_i(1 + \exp(-\xi_i))}.$$

The function $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i)$ is a variational approximation to $P(t_i \,|\, \boldsymbol{x})$, hence $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i) \leq P(t_i \,|\, \boldsymbol{x})$ for all ξ_i and $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i) = P(t_i \,|\, \boldsymbol{x})$ for some particular choice of ξ_i. It can easily be verified that equality is obtained if and only if $\xi_i = (2t_i - 1)(\boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{x} + b_i)$.

Example 3 *The left pane of Fig. 6 shows the logistic function together with three variational approximations defined by $\xi = 1, 2, 3$. From the example, we see that for a given value of $A = \boldsymbol{w}^T \boldsymbol{x} + b$ the quality of the variational approximation depends on the chosen value of ξ. For instance, with $\xi = 1$, the approximation is accurate for $A \leq 1.5$. The right figure shows the variational error for $\tilde{P}(T = 1 \,|\, X, \xi = 1)$ as a function of A (dotted line). Using $N(0, 1)$ as prior distribution for X, the figure also shows the resulting variational approximation for the posterior distribution for X given $T = 1$ (specified below).* [2]

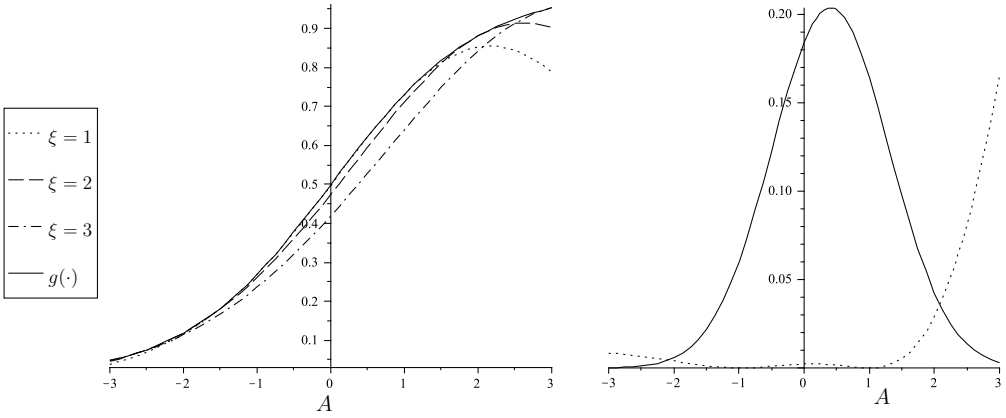

Fig. 6. The left pane shows $P(T = 1 \,|\, X)$ (solid line) and $\tilde{P}(T = 1 \,|\, X, \xi)$ as functions of A; for the variational approximations we have used $\xi = 1, 2, 3$. The right pane shows the variational error for $\tilde{P}(T = 1 \,|\, X, \xi = 1)$ as a function of A (dotted line), as well as the (scaled) variational approximation for the posterior distribution for X given $T = 1$.

Since the variational distribution is of a Gaussian shape (quadratic in x_j in the exponential), the variational approximation of the posterior of $\boldsymbol{X}$, $\boldsymbol{X} \,|\, \{\boldsymbol{T} = \boldsymbol{t}, Y = y, \boldsymbol{\xi}\}$, is also of this type. We use $\boldsymbol{\mu}_y^p$ and $\boldsymbol{\Gamma}_y^p$ for the expectation and

[2] The posterior approximation is scaled to fit the graph.

variance of this posterior. The updated parameters can be found after some algebraic manipulation (see Appendix A):

$$\boldsymbol{\Gamma}_y^p = \left[\boldsymbol{\Gamma}_y^{-1} - 2\sum_{i=1}^{d} \lambda(\xi_i)\boldsymbol{w}_i\boldsymbol{w}_i^{\mathrm{T}}\right]^{-1}; \tag{6}$$

$$\boldsymbol{\mu}_y^p = \boldsymbol{\Gamma}_y^p \left\{\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y + \sum_{i=1}^{d}\left[t_i - \frac{1}{2} + 2\lambda(\xi_i)b_i\right]\boldsymbol{w}_i\right\}, \tag{7}$$

where, $\boldsymbol{\mu}_y = (\mu_{1,y}, \ldots, \mu_{q,y})^{\mathrm{T}}$ and $\boldsymbol{\Gamma}_y = \mathrm{diag}(\sigma_{1,y}^2, \ldots, \sigma_{q,y}^2)$ are the a priori expectation and variance of $\boldsymbol{X}$ given $Y = y$.

We are also able to calculate a lower bound for the integral in Equation (4) (see Appendix B):

$$\begin{aligned} f(\boldsymbol{t} \mid y) &\geq \int_{\mathbb{R}^q} \left\{\prod_{i=1}^{d} \tilde{P}(t_i \mid \boldsymbol{x}, \xi_i)\right\} \left\{\prod_{j=1}^{q} \frac{1}{\sqrt{2\pi}\sigma_{j,y}} \exp\left(-\frac{(x_j - \mu_{j,y})^2}{2\sigma_{j,y}^2}\right)\right\} d\boldsymbol{x} \\ &= \exp\left\{-\frac{1}{2}\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y + \frac{1}{2}(\boldsymbol{\mu}_y^p)^{\mathrm{T}}(\boldsymbol{\Gamma}_y^p)^{-1}\boldsymbol{\mu}_y^p + \frac{1}{2}\log\left(\frac{|\boldsymbol{\Gamma}_y^p|}{|\boldsymbol{\Gamma}_y|}\right)\right\} \cdot \\ &\quad \exp\left\{\sum_{i=1}^{d}\left\{\log(g(\xi_i)) - \xi_i/2 + \lambda_i(b_i^2 - \xi_i^2) + \frac{1}{2}(2t_i - 1)b_i\right\}\right\}. \end{aligned} \tag{8}$$

The approximation above depends on $\boldsymbol{\xi} = (\xi_1, \ldots, \xi_d)^{\mathrm{T}}$, but since $\boldsymbol{X}$ is not observed we cannot directly calculate the value for $\boldsymbol{\xi}$ that maximizes the lower bound $\tilde{f}(\boldsymbol{t} \mid y, \boldsymbol{\xi})$. Instead we can maximize the expected complete data log-likelihood $\mathbb{E}(\log f(\boldsymbol{t}, \boldsymbol{x} \mid y, \boldsymbol{\xi}))$. It was shown by Murphy [20] that the ξ_i maximizing this expression is determined by $\xi_i^2 = \mathbb{E}\left[(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{X} + b_i)^2 | y, \boldsymbol{T}\right]$. However, since this value depends on $\boldsymbol{\Gamma}_y^p$ and $\boldsymbol{\mu}_y^p$ an iteration scheme is required, as shown in Algorithm 1.

For the initial guesses on $\boldsymbol{\Gamma}_y^p$ and $\boldsymbol{\mu}_y^p$ we simply use the prior covariance matrix and mean vector. For the initial value of $\boldsymbol{\xi}$ we follow the approach by [20]: for a data case with $Y = y$, we take $\boldsymbol{\Gamma}_y$ and $\boldsymbol{\mu}_y$ and plug them into Equation (9) to estimate $\boldsymbol{\xi}$. That is, the initial estimate is found by only taking the state of Y into account. For instance, in Example 3 we used $N(0, 1)$ as prior distribution for X, which resulted in the initial estimate $\xi = 1$. Moreover, by conditioning on $T = 1$ the iterative updating procedure above returns $\boldsymbol{\Gamma}_y^p = 0.812$ and $\boldsymbol{\mu}_y^p = 0.406$ after three iterations (see the right pane in Fig. 6).

Algorithm 1 Approximate $f(\boldsymbol{t}|y)$ using the variational approximation

1: Start with initial guesses for $\boldsymbol{\Gamma}_y^p$, and $\boldsymbol{\mu}_y^p$. $\boldsymbol{w}_1, \ldots, \boldsymbol{w}_d$ and $\boldsymbol{b}$ are assumed to be known.
2: **repeat**
3: Update values for $\boldsymbol{\xi}$ by setting

$$\begin{aligned}\xi_i &\leftarrow \sqrt{E\left[(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{X} + b_i)^2 | \boldsymbol{T}, y\right]} \\ &= \sqrt{(\boldsymbol{\mu}_y^p)^{\mathrm{T}}\boldsymbol{\mu}_y^p + \boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{\Gamma}_y^p\boldsymbol{w}_i + 2b_i\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{\mu}_y + b_i^2}\end{aligned} \tag{9}$$

4: Calculate $\boldsymbol{\Gamma}_y^p$ and $\boldsymbol{\mu}_y^p$ using the current $\boldsymbol{\xi}$ (Equations 6 and 7).
5: **until** Finished

It should also be noted that although we can always fix $\boldsymbol{\xi}$ and make the lower bound arbitrarily tight for given values of $A_i = (2t_i - 1)(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{x} + b_i)$, the lower bound is tight only *point-wise*. However, when approximating the likelihood $f(\boldsymbol{t}, y)$ variationally, we will select *one* value for ξ_i when defining $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i)$, i.e., we treat ξ_i as a constant when we integrate over $\boldsymbol{x}$. This will give us a "variational error"; to obtain equality we would have to set ξ_i equal to A_i for each $\boldsymbol{x}$ (and A_i is obviously not constant in $\boldsymbol{x}$). This is illustrated in the right pane of Fig. 6, where the total variational error can be found by integrating the error function (with the updated value for the variational parameter) using the prior distribution over X.

4.1 Inference in Mixture Models

When performing inference in a bLCM with a mixture variable M, we need to calculate

$$\begin{aligned}f(\boldsymbol{t} \,|\, y) &= \int_{\mathbb{R}^q} \sum_{m \in \mathrm{sp}(M)} P(y)P(m \,|\, y)f(\boldsymbol{x} \,|\, y)P(\boldsymbol{t} \,|\, \boldsymbol{x}, m)d\boldsymbol{x} \\ &= P(y) \sum_{m \in \mathrm{sp}(M)} P(m \,|\, y) \int_{\mathbb{R}^q} f(\boldsymbol{x} \,|\, y)P(\boldsymbol{t} \,|\, \boldsymbol{x}, m)d\boldsymbol{x}.\end{aligned}$$

Evaluating the integral is done exactly as before, except that the weight vectors are also indexed with m. Thus, when applying the variational approximation to evaluate the integral we introduce a variational parameter $\xi_{i,m}$ for each attribute T_i and for each mixture component $m \in \mathrm{sp}\,(M)$.

Finally, since the variational parameters are conditioned on the mixture variable, the updating rule in Equation (9) as well as the posterior covariance matrix (Equation (6)) and the mean vector (Equation (7)) are of exactly the same form as before. This also means that the complexity of performing infer-

ence in a bLCM increases only linearly in the number of mixture components.

5 Learning bLCM models

In this section we describe a method for learning bLCMs from data. The algorithm basically consist of two parts: a score function for evaluating the quality of a model and a search strategy for investigating the space of bLCMs.

In the proposed algorithm we score a model based on its accuracy, which is estimated using the wrapper approach [21]. That is, the score is given as the average accuracy found by applying cross-validation over the *training data*.

5.1 The general structure

In order to specify a search strategy, we first note that the space of bLCMs is defined by (*i*) the number of latent variables, (*ii*) the number of mixture components, and (*iii*) the parametrization of the probability distributions.

Thus, the learning algorithm can be divided into two parts: (*i*) a systematic approach for selecting appropriate values for q and the number of mixture components, $|\mathrm{sp}\,(M)|$, and, given such a pair of values, (*ii*) algorithms for learning the parameters in the model. More formally, a general bLCM learning algorithm can be formulated as in Algorithm 2, where appropriate values for q and $|\mathrm{sp}\,(M)|$ are selected using the wrapper approach.

Algorithm 2 Learn a bLCM classifier from a database $\mathcal{D}_N$ using the wrapper approach.

1: **for** possible values of q and $|\mathrm{sp}\,(M)|$ **do**
2: Partition the database into W wrapper folds $\mathcal{W}_1, \ldots, \mathcal{W}_W$.
3: **for** $w = 1, \ldots, W$ **do**
4: Learn a classifier from the dataset $\mathcal{D}_N \setminus \mathcal{W}_w$.
5: Calculate the accuracy on the remaining training-set $\mathcal{W}_w$.
6: **end for**
7: Score the parameter-pair $(q, |\mathrm{sp}\,(M)|)$ by the average accuracy obtained over the wrapper folds.
8: **end for**
9: Select the optimal values of q and $|\mathrm{sp}\,(M)|$.
10: **return** classifier learned with these parameters.

The parameters in the model are estimated by applying an EM-algorithm [22] for bLCMs. Unfortunately, taking direct outset in the bLCM specification is not possible, since the E-step of the algorithm requires inference in the underlying model, and as we have seen, this is not analytically available. Instead we focus on the variational approximation. By using the variational approximation we get a lower bound $\tilde{f}$ on the marginal likelihood. Thus, rather than maximizing the marginal likelihood directly, we instead maximize the expected data-complete variational log-likelihood $\mathbb{E}\log(\prod_{i=1}^{N}\tilde{f}(\cdot\,|\,\boldsymbol{D}_i))$, which is guaranteed never to decrease the marginal likelihood.

In the following we let $\vec{\boldsymbol{w}}_i$ be the vector defined by $\vec{\boldsymbol{w}}_i = [\boldsymbol{w}_i^{\mathrm{T}}, b_i]^{\mathrm{T}}$, and define $\vec{\boldsymbol{X}}$ as the augmented column vector of factors, i.e., $\vec{\boldsymbol{X}} = [\boldsymbol{X}^{\mathrm{T}}, 1]^{\mathrm{T}}$. Recall that we use y_j to denote the class belonging of observation $\boldsymbol{D}_j$, and we shall use $\#y$ to denote the number of observations in $\mathcal{D}$ for which $Y = y$.

The updating rules (M-step) for the EM-algorithm are given as follows (the derivations can be found in Appendix C):

$$\hat{P}(Y=y) \leftarrow \frac{\#j : y_j = y}{N}$$

$$\hat{P}(M=m|Y=k) \leftarrow \frac{\hat{P}(M=m|Y=k)\sum_{j:y^j=k} P(\boldsymbol{t}^j|M=m,Y=k)}{\#\{j : y^j = k\}}$$

$$\hat{\boldsymbol{\mu}}_y \leftarrow \frac{1}{\#y}\sum_{j=1:y_j=y}^{N}\sum_{m} P(M=m\,|\,\boldsymbol{D}_j)\mathbb{E}(\boldsymbol{X}\,|\,M=m,\boldsymbol{D}_j)$$

$$\hat{\boldsymbol{\Gamma}}_y \leftarrow \operatorname{diag}\left(\frac{1}{\#y}\sum_{j=1:y:j=y}^{N}\sum_{m} P(M=m\,|\,\boldsymbol{D}_j)\cdot \mathbb{E}((\boldsymbol{X}-\boldsymbol{\mu}_y)(\boldsymbol{X}-\boldsymbol{\mu}_y)^{\mathrm{T}}\,|\,\boldsymbol{D}_j, M=m)\right)$$

$$\widehat{\vec{\boldsymbol{w}}}_{i,m} \leftarrow -\left[2\sum_{j=1}^{N} P(M=m\,|\,\boldsymbol{D}_j)\lambda(\xi_{ijm})\cdot\mathbb{E}(\vec{\boldsymbol{X}}\vec{\boldsymbol{X}}^{\mathrm{T}}|\boldsymbol{D}_j, M=m)\right]^{-1}\cdot \left[\sum_{j=1}^{N}\left(t_{ij}-\frac{1}{2}\right)P(M=m\,|\,D_j)\mathbb{E}(\vec{\boldsymbol{X}}|\boldsymbol{D}_j, M=m)\right]$$

The E-step basically amounts to calculating $\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)$ and $\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m)$ (see Appendix C). The expectation $\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)$ is given by Equation (7) (conditioned on $M = m$) and $\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m)$ is found using

$$\Sigma^p = \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m) - \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m),$$

where Σ^p is given by Equation (6). Finally, $\mathbb{E}(\vec{\boldsymbol{X}}|\boldsymbol{D}_j) = [\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j)^{\mathrm{T}}, 1]^{\mathrm{T}}$ and

$$\mathbb{E}(\vec{\boldsymbol{X}}\vec{\boldsymbol{X}}^{\mathrm{T}}|\boldsymbol{D}_j) = \begin{bmatrix} \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j) & \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j) \\ \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j)^{\mathrm{T}} & 1 \end{bmatrix}.$$

It should be noticed that the updating steps above depend on the variational parameters, which in turn depend on $\boldsymbol{\Gamma}^p$ and $\boldsymbol{\mu}^p$. Hence, each iteration of the EM algorithm also involves updating the values for $\boldsymbol{\xi}$.

We end this section by noting that as an alternative to the generative models described here, one could also look for *discriminative models* inside the class of bLCM models, i.e., learn the parameters that maximise the conditional log likelihood, $\mathbb{E}\log(\prod_{i=1}^{N} P(y_i \,|\, \boldsymbol{t}_i))$ or a variational variant thereof. Empirical evidence [23–26] support that discriminative models generally obtain better classification results than generative models. However, learning the parameters that maximize the descriminative likelihood is NP-hard even when all data is observed [24], and we therefore leave learning of discriminative models as a topic for future research.

6 Experimental results

6.1 Classification accuracy

In this section we investigate the classification accuracy of the proposed classifier. We start by considering classification of handwritten digits collected in the USPS database [14]. The results are based on the *re-sampled* database, prepared by Rasmussen and Williams [15]. This database consists of 4649 training examples and 4649 test-examples. The examples are distributed unevenly among the classes, as described in Table 1.

The twenty first images in the test-set are shown in Fig. 7. The images are of

Digit	Training-set		Test-set		Digit	Training-set		Test-set	
0	767	16.5%	786	16.9%	5	361	7.8%	355	7.6%
1	622	13.4%	647	13.9%	6	420	9.0%	414	8.9%
2	475	10.2%	454	9.8%	7	390	8.4%	402	8.6%
3	406	8.7%	418	9.0%	8	377	8.1%	331	7.1%
4	409	8.8%	443	9.5%	9	422	9.1%	399	8.6%

Table 1
USPS dataset

size 16×16 pixels; they were originally grey-scale, but have been binarized for our application. Most digits are easily recognizable by humans, although the 16th image is difficult (it is a 4). Note also the difference in writing style between the different images (there are, for instance, three different ways to write the number 5 among the 20 examples).

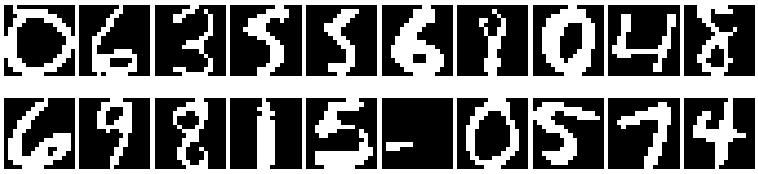

Fig. 7. The first 20 images of the test-set.

When learning the bLCM models we use Algorithm 2 with 10 wrapper folds. We report results for bLCMs *without* mixtures (denoted bLCM ($|\mathrm{sp}\,(M)| = 1$) in Table 2), as well as the general mbLCM model. [3] In order to learn the probability parameters in the models we applied the EM algorithm with standard parameter settings: The algorithm terminates when the relative increase in log (variational) likelihood falls below 10^{-3} or after a maximum of 50 iterations. The EM algorithm was run with 10 restarts; this gives a number of different candidate models from which we should select one. The standard solution is to choose the candidate model with the highest log-likelihood on the training data. However, since our focus is classification we instead pick the model that obtains the *highest classification accuracy* on the training-set. [4] The iterations of the variational approximation (Algorithm 1) were terminated when the relative increase in log (variational) likelihood of the data was less than 10^{-3}, or when a total of 10 iterations had been performed. [5]

[3] For the tests reported in this section, we have restricted q to take values form the set $\{2, 5, 10, 15, 20, 25, 30, 35, 40, 50, 75, 100\}$. We have considered a maximum of 2 mixtures.

[4] This is motivated by Vapnik's bound (see, e.g., [27, Section 2]), and the fact that all candidate models per definition have the same VC-dimension.

[5] Earlier work, including [17,19] conclude that the variational iterations converge very quickly, and that seldom more than three iterations are required to obtain

For comparison, a number of other classification algorithms have also been tested on the same dataset. The classification accuracies of the straw-men (further described in Appendix D) are given in the left column in Table 2. For each classifier, we give the results for three different classification problems for the USPS dataset: "$\{0, 1\}$ vs. $\{6, 7\}$" (as presented in Example 1), the ten-class problem of classifying all digits (denoted "$\{0\}$ – $\{9\}$"), and the "$\{3\}$ vs. $\{5\}$" dataset. The results show that, except for $\{0\}$ – $\{9\}$, the classification accuracy of the bLCMs is higher than the accuracy of the other classifiers in this domain. Of particular interest is the "$\{3\}$ vs. $\{5\}$" dataset, which was singled out as being particularly difficult by Rasmussen and Williams [15]. [6] The 20 images from the "$\{3\}$ vs. $\{5\}$" dataset that the bLCM classified wrongly are shown in Fig. 8. Although some of the images can be classified by humans (most notably images 1, 5, 6, and 7, 11 and 12), others are inherently difficult (i.e., images 3, 4, 9, 10, 15, 16 and 17).

1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20

Fig. 8. The 20 images that were misclassified by the bLCM classifier.

Next, we turn to classification of text documents, and the REUTERS-21578 dataset (Release 1.0) as used by Vomlel [28]. The split of data into training and test sets was made according to the time of publication of the documents (ModApte). Classes that contained only one document were eliminated together with the corresponding documents. The resulting datasets contain 7769 documents for training and 3018 documents for testing. The tests were performed on the three classes containing the most documents, and for each document-class we created a classification problem, where the task was to decide whether or not a test document belonged to that particular document-class (hence, we made three two-class classification problems). After removing function words and words that appear in only one document, a total of 15515 words remained. The words were coded as binary attributes, were each

a good approximation. In our high-dimentional data we have observed a different effect, and conclude that up to ten iterations are sometimes required for approximations that are accurate enough for our learning procedure.

[6] Rasmussen and Williams [15] reported a classification accuracy of 97.28% for their Gaussian process classifier (with expectation propagation) on the "$\{3\}$ vs. $\{5\}$" dataset using the original grey-scale images; using the the binarized data we obtained an accuracy of 95.99%. For additional comparison, we can also mention that the accuracy of the classifier was found to be 98.71% for the "$\{0, 1\}$ vs. $\{6, 7\}$" problem using binarized data. Note that Rasmussen and Williams' implementation only supports two-class problems, hence fails to handle the "$\{0\}$ – $\{9\}$" dataset. We were also not able to obtain results for the REUTERS datasets; tests were terminated after 48 hrs. CPU time on a MacBook Pro 2.6GHz Intel Core 2 Duo with 4GB RAM.

Classifier	{0&1} vs. {6&7}	{0} – {9}	{3} vs. {5}	`crude`	`earn`	`acq`
Majority Vote	63.72%	16.91%	54.08%	93.74%	63.98%	76.18%
Winnow	61.32%	60.77%	83.83%	97.42%	97.22%	90.95%
ANN	96.27%	94.04%	95.34%	97.35%	98.51%	97.38%
ADTree	95.02%	83.63%	92.24%	97.45%	96.36%	89.89%
1-NN	98.31%	92.88%	95.73%	95.99%	95.69%	93.24%
SVM	96.27%	84.82%	95.34%	97.78%	98.51%	97.18%
Naïve Bayes	89.42%	85.33%	94.44%	95.16%	94.00%	96.95%
TAN	96.04%	88.41%	94.44%	96.36%	93.47%	96.72%
ID3	95.33%	80.68%	89.91%	96.72%	96.42%	94.10%
Logistic Regression	92.62%	79.74%	92.63%	95.43%	96.39%	94.30%
Radial Basis Functions	92.89%	88.15%	94.67%	96.55%	94.47%	96.82%
AODE	97.69%	91.37%	95.86%	97.71%	97.65%	97.12%
HNB	96.13%	87.80%	95.47%	96.58%	92.88%	96.79%
BayesNet	89.42%	85.37%	94.44%	95.20%	94.07%	96.92%
bLCM ($\lvert \mathrm{sp}\,(M) \rvert = 1$)	98.84%	89.09%	97.41%	97.91%	97.55%	97.28%
mbLCM	99.20%	93.44%	97.41%	97.02%	98.38%	97.32%

Table 2
Classification results for different digit collections from the USPS database as well as the REUTERS datasets. Both probabilistic as well as non-probabilistic classifiers are included (separated by the horizontal line).

attribute told of the existence (or non-existence) of a specific word in a document. For each classification task, we then selected the 500 most informative features using the expected information gain as feature selection criteria. The results can be seen in the right column of Table 2. In particular, we see that bLCMs and ANNs perform at comparative levels, and that both perform better than the other classifiers within this domain. To put the accuracy differences into perspective, we see that when the ANN obtain better results than the bLCM the difference reduces to a misclassification of at most four instance (out of a total of 3018 test instances). As noted by, e.g., Kohavi [29], the generated accuracies are in fact estimators with their own underlying statistical distribution. The standard deviations of the estimators reported in Table 2 are of the order of approximately .5%. For instance, the 95% confidence interval for the mbLCM accuracy on the `earn` dataset is $[97.86\%, 98.77\%]$.

6.2 Generative models

The previous subsection showed that bLCMs offer classification accuracies at a comparative or higher level than other classifiers. In this subsection we investigate another aspect of the bLCM classifier, namely how the underlying generative model can be used for classification and extrapolation of partially

disclosed images. That is, from a partial observation, the system will generate not only a classification, but also impute the missing part of the image based on what is already known. We believe this to be an interesting ability for a classifier; one potential application being a PDA that can recognize and auto-complete symbols *while* they are being written. Note that this is not possible using e.g. ANNs as they do not specify a generative model.

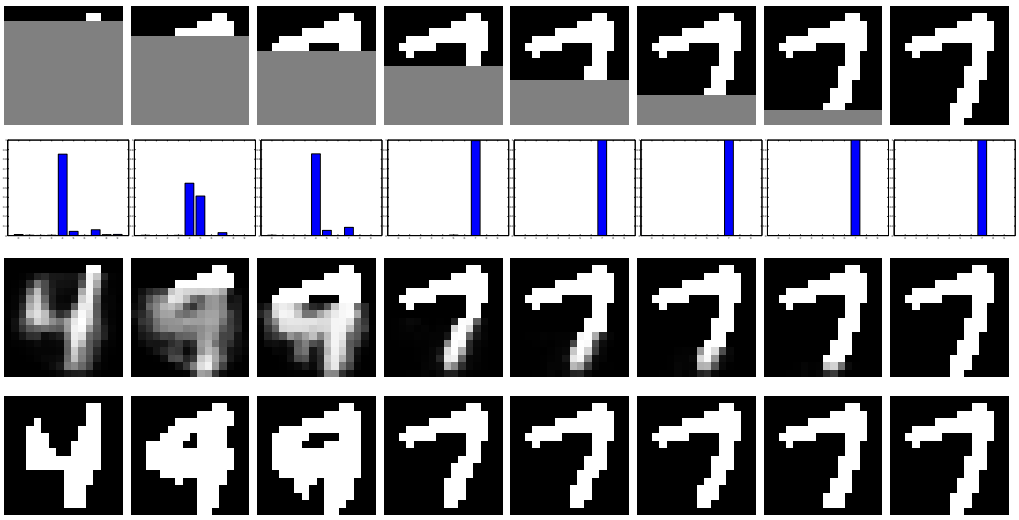

Fig. 9. Using a bLCM for classifying and extrapolating an image of the digit 7 as it is being disclosed.

An example of this process is given in Fig. 9. The first row of images shows how the information is partially disclosed, and the second row shows how the probability distribution $P(y \mid \text{Partial image})$ is changing as more information is given. The third row shows the expected completion of the image given the partial observation, and finally the last row shows the most probable (pixel-wise) completion of the image assuming that the most probable class is indeed the correct one.[7] Notice how the system, after having seen the two first lines of pixels (first column of images in Fig. 9), is fairly convinced that the image is of a 4; only 5 and 7 are considered as possible alternative hypothesis. The reason for this is that the partial observation is consistent with an already observed writing-style for the number 4. The second column shows the status when two more lines of pixels are observed. The system still believes it is a 4, but now has a different belief regarding the shape of the digit. Note how the probability for the digit being a 5 has increased considerably. For the third column, the white part on the left-hand side of the image is not consistent with the image being a 5, so that is not deemed as probable as before. The completion of the image is however difficult to interpret. Next, half of the image is disclosed in the fourth column, and the system recognizes a 7 with

[7] This example is used only to illustrate how the generative properties of the bLCM may be exploited. Ideally, one should consider the most probable configuration over all the unobserved attributes.

high confidence. When the remaining pixels in the image are observed, this hypothesis is confirmed. For comparison, Fig. 10 shows the same process for the naïve Bayes classifier. The results of the naïve Bayes are not impressive, as the extrapolated images (bottom row) do not look like real digits. It is also worth noticing that the naïve Bayes ends up believing that the fully disclosed image is a 4 instead of a 7.

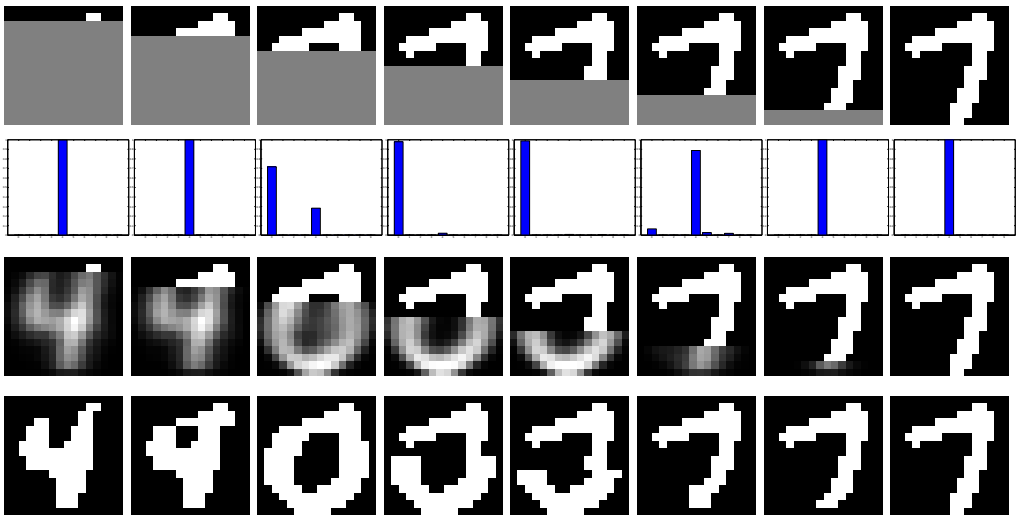

Fig. 10. Using a naïve Bayes for classifying and extrapolating an image of the digit 7 as it is being disclosed.

7 Discussion and future work

In this paper we have further developed the class of latent classification models (LCMs) [6]. Whereas the original model class were used for probabilistic classification in *continuous domains*, the present extension focuses on *binary* domains, e.g. black and white pictures. A binary LCM (bLCM) can, as the LCM model, roughly be seen as a mixture of factor analyzers integrated with a naïve Bayes model. This combination enables concurrent clustering and, within each cluster, localized classification. bLCMs relax the conditional independence assumptions embedded in the naïve Bayes models, thereby allowing any probability distribution over binary attributes to be approximated arbitrarily well.

In our experiments, we have demonstrated that bLCMs provide good classification results in binary domains, and we found that bLCMs appear to be better than a wide range of other probabilistic classifiers. Finally, we also showed how the generative properties of the classifier can be exploited. In particular, we considered the classification and extrapolation of partial images, with potential application for e.g. real-time optical character recognition.

As part of future work, we plan to extend the bLCM/LCM model class to general *hybrid* domains; a process that has already started. First, we note that merging bLCMs for binary domains with our previous work in continuous domains [6] is straight-forward using a two-pass scheme: Using evidence from the continuous attributes only, we calculate the posterior distribution over the latent variables given this partial observation. Next, we treat the posterior distribution as a *prior* distribution, when the binary variables are considered, and classification can then proceed as previously described. The main challenge is therefore to extend the bLCM framework to *discrete* variables. Naïvely, one could redefine a dataset containing discrete variables by translating each discrete variable into a set of binary variables: Consider the discrete variable D with r states. Then, D can be represented using $\lceil \log_2(r) \rceil$ binary variables B_i [17]. Note that the new variables B_i are conditionally dependent, and that latent variables must therefore be introduced to model this dependency. From Proposition 1 we know that this can be handled within the bLCM framework, but unfortunately the number of mixture components required is exponential in the number of attributes. In total the number of mixture components required to model D is linear in the number of states in D. This complexity is prohibitive, and we should rather try to find a more direct representation. To support the integration with LCMs, we want to maintain the structure of the bLCMs, and only modify the distributional assumption for the attributes. The natural choice is to let the conditional distribution of a discrete variable D with continuous parents $\boldsymbol{X}$ be defined by the soft-max function. In this formulation, we have one set of parameters ($\boldsymbol{w}$ and b) per state d of D, and use

$$P(D = d \,|\, \boldsymbol{x}) = \frac{\exp(-(\boldsymbol{w}_d^{\mathrm{T}} \boldsymbol{x} + b_d))}{\sum_{d'} \exp(-(\boldsymbol{w}_{d'}^{\mathrm{T}} \boldsymbol{x} + b_{d'}))}.$$

The lowerbound Equation (8) does not extend to soft-max functions [20,30], but recent research (see, e.g., [31]) has brought some possible solutions that we want to pursue in the future.

A The variational posterior distribution for the latent variables

In this section we derive the posterior variational distribution of the latent variables $\boldsymbol{X}$ given a configuration $\boldsymbol{T} = \boldsymbol{t}$ and $Y = y$. For ease of notation we shall restrict our attention to bLCMs having no mixture variables, however, the generalization to mixture bLCMs is straightforward.

First of all, recall that

$$\begin{aligned}\tilde{f}(\boldsymbol{t},\boldsymbol{x}\,|\,y) &= \tilde{P}(\boldsymbol{t}\,|\,\boldsymbol{x},\boldsymbol{\xi})f(\boldsymbol{x}\,|\,y)\\ &= (2\pi)^{-q/2}|\boldsymbol{\Gamma}_y|^{-1/2}\exp\left(-\frac{1}{2}(\boldsymbol{x}-\boldsymbol{\mu}_y)^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_y)\right)\\ &\quad\prod_{j=1}^{d} g(\xi_j)\exp((A_j-\xi_j)/2+\lambda(\xi_j)(A_j^2-\xi_j^2)),\end{aligned}$$

where $A_i = (2t_i-1)(\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{x}+b_i)$ and $A_i^2 = (\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{x})^2 + b_i^2 + 2\boldsymbol{w}_i^{\mathrm{T}}\boldsymbol{x}b_i$; for the latter we have used that $(2t_i-1)^2 = 1$.

By exploiting that

$$\begin{aligned}\tilde{P}(t_j\,|\,\boldsymbol{x},\boldsymbol{w}_j,\xi_j) = \exp\Bigg(&\log(g(\xi_j)) + \frac{1}{2}(2t_j-1)\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x} + \frac{1}{2}(2t_j-1)b_j - \frac{1}{2}\xi_j\\ &+\lambda(\xi_j)(\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x})^2 + \lambda(\xi_j)b_j^2 + \lambda(\xi_j)2\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}b_j - \lambda(\xi_j)\xi_j^2\Bigg)\end{aligned}$$

we can write $\tilde{P}(t_j\,|\,\boldsymbol{x},\boldsymbol{w}_j,\xi_j)$ on canonical form [32]. That is, $\tilde{P}(t_j\,|\,\boldsymbol{x},\boldsymbol{w}_j,\xi_j)$ can be written as $\exp(a_j^+ + \boldsymbol{b}_j^+\boldsymbol{x} - \boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}_j^+\boldsymbol{x})$, where a_j^+ is a constant, $\boldsymbol{b}_j^+$ is a vector, and $\boldsymbol{C}_j^+$ is a full-rank square matrix. By letting a_j, b_j, and c_j denote the contributions from a_j^+, $\boldsymbol{b}_j^+\boldsymbol{x}$, and $\boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}_j^+\boldsymbol{x}$, respectively, we get: [8]

$$\begin{aligned}a_j =& \log(g(\xi_j)) + \frac{1}{2}(2t_j-1)b_j - \frac{1}{2}\xi_j - \lambda(\xi_j)\xi_j^2 + \lambda(\xi_j)b_j^2;\\ b_j =& \frac{1}{2}(2t_j-1)\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x} + \lambda(\xi_j)(2\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}b_j) = \left(\frac{1}{2}(2t_j-1)+\lambda(\xi_j)2b_j\right)\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x};\\ c_j =& -\lambda(\xi_j)(\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x})^2 = -\lambda(\xi_j)(\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}) = -\boldsymbol{x}^{\mathrm{T}}\lambda(\xi_j)\boldsymbol{w}_j\boldsymbol{w}_j^{\mathrm{T}}\boldsymbol{x}.\end{aligned}$$

Similarly, $f(\boldsymbol{x}\,|\,y)$ can be written on canonical form with:

$$\begin{aligned}a' =& -\frac{q}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Gamma}_y|) - \frac{1}{2}\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y;\\ b' =& \boldsymbol{x}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y = \boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{x};\\ c' =& \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{x}.\end{aligned}$$

Now, for the products $\tilde{P}(t_j\,|\,\boldsymbol{x},\boldsymbol{w}_j,\xi_j)f(\boldsymbol{x}\,|\,y)$ and $\prod_{j=1}^{d}\tilde{P}(t_j\,|\,\boldsymbol{x},\boldsymbol{w}_j,\xi_j)f(\boldsymbol{x}\,|\,y)$

[8] For this we exploit $(\boldsymbol{x}-\boldsymbol{\mu}_y)^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}(\boldsymbol{x}-\boldsymbol{\mu}_y) = \boldsymbol{x}^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{x} + \boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y - 2\boldsymbol{x}^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y$.

we get:

$$
\begin{aligned}
a'_j =& -\frac{q}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Gamma}_y|) - \frac{1}{2}\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y + \log(g(\xi_j)) + \frac{1}{2}(2t_j - 1)b_j - \frac{1}{2}\xi_j \\
& - \lambda(\xi_j)\xi_j^2 + \lambda(\xi_j)b_j^2; \\
b'_j =& \left(\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1} + \left(\frac{1}{2}(2t_j - 1) + \lambda(\xi_j)2b_j\right)\boldsymbol{w}_j^{\mathrm{T}}\right)\boldsymbol{x}; \\
c'_j =& \boldsymbol{x}^{\mathrm{T}}\left(\frac{1}{2}\boldsymbol{\Gamma}_y^{-1} - \lambda(\xi_j)\boldsymbol{w}_j\boldsymbol{w}_j^{\mathrm{T}}\right)\boldsymbol{x},
\end{aligned}
$$

and

$$
\begin{aligned}
a^* =& \sum_{j=1}^{d} g'_j; \\
b^* =& \left(\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1} + \sum_{j=1}^{d}\left(\frac{1}{2}(2t_j - 1) + \lambda(\xi_j)2b_j\right)\boldsymbol{w}_j^{\mathrm{T}}\right)\boldsymbol{x}; \\
c^* =& \boldsymbol{x}^{\mathrm{T}}\left(\frac{1}{2}\boldsymbol{\Gamma}_y^{-1} - \sum_{j=1}^{d}\lambda(\xi_j)\boldsymbol{w}_j\boldsymbol{w}_j^{\mathrm{T}}\right)\boldsymbol{x},
\end{aligned}
$$

respectively. From a^*, b^*, and c^* we have that the posterior for $\boldsymbol{X}$ given $\boldsymbol{t}$ and y is a Gaussian distribution and by transforming back to moment form we get

$$
\begin{aligned}
\boldsymbol{\Gamma}_y^p =& \left[\boldsymbol{\Gamma}_y^{-1} - 2\sum_{j=1}^{d}\lambda(\xi_j)\boldsymbol{w}_j\boldsymbol{w}_j^{\mathrm{T}}\right]^{-1}; \\
\boldsymbol{\mu}_y^p =& \boldsymbol{\Gamma}_y^p\left[\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1} + \sum_{j=1}^{d}\left(t_j - \frac{1}{2} + 2\lambda(\xi_j)b_j\right)\boldsymbol{w}_j^{\mathrm{T}}\right].
\end{aligned}
$$

B A lower bound on $f(\boldsymbol{t})$

Since $\tilde{P}(\boldsymbol{t}\,|\,\boldsymbol{x},\boldsymbol{\xi}) \leq P(\boldsymbol{t}\,|\,\boldsymbol{x})$ for all $\boldsymbol{\xi}$ we have that $\tilde{f}(\boldsymbol{x},\boldsymbol{t}) \leq f(\boldsymbol{x},\boldsymbol{t})$ and therefore $\tilde{f}(\boldsymbol{t}) \leq f(\boldsymbol{t})$. In order to evaluate the integral $\tilde{f}(\boldsymbol{t}) = \int_{\boldsymbol{x}\in\mathbb{R}^q}\tilde{f}(\boldsymbol{x},\boldsymbol{t})d\boldsymbol{x}$ we

rewrite $\tilde{f}(\boldsymbol{x}, \boldsymbol{t})$ on canonical form (see Appendix A):

$$\begin{aligned}
\tilde{f}(\boldsymbol{t}) &= \int_{\boldsymbol{x}\in\mathbb{R}^q} \tilde{f}(\boldsymbol{x}, \boldsymbol{t}) d\boldsymbol{x} \\
&= \int_{\boldsymbol{x}\in\mathbb{R}^q} \exp\left(a + \boldsymbol{b}\boldsymbol{x} - \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{x}\right) d\boldsymbol{x} \\
&= \exp(a) \int_{\boldsymbol{x}\in\mathbb{R}^q} \exp\left(\boldsymbol{b}\boldsymbol{x} - \frac{1}{2}\boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{x}\right) d\boldsymbol{x} \\
&= \exp(a) \exp\left(\frac{1}{2}\boldsymbol{b}\boldsymbol{C}^{-1}\boldsymbol{b}^{\mathrm{T}}\right) \int_{\boldsymbol{x}\in\mathbb{R}^q} \exp\left(-\frac{1}{2}\left(\boldsymbol{x}^{\mathrm{T}}\boldsymbol{C}\boldsymbol{x} - 2\boldsymbol{b}\boldsymbol{x} + \boldsymbol{b}\boldsymbol{C}^{-1}\boldsymbol{b}^{\mathrm{T}}\right)\right) d\boldsymbol{x} \\
&= \exp(a) \exp\left(\frac{1}{2}\boldsymbol{b}\boldsymbol{C}^{-1}\boldsymbol{b}^{\mathrm{T}}\right) \int_{\boldsymbol{x}\in\mathbb{R}^q} \exp\left((\boldsymbol{x} - \boldsymbol{C}^{-1}\boldsymbol{b})^{\mathrm{T}}\boldsymbol{C}(\boldsymbol{x} - \boldsymbol{C}^{-1}\boldsymbol{b})\right) d\boldsymbol{x}.
\end{aligned}$$

Since

$$\int_{\boldsymbol{x}\in\mathbb{R}^q} \exp\left((\boldsymbol{x} - \boldsymbol{C}^{-1}\boldsymbol{b})^{\mathrm{T}}\boldsymbol{C}(\boldsymbol{x} - \boldsymbol{C}^{-1}\boldsymbol{b})\right) d\boldsymbol{x} = (2\pi)^{q/2}|\boldsymbol{C}^{-1}|^{1/2}$$

we get

$$\tilde{f}(\boldsymbol{t}) = \exp(a) \exp\left(\frac{1}{2}\boldsymbol{b}\boldsymbol{C}^{-1}\boldsymbol{b}^{\mathrm{T}}\right)(2\pi)^{q/2}|\boldsymbol{C}^{-1}|^{1/2}.$$

From Appendix A we have that

$$\begin{aligned}
a =& -\frac{q}{2}\log(2\pi) - \frac{1}{2}\log(|\boldsymbol{\Gamma}_y|) - \frac{1}{2}\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y + \sum_{j=1}^{d}\Bigg(\log(g(\xi_j)) + \frac{1}{2}(2t_j - 1)b_j \\
& - \frac{1}{2}\xi_j - \lambda(\xi_j)(\xi_j^2 - b_j^2)\Bigg); \\
\boldsymbol{b} =& \boldsymbol{\mu}_y^p \boldsymbol{\Gamma}^{p-1}; \\
\boldsymbol{C}^{-1} =& \boldsymbol{\Gamma}_y^p.
\end{aligned}$$

hence,

$$\begin{aligned}
\tilde{f}(\boldsymbol{t}) =& \exp\left\{-\frac{1}{2}\boldsymbol{\mu}^{\mathrm{T}}\boldsymbol{\Gamma}^{-1}\boldsymbol{\mu} + \frac{1}{2}(\boldsymbol{\mu}^p)^{\mathrm{T}}(\boldsymbol{\Gamma}^p)^{-1}\boldsymbol{\mu}^p + \frac{1}{2}\log\left(\frac{|\boldsymbol{\Gamma}^p|}{|\boldsymbol{\Gamma}|}\right)\right\} \cdot \\
& \exp\left\{\sum_{i=1}^{d}\left\{\log(g(\xi_i)) - \xi_i/2 + \lambda_i(b_i^2 - \xi_i^2) + \frac{1}{2}(2t_i - 1)b_i\right\}\right\}.
\end{aligned}$$

C The EM algorithm for bLCMs

In this section we derive an EM algorithm for bLCMs. Unfortunately, taking direct outset in the bLCM specification is not possible, since the E-step of the algorithm requires inference in the underlying model. In particular, we should be able to calculate the marginal likelihood of the data:

$$f(\boldsymbol{t}, y) = P(y) \int_{\mathbb{R}^q} \left\{ \prod_{i=1}^{d} P(t_i \,|\, \boldsymbol{x}) \right\} f(\boldsymbol{x}|y) d\boldsymbol{x},$$

but this integral cannot be evaluated analytically. Instead we use a variational approximation $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i)$ (see Equation 5) to the logistic function, which ensures that $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i) \leq P(t_i \,|\, \boldsymbol{x})$ for all ξ_i and $\tilde{P}(t_i \,|\, \boldsymbol{x}, \xi_i) = P(t_i \,|\, \boldsymbol{x})$ for some particular choice of ξ_i. By using the variational approximation we get a lower bound $\tilde{f}$ on the marginal likelihood, and rather than maximizing the marginal likelihood directly, we instead maximize the variational lower bound. This operation is guaranteed to never decrease the marginal likelihood.

In order to derive the updating rules we first note that

$$\begin{aligned}
\tilde{f}(\boldsymbol{t}, \boldsymbol{x}, m, y) &= \tilde{P}(\boldsymbol{t} \,|\, \boldsymbol{x}, m) f(\boldsymbol{x} \,|\, y) P(m \,|\, y) P(y) \\
&= P(y) P(m \,|\, y) (2\pi)^{-q/2} |\boldsymbol{\Gamma}_y|^{-1/2} \exp\left(-\frac{1}{2} (\boldsymbol{x} - \boldsymbol{\mu}_y)^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_y) \right) \\
&\quad \prod_{j=1}^{d} g(\xi_{j,m}) \exp((A_{j,m} - \xi_{j,m})/2 + \lambda(\xi_{j,m})(A_{j,m}^2 - \xi_{j,m}^2))
\end{aligned}$$

By exploiting that

$$(\boldsymbol{x} - \boldsymbol{\mu}_y)^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} (\boldsymbol{x} - \boldsymbol{\mu}_y) = tr(\boldsymbol{\Gamma}_y^{-1} \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}) - 2\boldsymbol{\mu}_y^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} \boldsymbol{x} + \boldsymbol{\mu}_y^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} \boldsymbol{\mu}_y$$

and taking the logarithm we get

$$\begin{aligned}
\log \tilde{f}(\boldsymbol{t}, \boldsymbol{x}, m, y) &= \log P(y) + \log P(m \,|\, y) - \frac{q}{2} \log 2\pi \\
&\quad - \frac{1}{2} |\boldsymbol{\Gamma}_y| - \frac{1}{2} tr(\boldsymbol{\Gamma}_y^{-1} \boldsymbol{x}\boldsymbol{x}^{\mathrm{T}}) + \boldsymbol{\mu}_y^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} \boldsymbol{x} - \frac{1}{2} \boldsymbol{\mu}_y^{\mathrm{T}} \boldsymbol{\Gamma}_y^{-1} \boldsymbol{\mu}_y \\
&\quad - \sum_{j=1}^{d} \log(1 + \exp(-\xi_{j,m}) + \sum_{j=1}^{d} \frac{A_{j,m} - \xi_{j,m}}{2} + \lambda(\xi_{j,m})(A_{j,m}^2 - \xi_{j,m}^2).
\end{aligned}$$

The expected data-complete variational log-likelihood is now given by

$$\begin{aligned}
\mathcal{Q} =& \mathbb{E}\log(\prod_{i=1}^{N}\tilde{f}(\cdot \mid \boldsymbol{D}_i)) = \sum_{i=1}^{N}\mathbb{E}\log(\tilde{f}(\cdot \mid \boldsymbol{D}_i)) \\
=& \sum_{i=1}^{N}\log P(y_i) + \sum_{i=1}^{N}\mathbb{E}(\log P(M \mid y_i) \mid \boldsymbol{D}_i) - \frac{Nq}{2}\log 2\pi - \sum_{h=1}^{|\mathrm{sp}(Y)|}\frac{\#y_h}{2}\log|\boldsymbol{\Gamma}_{y_h}| - \\
& \frac{1}{2}\sum_{i=1}^{N} tr(\boldsymbol{\Gamma}_{y_i}^{-1}\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \mid \boldsymbol{D}_i)) + \sum_{i=1}^{N}\boldsymbol{\mu}_{y_i}^{\mathrm{T}}\boldsymbol{\Gamma}_{y_1}^{-1}\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i) - \sum_{h=1}^{|\mathrm{sp}(Y)|}\frac{\#y_h}{2}\boldsymbol{\mu}_{y_h}^{\mathrm{T}}\boldsymbol{\Gamma}_{y_h}^{-1}\boldsymbol{\mu}_{y_h} - \\
& \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\log(1+\exp(-\xi_{i,j,M})) \mid \boldsymbol{D}_i) + \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}\left(\frac{A_{i,j,M}-\xi_{i,j,M}}{2}\middle| \boldsymbol{D}_i\right) + \\
& \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\lambda(\xi_{i,j,M})(A_{i,j,M}^2 - \xi_{i,j,M}^2) \mid \boldsymbol{D}_i)
\end{aligned} \tag{C.1}$$

The last two terms can be rewritten by first noticing that $\mathbb{E}((A_{i,j,M}-\xi_{i,j,M})/2) = \mathbb{E}(A_{i,j,M}/2 \mid \boldsymbol{D}_i) - \mathbb{E}(\xi_{i,j,M}/2 \mid \boldsymbol{D}_i)$ and $\mathbb{E}(A_{i,j,M}/2 \mid \boldsymbol{D}_i) = \mathbb{E}((2t_{i,j}-1)(\boldsymbol{w}_{j,M}^{\mathrm{T}}\boldsymbol{X} + b_{j,M}) \mid \boldsymbol{D}_i)$. By defining $\vec{\boldsymbol{w}} = [\boldsymbol{w}_{j,M}^{\mathrm{T}}, b_{j,M}]^{\mathrm{T}}$ and $\vec{\boldsymbol{X}} = [\boldsymbol{X}^{\mathrm{T}}, 1]^{\mathrm{T}}$ we get

$$\mathbb{E}\left(\frac{A_{i,j,M}}{2}\middle| \boldsymbol{D}_i\right) = \frac{1}{2}(2t_{i,j}-1)\mathbb{E}(\vec{\boldsymbol{w}}_{j,M}^{\mathrm{T}}\vec{\boldsymbol{X}} \mid \boldsymbol{D}_i).$$

By exploiting that $(2t_{i,j}-1)^2 = 1$ for $t_{i,j} \in \{0,1\}$ we can rewrite Equation C.1 as

$$\begin{aligned}
\mathcal{Q} =& \mathbb{E}\log(\prod_{i=1}^{N}\tilde{f}(\cdot \mid \boldsymbol{D}_i)) = \sum_{i=1}^{N}\mathbb{E}\log(\tilde{f}(\cdot \mid \boldsymbol{D}_i)) \\
=& \sum_{i=1}^{N}\log P(y_i) + \sum_{i=1}^{N}\mathbb{E}\log P(M \mid y_i) - \frac{Nq}{2}\log 2\pi - \sum_{h=1}^{|\mathrm{sp}(Y)|}\frac{\#y_h}{2}\log|\boldsymbol{\Gamma}_{y_h}| - \\
& \frac{1}{2}\sum_{i=1}^{N} tr(\boldsymbol{\Gamma}_{y_i}^{-1}\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \mid \boldsymbol{D}_i) + \sum_{i=1}^{N}\boldsymbol{\mu}_{y_i}^{\mathrm{T}}\boldsymbol{\Gamma}_{y_1}^{-1}\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i) - \sum_{h=1}^{|\mathrm{sp}(Y)|}\frac{\#y_h}{2}\boldsymbol{\mu}_{y_h}^{\mathrm{T}}\boldsymbol{\Gamma}_{y_h}^{-1}\boldsymbol{\mu}_{y_h} - \\
& \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\log(1+\exp(-\xi_{i,j,M})) \mid \boldsymbol{D}_i) + \frac{1}{2}\sum_{j=1}^{d}\sum_{i=1}^{N}(2t_{i,j}-1)\mathbb{E}(\vec{\boldsymbol{w}}_{j,M}^{\mathrm{T}}\vec{\boldsymbol{X}} \mid \boldsymbol{D}_i) - \\
& \frac{1}{2}\sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\xi_{i,j,M} \mid \boldsymbol{D}_i) + \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\lambda(\xi_{i,j,M})\vec{\boldsymbol{w}}_{j,M}^{\mathrm{T}}\vec{\boldsymbol{X}}\vec{\boldsymbol{w}}_{j,M}^{\mathrm{T}}\vec{\boldsymbol{X}} \mid \boldsymbol{D}_i) - \\
& \sum_{j=1}^{d}\sum_{i=1}^{N}\mathbb{E}(\lambda(\xi_{i,j,M})\xi_{i,j,M} \mid \boldsymbol{D}_i).
\end{aligned}$$

Based on the above expression we can now derive the updating rules (the M-step) for the EM algorithm.

$$\frac{\partial \mathcal{Q}}{\partial \vec{\boldsymbol{w}}_{j,m}} = \sum_{i=1}^{N}(t_{i,j} - \frac{1}{2})P(M = m \mid \boldsymbol{D}_i)\mathbb{E}(\vec{\boldsymbol{X}} \mid \boldsymbol{D}_i, M = m) + 2\sum_{i=1}^{N} P(M = m \mid \boldsymbol{D}_i)\lambda(\xi_{i,j,m})\mathbb{E}(\vec{\boldsymbol{X}}\vec{\boldsymbol{X}}^{\mathrm{T}} \mid \boldsymbol{D}_i, M = m)\vec{\boldsymbol{w}}_{j,m}$$

By setting the derivative equal to 0 we get the following updating rule for $\vec{\boldsymbol{w}}_{j,m}$:

$$\hat{\vec{\boldsymbol{w}}}_{j,m} \leftarrow -\left[2\sum_{i=1}^{N} P(M = m \mid \boldsymbol{D}_i)\lambda(\xi_{i,j,m})\mathbb{E}(\vec{\boldsymbol{X}}\vec{\boldsymbol{X}}^{\mathrm{T}} \mid \boldsymbol{D}_i, M = m)\right]^{-1} \left[\sum_{i=1}^{N}(t_{i,j} - \frac{1}{2})P(M = m \mid \boldsymbol{D}_i)\mathbb{E}(\vec{\boldsymbol{X}} \mid \boldsymbol{D}_i, M = m)\right]$$

For $\boldsymbol{\mu}_y$ we have

$$\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\mu}_y} = \sum_{i=1:y_i=y}^{N} \boldsymbol{\Gamma}_y^{-1}\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i) - \#y\boldsymbol{\Gamma}_y^{-1}\boldsymbol{\mu}_y,$$

which results in the following updating rule

$$\hat{\boldsymbol{\mu}_y} \leftarrow \frac{1}{\#y}\boldsymbol{\Gamma}_y\boldsymbol{\Gamma}_y^{-1}\sum_{i=1:y_i=y}^{N}\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i) = \frac{1}{\#y}\sum_{i=1:y_i=y}^{N}\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i).$$

Finally, for $\boldsymbol{\Gamma}_y$ the partial derivative is

$$\begin{aligned}\frac{\partial \mathcal{Q}}{\partial \boldsymbol{\Gamma}_y} =& -\frac{\#y}{2}\boldsymbol{\Gamma}_g^{-1^{\mathrm{T}}} + \frac{1}{2}\sum_{i=1:y_i=y}^{N}\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}}\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \mid \boldsymbol{D}_i)\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}} - \\ & \sum_{i=1:y_i=y}^{N}\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}}\boldsymbol{\mu}_y\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i)^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}} + \frac{\#y}{2}\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}}\boldsymbol{\mu}_y\boldsymbol{\mu}_y^{\mathrm{T}}\boldsymbol{\Gamma}_y^{-1^{\mathrm{T}}} \\ =& \boldsymbol{\Gamma}_y^{-1}\left(-\frac{\#y}{2} + \sum_{i=1:y_i=y}^{N}\left(-\frac{1}{2}\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \mid \boldsymbol{D}_i) - \boldsymbol{\mu}_y\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i)^{\mathrm{T}} + \frac{1}{2}\boldsymbol{\mu}_y\boldsymbol{\mu}_y^{\mathrm{T}}\right)\boldsymbol{\Gamma}_y^{-1}\right),\end{aligned}$$

which gives

$$\begin{aligned}\hat{\boldsymbol{\Gamma}_y} \leftarrow& \frac{1}{\#y}\sum_{i=1:y_i=y}^{N}(\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}} \mid \boldsymbol{D}_i) - 2\boldsymbol{\mu}_y\mathbb{E}(\boldsymbol{X} \mid \boldsymbol{D}_i)^{\mathrm{T}} + \boldsymbol{\mu}_y\boldsymbol{\mu}_y^{\mathrm{T}}) = \\ =& \frac{1}{\#y}\sum_{i=1:y_i=y}^{N}\sum_{m}P(M = m \mid \boldsymbol{D}_i)[\mathbb{E}((\boldsymbol{X} - \boldsymbol{\mu}_y)(\boldsymbol{X} - \boldsymbol{\mu}_y)^{\mathrm{T}} \mid \boldsymbol{D}_i, M = m)]\end{aligned}$$

To estimate the probability $P(y)$ we perform simple frequency counting in the database, and for $P(M = m \,|\, Y = y)$ we use

$$\begin{aligned}\hat{P}(M = m \,|\, Y = y) \leftarrow & \frac{\sum_{i=1:y_i=y}^{N} P(M = m, Y = y \,|\, \boldsymbol{D}_i)}{\#y} \\ = & \frac{P(M = m \,|\, Y = y) \sum_{i=1:y_i=y}^{N} \frac{P(\boldsymbol{t}_i \,|\, M=m, Y=y)}{P(\boldsymbol{t}_i \,|\, y)}}{\#y}\end{aligned}$$

In additions to the expectations (derived below), the updating rules above also require $P(M = m \,|\, \boldsymbol{D}_i)$. This probability can be found by straight-forward application of Bayes' rule:

$$P(M = m \,|\, \boldsymbol{D}_i) = \frac{P(\boldsymbol{D}_i \,|\, M = m)P(M = m)}{\sum_m P(\boldsymbol{D}_i \,|\, M = m)P(M = m)};$$

$P(\boldsymbol{D}_i \,|\, M = m)$ can be found from Equation 8 and $P(M = m) = \sum_y P(M = m \,|\, Y = y)P(Y = y)$

The E-step amounts to calculating $\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)$ and $\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m)$, since $\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j) = \sum_m P(M = m \,|\, \boldsymbol{D}_i)\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)$ and $\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j) = \sum_m P(M = m \,|\, \boldsymbol{D}_i)\ \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m)$. The expectation $\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)$ is given by Equation 7 (conditioned on $M = m$) and $\mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m)$ is found by exploiting that

$$\Sigma^p = \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j, M = m) - \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m)\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j, M = m),$$

where Σ^p is given by Equation 6. In addition, $\mathbb{E}(\vec{\boldsymbol{X}}|\boldsymbol{D}_j) = [\mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j)^{\mathrm{T}}, 1]^{\mathrm{T}}$ and

$$\mathbb{E}(\vec{\boldsymbol{X}}\vec{\boldsymbol{X}}^{\mathrm{T}}|\boldsymbol{D}_j) = \begin{bmatrix} \mathbb{E}(\boldsymbol{X}\boldsymbol{X}^{\mathrm{T}}|\boldsymbol{D}_j) & \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j) \\ \mathbb{E}(\boldsymbol{X}|\boldsymbol{D}_j)^{\mathrm{T}} & 1 \end{bmatrix}.$$

D Straw-men

In this last section we briefly describe the learning algorithms used as straw-men in Table 2. The straw-men are all implemented in the Weka system version 3.5 [33], and all models were learned using default parameter settings.

Majority vote: This classifier chooses the class label that is most frequent in the training data. It is known as the `ZeroR` classifier in Weka.

Winnow: We used the unbalanced Winnow classifier [34] with default parameters $\alpha = 2$, $\beta = .5$, and start weight $w = 2$.

ANN: ANN implements a multilayer perceptron with back-propagation learning, see, e.g., [8]. The default model structure was used in our experiments, i.e., one hidden layer, containing a number of nodes equal to the average of the number of classes and the number of attributes. The learning rate was .3 and the momentum .2. The back-propagation was performed for 500 epochs. The classifier is called `MultilayerPerceptron` in Weka.

ADTree: The Alternating Decision Tree [35] used in our experiments was based on exhaustive search, and the classifier was improved using 10 boosting iterations. The ADTree implementation currently only supports two-class problems, so for the "{0} – {9}" dataset we used the `MultiClassClassifier` wrapper to generate 10 classification problems (each classifier learned to separate one digit from the rest), and chose the class that was most probable.

1-NN: This classifier, called `IB1` in Weka, implements the nearest-neighbour classifier [36].

SVM: SVM denotes the support sector machines using the sequential minimal optimisation algorithm for training the classifier [37]. The "City-block distance" was used as distance-measure. The classifier is called `SMO` in Weka.

Naïve Bayes: The Naïve Bayes model [1] without virtual counts for parameter learning.

TAN: The TAN model [3] is learned in Weka by choosing the `BayesianNetwork` classifier and `TAN` as search method. The reported results were generated using virtual count $N' = .5$.

ID3: The ID3 decision tree [38].

Logistic Regression: The logistic regression classifier was enhanced with Ridge regression (parameter value 10^{-8}) to avoid local maxima [39].

Radial Basis Functions: The RBF network was generated using $k = 2$ clusters (found by the k-means algorithm); thereafter logistic regression models were fit to each cluster (as above) [40]. The classifier is called `RBFNetwork` in Weka.

AODE: The Aggregating One-Dependence Estimators-classifier [41] was learned with frequency limit $f = 1$.

HNB: The Hidden Naive Bayes classifier [42].

BayesNet: A Bayesian network structure is learned from data using the K2 search algorithm [43]. Parameters are estimated using $N' = .5$ virtual counts.

References

[1] R. O. Duda, P. E. Hart, Pattern Classification and Scene Analysis, John Wiley & Sons, New York, 1973.

[2] P. Domingos, M. Pazzani, On the optimality of the simple Bayesian classifier under zero-one loss, Machine Learning 29 (2–3) (1997) 103–130.

[3] N. Friedman, D. Geiger, M. Goldszmidt, Bayesian network classifiers, Machine Learning 29 (2–3) (1997) 131–163.

[4] M. Bressan, J. Vitrià, Improving naive Bayes using class-conditional ICA, in: Advances in Artificial Intelligence, Vol. 2527 of Lecture Notes in Artificial Intelligence, Springer-Verlag, Berlin, Germany, 2002, pp. 1–10.

[5] A. Hyvärinen, J. Karhunen, E. Oja, Independent Component Analysis, Adaptive and learning systems for signal processing, communications, and control, John Wiley & Sons, New York, 2001.

[6] H. Langseth, T. D. Nielsen, Latent classification models, Machine Learning 59 (3) (2005) 237–265.

[7] Z. Ghahramani, G. E. Hinton, The EM algorithm for mixtures of factor analyzers, Technical Report CRG-TR-96-1, Department of Computer Science, University of Toronto, Canada (1996).

[8] T. M. Mitchell, Machine Learning, McGraw Hill, Boston, MA., 1997.

[9] B. D. Ripley, Pattern Recognition and Neural Networks, Cambridge University Press, Cambridge, UK, 1996.

[10] G. J. McLachlan, Discriminant Analysis and Statistical Pattern Recognition, John Wiley & Sons, New York, 2004.

[11] D. B. Rubin, D. T. Thayer, EM algorithms for ML factor analysis, Psychometrika 47 (1) (1982) 69–76.

[12] D. J. Bartholomew, Latent Variable Models and Factor Analysis, Charles Griffin & Co., London, UK, 1987.

[13] J. D. Martin, K. VanLehn, Discrete factor analysis: Learning hidden variables in Bayesian networks, Technical Report LRDC-ONR-94-1, Department of Computer Science, University of Pittsburgh, `http://www.pitt.edu/~vanlehn/distrib/Papers/Martin.pdf` (1994).

[14] J. J. Hull, A database for handwritten text recognition research, IEEE Transactions on Pattern Analysis and Machine Intelligence 16 (5) (1994) 550–554.

[15] C. E. Rasmussen, C. K. Williams, Gaussian Processes for Machine Learning, Adaptive Computation and Machine Learning, The MIT Press, Cambridge, MA, 2006.

[16] T. S. Jaakkola, Variational methods for inference and estimation in graphical models, Ph.D. thesis, Dept. of Brain and Cognitive Sciences, Massachusetts Institute of Technology (1997).

[17] M. E. Tipping, Probabilistic visualisation of high-dimensional binary data, in: M. S. Kearns, S. A. Solla, D. A. Cohn (Eds.), Advances in Neural Information Processing Systems 12, The MIT Press, 1999, pp. 592– 598.

[18] M. I. Jordan, Z. Ghahramani, T. S. Jaakkola, L. K. Saul, An introduction to variational methods for graphical models, Machine Learning 37 (1999) 183–233.

[19] T. Jaakkola, M. I. Jordan, Bayesian parameter estimation via variational methods, Statistics and Computing 10 (1999) 25–37.

[20] K. P. Murphy, A variational approximation for Bayesian networks with discrete and continuous latent variables, in: K. B. Laskey, H. Prade (Eds.), Proceedings of the Fifteenth Conference on Uncertainty in Artificial Intelligence, San Fransisco, CA., 1999, pp. 467–475.

[21] R. Kohavi, G. H. John, Wrappers for feature subset selection, Artificial Intelligence 97 (1–2) (1997) 273–324.

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

[23] A. Y. Ng, M. I. Jordan, On discriminative vs. generative classifiers: A comparison of logistic regression and naive Bayes, in: Advances in Neural Information Processing Systems 15, The MIT Press, Vancouver, British Columbia, Canada, 2002, pp. 841–848.

[24] R. Greiner, W. Zhou, Structural extension to logistic regression: Discriminative parameter learning of belief net classifiers, in: Proceedings of the Eighteenth National Conference on Artificial Intelligence, The AAAI Press, Menlo Park, CA., 2002, pp. 167–173.

[25] H. Wettig, P. Grünwald, T. Roos, P. Myllymäki, H. Tirri, When discriminative learning of Bayesian network parameters is easy, in: Proceedings of the Eighteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann Publishers, 2003, pp. 491–496.

[26] D. Grossman, P. Domingos, Learning Bayesian network classifiers by maximizing conditional likelihood, in: Proceedings of the Twentyfirst International Conference on Machine Learning, ACM Press, Banff, Canada, 2004, pp. 361–368.

[27] C. Burges, A tutorial on support vector machines for pattern recognition, Data Mining and Knowledge Discovery 2 (2) (1998) 955–974.

[28] J. Vomlel, Noisy-or classifier, International Journal of Intelligent Systems 21 (3) (2006) 281–398.

[29] R. Kohavi, A study of cross-validation and bootstrap for accuracy estimation and model selection, in: Proceedings of the Fourteenth International Joint Conference on Artificial Intelligence, Morgan Kaufmann Publishers, San Mateo, CA., 1995, pp. 1137–1143.

[30] C. M. Bishop, Discussion of "Bayesian treed generalized linear models" by H.A Chipman, E. I. George, and R. E. McCulloch, in: Proceedings of the Seventh Valencia International Meeting on Bayesian Statistics, Oxford University Press, 2002, pp. 98–101.

[31] G. Bouchard, Efficient bounds for the softmax function, applications to inference in hybrid models, Presentation at the Workshop for Approximate Bayesian Inference in Continuous/Hybrid Systems at NIPS-07 (2007).

[32] S. L. Lauritzen, Propagation of probabilities, means and variances in mixed graphical association models, Journal of the American Statistical Association 87 (420) (1992) 1098–1108.

[33] I. H. Witten, E. Frank, Data Mining: Practical machine learning tools with Java implementations, Morgan Kaufmann Publishers, San Francisco, CA, 2000.

[34] N. Littlestone, Learning quickly when irrelevant attributes abound: A new linear threshold algorithm, Machine Learning 2 (1988) 285–318.

[35] Y. Freund, L. Mason, The alternating decision tree learning algorithms, in: Proceedings of the Sixteenth International Conference on Machine Learning, 1999, pp. 124–133.

[36] D. Aha, D. F. Kibler, M. Albert, Instance-based learning algorithms, Machine Learning 6 (1991) 37–66.

[37] J. Platt, Machines using sequential minimal optimization, in: B. Schoelkopf, C. Burges, A. Smola (Eds.), Advances in Kernel Methods - Support Vector Learning, The MIT Press, 1998, pp. 185–208.

[38] R. Quinlan, Induction of decision trees, Machine Learning 1 (1986) 81–106.

[39] S. le Cessie, J. van Houwelingen, Ridge estimators in logistic regression, Applied Statistics 41 (1) (1992) 191–201.

[40] P. V. Yee, S. Haykin, Regularized radial basis function networks: Theory and applications, Adaptive and learning systems for signal processing, communications, and control, John Wiley & Sons, New York, 2001.

[41] G. Webb, J. Boughton, Z. Wang, Not so naive Bayes: Aggregating one-dependence estimators, Machine Learning 58 (1) (2005) 5–24.

[42] H. Zhang, L. Jiang, J. Su, Hidden naive Bayes, in: Proceedings of the Twentieth National Conference on Artificial Intelligence, The AAAI Press, 2005, pp. 919–924.

[43] G. F. Cooper, E. Herskovits, A Bayesian method for the induction of probabilistic networks from data, Machine Learning 9 (1992) 309–347.