Averaging, Maximum Penalized Likelihood and Bayesian Estimation for Improving Gaussian Mixture Probability Density Estimates

Dirk Ormoneit and Volker Tresp

D. Ormoneit is a member of the Graduiertenkolleg at the Department of Computer Science of the Technische Universität München, Germany. During part of the research for this paper he was visiting at the Department of Economics at the University of California, San Diego. This visit was supported by a grant from the German Academic Exchange Service ("DAAD-Doktorandenstipendium aus Mitteln des zweiten Hochschulsonderprogramms"). E-mail: ormoneit@informatik.tu-muenchen.de.

V. Tresp is with Siemens AG, Corporate Technology, Department of Information and Communications, 81730 München, Germany. E-mail: Volker.Tresp@mchp.siemens.de. Supported by grant number -01 IN 505 A9- from the Bundesministerium für Bildung, Wissenschaft, Forschung und Technologie.

Abstract

We apply the idea of averaging ensembles of estimators to probability density estimation. In particular we use Gaussian mixture models which are important components in many neural network applications. One variant of averaging is Breiman's "bagging", which recently produced impressive results in classification tasks. We investigate the performance of averaging using three data sets. For comparison, we employ two traditional regularization approaches, i.e. a maximum penalized likelihood approach and a Bayesian approach. In the maximum penalized likelihood approach we use penalty functions derived from conjugate Bayesian priors such that an EM algorithm can be used for training. In all experiments, the maximum penalized likelihood approach and averaging improved performance considerably if compared to a maximum likelihood approach. In two of the experiments, the maximum penalized likelihood approach outperformed averaging. In one experiment averaging was clearly superior. Our conclusion is that maximum penalized likelihood gives good results if the penalty term in the cost function is appropriate for the particular problem. If this is not the case, averaging is superior since it shows greater robustness by not relying on any particular prior assumption. The Bayesian approach worked very well on a low-dimensional toy problem but failed to give good performance in higher-dimensional problems.

Keywords

Gaussian Mixture Model, Probability Density Estimation, Penalized Likelihood, Ensemble Averaging, Bagging, EM Algorithm, Data Augmentation, Gibbs Sampling

I. Introduction

Gaussian mixtures model probability densities by weighted sums of normal distributions. Gaussian mixture models have found a number of important applications in neural computation. They are used to train radial basis function classifiers [1] and they are employed both in learning from patterns with missing features [2], [3] and active learning [4]. Their appeal is based to a high degree on the applicability of the EM (Expectation Maximization) learning algorithm, which can be implemented as a fast neural network learning rule [1], [5]. Severe problems arise, however, due to singularities and local maxima in the log-likelihood function. Particularly in high-dimensional spaces these problems frequently cause the computed density estimates to possess only relatively limited generalization capabilities in terms of predicting the densities at new data points.

As a solution to this problem we investigate the benefits of averaging an ensemble of esti-

mators to Gaussian mixture models. The idea of averaging has newly been introduced into the neural network community and was very successful when applied to neural networks trained as classifiers and regressors. The individual neural networks in the ensemble were either trained on identical data *(simple averaging)* and varied only since they converged into different local minima (an idea introduced by Perrone and Cooper [6]) or were trained on bootstrap samples of the training data set, a procedure which was coined "bagging" predictors by Breiman [7]. Alternatively, the individual networks can be trained on different subsets of the training data *(subset averaging)*. In this paper we apply the three averaging approaches to Gaussian mixture models and demonstrate that averaging can lead to improved models.

Averaging can be considered as a form of regularization since the effect of "overtraining" is reduced by averaging the predictions of models which converged into different local minima. To evaluate the averaging approach we review two more traditional approaches to regularization, i.e. a maximum penalized likelihood approach and a Bayesian approach. In the former regularization is achieved by adding a penalty term to the log-likelihood cost function. The penalty function we use is derived based on a conjugate Bayesian prior such that we can apply the EM algorithm to find the optimal parameter estimates. In the Bayesian approach we approximate the predictive distribution by averaging the forecasts of a sequence of parameter vectors which are selected according to the posterior probability density of the parameter vectors. Interestingly, the Bayesian approach is related to both regularization (via the prior) and averaging (by averaging models with different parameters).

On a historical note, the application of Gaussian mixture models for statistical inference can be traced back to Pearson [8], who investigated the case of a Gaussian mixture with two components. Ever since, considerable interest has been focused on various methods to estimate the mixture parameters. While Pearson originally followed a method of moments approach, maximum likelihood has later become the method of choice. The theoretical framework of the EM algorithm was first introduced in a paper by Dempster, Laird, and Rubin [9], even though the resulting update formulas for the Gaussian mixture parameters had been used previously (e.g. Hasselblad [10], Day [11], Wolfe [12], Duda and Hart [13]).

A thorough treatment of this topic can be found in [14].

A detailed illustration of the Bayesian perspective on density estimation using Gaussian mixtures was recently provided by Roeder and Wasserman [15]. The sampling approach to Bayesian inference in the context of Gaussian mixture models —in the form used in this paper— was first described by Diebolt [16]. An interesting extension of Bayesian sampling to cases where the number of Gaussian components is unknown has recently been suggested by Richardson and Green [17]. Green [18] was also one of the first authors who used the EM algorithm for maximum penalized likelihood estimation. The first application of Gaussian mixture models to neural networks is attributed to Nowlan [1] who used them for the training of radial basis function networks.

The averaging approach was introduced to the neural network community by Perrone and Cooper [6] although related approaches had been used by other authors (Wolpert [19], Drucker, Schapire and Simard [20]). Breiman suggested training the individual predictors on resamples of the original data set, which led to the "bagging" algorithm [7]. Even more recent developments include "arcing" [21], [22], where the resampling probabilities are adapted dynamically to further improve the predictive performance of the averaged forecast.

The paper is organized as follows. In the following section we introduce the Gaussian mixture model and the EM algorithm. In section III we discuss the different averaging approaches and in section IV we introduce the maximum penalized likelihood approach including the associated EM learning rules. The Bayesian approach is presented in section V. In section VI we describe experimental results comparing the three approaches and in section VII we present conclusions.

II. Gaussian Mixtures and the EM Algorithm

Consider the problem of estimating the probability density of a continuous random vector $x \in \mathcal{R}^d$ based on a set $x^* = \{x^k | 1 \leq k \leq m\}$ of i.i.d. realizations of x. As a density model we choose the class of Gaussian mixtures $p(x|\Theta) = \sum_{i=1}^{n} \kappa_i N(x|\mu_i, \Sigma_i)$, where the restrictions $\kappa_i \geq 0$ and $\sum_{i=1}^{n} \kappa_i = 1$ apply. Θ denotes the parameter vector $(\kappa_i, \mu_i, \Sigma_i)_{i=1}^{n}$.

The $N(x|\mu_i, \Sigma_i)$ are multivariate normal densities:

$$N(x|\mu_i, \Sigma_i) = (2\pi)^{-\frac{d}{2}} |\Sigma_i|^{-1/2} \exp\left[-1/2(x-\mu_i)^t \Sigma_i^{-1}(x-\mu_i)\right].$$

The Gaussian mixture model is well suited to approximate a wide class of continuous probability densities. Based on the model and given the data x^*, we may formulate the log-likelihood as

$$l(\Theta) = \log\left[\prod_{k=1}^{m} p(x^k|\Theta)\right] = \sum_{k=1}^{m} \log \sum_{i=1}^{n} \kappa_i N(x^k|\mu_i, \Sigma_i).$$

Maximum likelihood parameter estimates $\hat{\Theta}$ may efficiently be computed with the EM (Expectation Maximization) algorithm [9]. It consists of the iterative application of the following two steps:

1. In the E-step, based on the current parameter estimates, the posterior probability that unit i is responsible for the generation of pattern x^k is estimated as

$$h_i^k = \frac{\kappa_i N(x^k|\mu_i, \Sigma_i)}{\sum_{j=1}^{n} \kappa_j N(x^k|\mu_j, \Sigma_j)}. \tag{1}$$

2. In the M-step, we obtain new parameter estimates (denoted by the prime):

$$\kappa_i' = \frac{1}{m} \sum_{k=1}^{m} h_i^k \tag{2}$$

$$\mu_i' = \frac{\sum_{k=1}^{m} h_i^k x^k}{\sum_{l=1}^{m} h_i^l} \tag{3}$$

$$\Sigma_i{}' = \frac{\sum_{k=1}^{m} h_i^k (x^k - \mu_i')(x^k - \mu_i')^t}{\sum_{l=1}^{m} h_i^l}. \tag{4}$$

   Note that κ_i' is a scalar, whereas μ_i' denotes a d-dimensional vector and $\Sigma_i{}'$ is a $d \times d$ matrix.

It is well known that training neural networks as predictors by maximizing the likelihood can lead to overfitting. The problem of overfitting is typically even more severe in density estimation due to singularities in the log-likelihood function: Obviously, the model likelihood becomes infinite in a trivial way if we concentrate all the probability mass on one or several samples of the training set. This is the case if the center of a Gaussian coincides with one of the data points and Σ_i approaches the null-matrix. Figure 1 compares the true and the estimated probability density using a toy problem. As can be seen, the contraction

of the Gaussians results in (possibly infinitely) high peaks in the Gaussian mixture density estimate. The problem of overfitting is even more severe in high-dimensional spaces. In the following sections we will compare three methods which can be used to improve the density estimates.

III. Averaging Gaussian Mixtures

In this section we discuss the application of averaging to Gaussian mixture models with the goal of achieving improved probability density estimates. As mentioned in the introduction the averaging over neural network ensembles has been applied previously to regression and classification tasks [6].

Applied to Gaussian mixture models the approach is to predict the density of an unknown data point x by forming the averaged density estimate $\bar{P} = \frac{1}{N}\sum_{i=1}^{N} P_i$, where each $P_i = p(x|\Theta_i)$ is the prediction of a Gaussian mixture density. Without going too much into the theoretical motivation of averaging (see, e.g. [6], [23], [24], [7], [25]) the advantage of averaging can be understood by noting that since $V(\bar{P}) = \frac{1}{N^2}\sum_{i=1}^{N} V(P_i) + \frac{2}{N^2}\sum_{i=1}^{N}\sum_{j=i+1}^{N} Cov(P_i, P_j)$, the variance of the averaged density estimate $V(\bar{P})$ is smaller than the average variance of an individual estimator $V(P_i)$ unless the predictors are perfectly correlated. Perrone ([26], pg. 22) generalized this result and showed, by using the Cauchy inequality, that even for averaged *biased* predictors the mean squared error of the average is always less or equal to the mean squared error of the population average.

In this paper we use three different averaging approaches. In *simple averaging*, several different Gaussian mixture models are trained on the complete training data set using EM. The Gaussian mixture models differ since the optimization procedure typically terminates in different local minima if different starting points are used. In the other two approaches the variability is further increased by training each model on a resampled version of the original data set. If we resample the data without replacement, the size of each training set is reduced, in our experiments to 70% of the original *(subset averaging)*. Finally, we resampling with replacement, which has recently been proposed under the name "bagging" by Breiman [7], who has achieved dramatically improved results in several classification tasks. Note that some samples will be present more than once in the resampled data set, while others will be left out completely in "bagging". Breiman also noted that a

considerable improvement of the prediction can only result if the estimation procedure is relatively unstable in that small changes in the training set could cause large changes in the resulting predictors. As discussed, this is particularly the case for Gaussian mixture training. We therefore expect "bagging" to be well suited for our task.

IV. The Maximum Penalized Likelihood Approach

In this section we consider maximum penalized likelihood estimation. Here, a penalty term is added to the log-likelihood function as a regularizer. The maximum penalized likelihood approach is equivalent to the maximum a posterior (MAP) parameter estimate in a Bayesian approach if we interpret the penalty as the logarithm of the prior distribution.

In particular if we chose the logarithm of a *conjugate* prior as the penalty function we can derive EM update rules to obtain the optimal parameter estimates.[1] A conjugate prior of a single multivariate normal density is a product of a normal density $N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i)$ and a Wishart density $Wi(\Sigma_i^{-1}|\alpha_i, \beta_i)$ [28]. A proper conjugate prior for the mixture weightings $\kappa = (\kappa_1, ..., \kappa_n)$ is a Dirichlet density $D(\kappa|\gamma)$. These densities are defined in appendix I. Consequently, the prior of the Gaussian mixture is the product

$$D(\kappa|\gamma)\prod_{i=1}^{n} N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i, \beta_i).$$

The MAP parameter estimate maximizes the log-posterior

$$\begin{aligned} l_p(\Theta) &= \sum_{k=1}^{m} \log \sum_{i=1}^{n} \kappa_i p(x^k|i, \mu_i, \Sigma_i) + \log D(\kappa|\gamma) \\ &\quad + \sum_{i=1}^{n} [\log N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i) + \log Wi(\Sigma_i^{-1}|\alpha_i, \beta_i)]. \end{aligned}$$

As in the unregularized case, we may use the EM-algorithm to find a local maximum of $l_p(\Theta)$ (for a derivation, see appendix I.). The E-step is identical to (1). The M-step becomes

$$\kappa_i' = \frac{\sum_{k=1}^{m} h_i^k + \gamma_i - 1}{m + \sum_{i=1}^{n} \gamma_i - n} \tag{5}$$

$$\mu_i' = \frac{\sum_{k=1}^{m} h_i^k x^k + \eta_i \nu_i}{\sum_{l=1}^{m} h_i^l + \eta_i} \tag{6}$$

$$\Sigma_i' = \frac{\sum_{k=1}^{m} h_i^k (x^k - \mu_i')(x^k - \mu_i')^t + \eta(\mu_i' - \nu_i)(\mu_i' - \nu_i)^t + 2\beta_i}{\sum_{l=1}^{m} h_i^l + 2\alpha_i - d}. \tag{7}$$

[1] A family $\mathcal{F}$ of probability distributions on Θ is said to be *conjugate* if, for every $p \in \mathcal{F}$, the posterior $p(\Theta|x^*)$ also belongs to $\mathcal{F}$ (see [27]).

We found that the described EM procedure leads to convergence after few hundred iterations in our experiments. Consequently, we chose to restrict the training time to 500 update steps. To determine appropriate values for the hyper-parameters α_i, β_i, γ_i, η_i, and ν_i we formulate our beliefs about the data generation process in terms of a prior distribution. For conjugate priors, the hyper-parameters may be interpreted as sufficient statistics of an additional set of artificial data points. In the following experiments we define three "equivalent sample sizes" ω_κ, ω_μ and ω_Σ, denoting the size of the artificial data set associated with each parameter.[2]

Consider the case where an additional data set y^* of size m' is generated by a Gaussian mixture of n components. Let y_i^* denote the subset of y^* generated by Gaussian i. In the absence of additional information,[3] Θ is after the observation of y^* distributed according to

$$\kappa \sim D(m_1 + 1, \ldots, m_n + 1), \quad \mu_i \sim N(\overline{y_i^*}, m_i^{-1}\Sigma_i), \quad \Sigma_i^{-1} \sim Wi\left(\frac{m_i + d}{2}, \frac{m_i}{2}\tilde{S}_i\right),$$

where $m_i = |y_i^*|$, $\overline{y_i^*} = \sum_{x \in y_i^*} x$, $S_i = \sum_{x \in y_i^*}(x - \nu_i)(x - \nu_i)^t$, and $\tilde{S}_i = \frac{1}{m_i}S_i$. Comparing this distribution to the definition of the conjugate prior (9) to (11) in appendix I we see that we can make the following identifications:

$$\alpha_i = \frac{\omega_\Sigma + d}{2}, \;\; \beta_i = \frac{\omega_\Sigma}{2}\tilde{S}_i, \;\; \gamma_i = \omega_\kappa + 1, \;\; \eta_i = \omega_\mu, \;\; \nu_i = \overline{y_i^*}, \quad \text{for } i = 1, \ldots, n.$$

This definition implies that each Gaussian generated an equal number of samples in the artificial data set, representing our prior belief that each Gaussian is equally likely to have generated a new data point. What remains to be done is to choose concrete values for the statistics $\overline{y_i^*}$ and $\tilde{S}_i$. For our experiments, we chose $\overline{y_i^*} = 0$ and $\tilde{S}_i = I^{d \times d}$. This encodes our prior believe that centers are zero and that the covariance matrix is the unit matrix. The degree of regularization is now determined by simply varying the equivalent sample sizes ω_κ, ω_μ and ω_Σ. In our experiments we will set $\omega_\kappa = \omega_\mu = 0$ and only vary ω_Σ, i.e. we only put a prior weight on Σ.

[2]The term "equivalent sample size" is taken from Heckerman and Geiger [29], who apply a similar approach to learning in Bayesian networks.

[3]i.e. assuming an (improper) non-informative uniform (hyper-) prior distribution. We employ the uniform prior in place of the more commonly used Jeffereys prior in order to obtain the EM update rules for maximum likelihood estimation (2) to (4) as the special case of the maximum penalized likelihood update rules (5) to (7) in which ω_κ, ω_μ and ω_Σ are equal to zero.

Figure 2 shows the predictive density for two values of ω_Σ. If ω_Σ is chosen rather small (figure 2, left), overfitting still occurs. As ω_Σ increases the density estimate becomes smoother and the covariance matrices of the Gaussians approach the prespecified matrix $\tilde{S}_i$ (figure 2, right). Typically, the optimal value for ω_Σ is not known a priori. In the following experiments we report results for various values of ω_Σ. If a validation set is available, one may choose that value which leads to the best performance, analogous to the determination of the optimal weight decay parameter in neural network training. As is apparent from the update equations (5) to (7), only a few additional computations are required for optimizing the maximum penalized likelihood as compared to standard EM.

V. A Bayesian Approach

In contrast to the maximum penalized likelihood approach, in a Bayesian approach we derive the predictive distribution

$$p(x|x^*) = \int p(x|\Theta)p(\Theta|x^*)d\Theta.$$

By using a conjugate prior $p(\Theta)$ we can obtain an analytically closed formulation of $p(x|x^*)$. Unfortunately, $p(x|x^*)$ is a sum of n^{m+1} terms and therefore is typically approximated. We use a stochastic approximation to $p(x|x^*)$ by employing the "data augmentation" method [16]. Data augmentation is an instantiation of Gibbs sampling, where one exploits the hierarchical structure of mixture models to generate a Markov chain $(\Theta)_t$ with stationary distribution $p(\Theta|x^*)$. More specifically, one generates samples from the posterior of the parameters by the iterative application of the following two steps:

1. **Generate a set of indicator variables $z^* \sim p(z^*|x^*, \Theta)$,**
   where $z^* = \{z^k | 1 \leq k \leq m\}$ and $z^k \in \{z \in \{0,1\}^n | \sum_{i=1}^n z_i = 1\}$. z^k is interpreted as an indicator variable with $z_i^k = 1$ exactly if x^k was generated by Gaussian i and $z_i^k = 0$ otherwise. To generate z^k we first compute h_i^k according to (1), using the most recent sample of parameter values. Then, we generate z^k according to a multinomial distribution with $P(z_i^k = 1) = h_i^k$.
2. **Generate $\Theta' \sim p(\Theta|x^*, z^*)$:**
   z^*, as generated in step 1, defines a unique partitioning $\rho = (\rho_1, \ldots, \rho_n)$ with partitions $\rho_i = \{k \in \{1, \ldots, m\} | z_i^k = 1\}$ of the data set x^*. Using this notation, $p(\Theta|x^*, z^*)$ may

be written as

$$p(\Theta|x^*, z^*) = D(\kappa|\gamma^\rho) \prod_{i=1}^{n} N(\mu_i|\nu_i^\rho, \eta_i^{\rho^{-1}}\Sigma_i) Wi(\Sigma_i^{-1}|\alpha_i^\rho, \beta_i^\rho). \quad (8)$$

$\alpha_i^\rho, \beta_i^\rho, \eta_i^\rho, \nu_i^\rho$, as well as a derivation of $p(\Theta|x^*, z^*)$ are found in appendix II. Using the conditional independencies which are implied by (8), the new parameter values may be generated according to

$$\kappa' \sim D(\gamma^\rho), \quad \Sigma_i^{-1'} \sim Wi(\alpha_i^\rho, \quad \beta_i^\rho), \quad \mu_i' \sim N(\nu_i', \eta_i^{\rho^{-1}}\Sigma_i') \quad \text{for } i = 1, \ldots, n.$$

Note the strong similarity between these steps and the E- and the M-step of the EM algorithm. As shown in [16], the Markov chain $(\Theta)_t$ resulting from this procedure is ergodic and the distribution of Θ_t converges uniformly geometrically towards $p(\Theta|x^*)$. The values of the hyper-parameters may be chosen in the same way as it was proposed in section IV. Note that when starting from arbitrary initial values, one might have to let the algorithm run for some time before it approaches its stationary distribution. In our experiments, we ignored the first 50 samples and used the subsequent 500 samples to approximate the predictive distribution. The latter number was chosen rather high because a random process generated by Gibbs sampling typically produces serially dependent samples. This means that the generated sequence initially only covers a small portion of the parameter space. Since $(\Theta)_t$ is ergodic, the series will eventually switch from one mode of the posterior to the next, but it might take a long time until all modes have been discovered. In our experiments we could not observe improvements after a few hundred iterations, such that the mentioned 500 samples should suffice to approximate $p(x|x^T)$.

VI. Experiments and Results

To assess the practical advantage resulting from averaging and regularization, we used the density estimates to construct classifiers and compared the resulting prediction accuracies using two toy problems and one real-world problem. The reason is that the generalization error of density estimates in terms of the likelihood based on the test data is rather unintuitive, whereas performance on a classification problem provides a good impression of the degree of improvement. Gaussian mixtures have previously been applied

for discriminant analysis by Hastie and Tibshirani [30] as well as Kambhatla and Leen [31].

Assume we have a set of m labeled data $z^* = \{(x^k, l^k)|k = 1, ..., m\}$, where $l^k \in \Upsilon = \{1, ..., C\}$ denotes the class label of each input x^k. A classifier of new inputs x is yielded by choosing the class l with the maximum posterior class-probability $p(l|x)$. The posterior probabilities may be derived from the class-conditional data likelihood $p(x|l)$ via Bayes theorem: $p(l|x) = p(x|l)p(l)/p(x) \propto p(x|l)p(l)$. The resulting partitions of the input space are optimal for the true $p(l|x)$. A viable way to approximate the posterior $p(l|x)$ is to estimate $p(x|l)$ and $p(l)$ from the sample data. In the following problems we have $C = 2$, so that we have to train two density estimators to approximate $p(x|1)$ and $p(x|2)$. The percentage of samples in z^* belonging to class l may be used as an estimate of $p(l)$.

A. Toy Problem I

In the first toy problem, the task is to classify the two sets of circularly arranged data shown in figure 3. As is apparent from the figure, the two distributions are distinguished by having different centers. The precise algorithm to generate these data sets is described in appendix III. We generated 200 data points per class and subdivided them into two sets of 100 data points. The first was used for training, the second to test the generalization performance. Prior to the training, the centers of the Gaussians were initialized randomly. The mixture weightings and the covariance matrices were set to fixed values to reduce the tendency of the learning algorithms to converge to "extreme" solutions. As a network architecture we chose a Gaussian mixture with 20 units. We found that the number of mixture components is not critical in our application. A thorough discussion of model selection techniques which may be applied in this context is given by Cheeseman et al. [32].

Table I summarizes the results. Row 1 shows the results of the maximum likelihood approach, and rows 2-4 show the performance of the averaging approaches. Each of the averaging forecasts is the average of a population of 50 individually trained networks. We chose to do 50 replications because this is the value which Breiman used successfully in his "bagging" experiments. Row 2 shows the results without resampling ("simple averaging"). The following rows show the results of resampling by drawing 70% subsamples from

the original data set without replacement ("subset averaging") and resampling with replacement ("bagging"). For the maximum penalized likelihood approach (rows 5-13) and the Bayesian approach (rows 14-21), we report results for various choices of the hyperparameter ω_Σ. The performances on the training set and the test set are measured in terms of the model log-likelihood such that larger values indicate a better performance. We report separate log-likelihoods for the Gaussian mixture models for class A and B, as well as the classification accuracy on the test set. The indicated numbers are average results from 20 simulations, for each of which we generated new training and test data sets. The numbers in brackets denote the standard deviations σ of the results. Assuming that the 20 results for each experiment are selected from a normal population, their mean times $\sigma/\sqrt{20}$ is distributed according to a t distribution with 19 degrees of freedom. Multiplying σ with $t_{(1-95\%)/2}/\sqrt{20} = 0.4680$ thus yields 95% confidence intervals. The best result in each category is underlined.

The results in table I demonstrate the considerable improvement which can be achieved with averaging, the maximum penalized likelihood approach, and the Bayesian approach. The best averaging approach is "bagging", which improved the classification performance on the generalization data set by 2.17 percentage points in comparison to the maximum likelihood approach. In comparison, the improvement achieved by averaging density estimators trained on identical data ("simple averaging") is only 1.4 percentage points. This indicates that the different density estimators tend to be correlated if trained on identical data. The resampling approach without replacement ("subset averaging") shows an improvement of 1.7 percentage points in the classification. This indicates that an important element in making averaging work is the variation of the data that each density estimator is trained on. For maximum penalized likelihood and the Bayesian approach, the performance depends entirely on the appropriate choice of the regularization parameter ω_Σ. For the optimal value of ω_Σ (between 0.1 and 0.2), both outperform averaging, with respective improvements of 3.25 and 3.67 percentage points over maximum likelihood. This is in accordance with the work of Taniguchi and Tresp [24], who showed, for an ensemble of neural networks, that averaging improves the performance of unregularized neural networks but averaged unregularized neural networks cannot achieve quite the performance

of optimally regularized neural networks.

B. Toy Problem II

In our second toy experiment, we used a data generation mechanism similar to that used in the previous section to generate two classes of ten-dimensional observations. The 400 samples that were generated for each of the two classes A and B are distributed in the form of two ten-dimensional hyper-spheres with different centers. A precise description of the data-generating mechanism is provided in appendix III. Half of the data in each class were used to train a Gaussian mixture consisting of 20 units; the rest served as a test set. The training of the individual estimators was done as described in the previous section.

The results are reported in table II. Also in the high-dimensional environment averaging and maximum penalized likelihood lead to significant improvements if compared to maximum likelihood estimation. The best averaging approach is again bagging with an improvement of 4.06 percentage points. Note that this time the difference between bagging and averaging without resampling is smaller than in the previous experiment. Maximum penalized likelihood decisively outperforms averaging, with an improvement of 9.11 percentage points relative to maximum likelihood estimation. It is interesting to observe that the Bayesian approach produced even worse results than maximum likelihood this time. One possible reason for the poor performance of the Bayesian approach lies in numerical difficulties that are involved in the data augmentation approach. Naturally, there arise situations where in step 1 of the sampling algorithm of section V individual Gaussians are not assigned any samples in x^*. Sampling from the posterior of such a Gaussian thus reduces to sampling from its prior, which (for the choice $\omega_\mu = 0$) is improper at least with regard to the centers μ_i. Such situations lead to numerical difficulties and a distortion of the stationary distribution. A discussion of non-informative prior distributions which can be applied to avoid this problem can be found in [15]. Our experiments indicate that the problem is more urgent in high-dimensional environments, where the Gaussians are distributed more sparsely among the data points.

C. BUPA Liver Disorder Data

As a third task, we applied our methods to a real-world problem from medicine. The objective is to detect liver disorders which might arise from excessive alcohol consumption. Available information consists of five blood tests and a measure of the patients' daily alcohol consumption. We subdivided the 345 available samples into a training set of 200 samples and a test set of 145 samples. The results of our experiments, using a Gaussian mixture of 12 units, are shown in table III. The setup is the same as in the previous sections. A resample is created by randomly splitting the original data set (of one class) into disjoint training and test sets. The results of the density estimation for each class (A: no disorder, B: disorder) are reported separately.

All averaging approaches show dramatic improvements on this data set, both with respect to the maximum likelihood approach and with respect to the maximum penalized likelihood approach. The maximum penalized likelihood approach with the optimal choice $\omega_\Sigma = 0.05$ shows a moderate improvement of 3.13 percentage points in the classification accuracy with respect to the maximum likelihood estimate. All averaging approaches show an improvement of about 10 percentage points in comparison to the maximum likelihood approach. The most remarkable result, however, is that bagging outperforms all other methods consistently under all performance measures. In particular, the bagging results are better than the results for simple averaging, which indicates that the increased diversity of the individual forecasts due to the resampling is also beneficial for the estimation in this case. The relatively poor performance of the Bayesian approach supports our hypothesis that the numerical limitations of data augmentation are particularly troublesome in high-dimensional environments.

This is an example where averaging clearly outperforms the maximum penalized likelihood approach. We suspect that this is a consequence of the prior assumption which is implicit in the penalty term, i.e. that the covariance matrix of each Gaussian equals the unity matrix. This assumption might not be appropriate here, which would also be another reason for the poor performance of the Bayesian approach. Of course, the "true" prior is unknown in this real-world example. Averaging is independent of any prior assumption and is therefore more robust, a clear advantage of this method.

VII. Conclusion

In this paper we applied the idea of averaging to Gaussian mixture models. In addition, we reviewed two alternative approaches to regularization: a maximum penalized likelihood approach and a Bayesian approach. Averaging and the maximum penalized likelihood approach always performed better than the maximum likelihood approach. The Bayesian approach gave good performance on a low-dimensional toy data set but failed on the two higher-dimensional problems with ten respectively six dimensions. We explain the poor performance here with instabilities in the sampling approach in high dimensions (note that, to our knowledge, these are the first experimental results with the Bayesian sampling approach where the dimension of the input space is larger than two; in [16], only experiments with very simple one-dimensional problems are reported). In the two toy data sets the optimally regularized Gaussian mixture model performs better than the averaged Gaussian mixture models. This is in accordance with the results reported by Taniguchi and Tresp with respect to averaging neural networks. They showed that averaging unregularized estimators leads to a great improvement in performance with respect to the unregularized estimators. On the other hand the averaged unregularized estimator is typically worse than an optimally regularized estimator.

The BUPA data seem to be an exception to the rule: here averaging clearly outperforms the maximum penalized likelihood approach. Our explanation is that the prior assumption which is represented by the penalty term might be inappropriate here. Although the maximum penalized likelihood approach still outperforms the maximum likelihood approach it is inferior to averaging. Since averaging makes fewer assumptions than the maximum penalized likelihood approach it appears to be more robust. If one compares the different averaging approaches we conclude that either all three (simple averaging, subset averaging, bagging) perform approximately equally well (as in the ten-dimensional toy problem) or bagging is significantly better (as in the two-dimensional toy-problem and the BUPA data set). This is in accordance with the results of Taniguchi and Tresp for neural networks [25]. It appears that in the ten-dimensional toy problem the variation achieved by the local minima in the error surface is sufficient and subsampling does not add significantly to the variation. Finally, we remark that averaging is computationally

expensive: we need to train N (in our examples 50) neural networks instead of only one. The maximum penalized likelihood and the Bayesian approaches are inexpensive in comparison to averaging but require a search for the optimal regularization parameters. Also this requires the training of a considerable number of networks, in particular if more than one hyper-parameter is unknown.

VIII. Acknowledgments

We would like to thank Mike Titterington, Trevor Hastie, Leo Breiman, Reimar Hoffmann and Todd Leen for helpful discussions. The liver disorder data were gathered by BUPA Medical Research and commendably made available through the UCI Machine Learning Database.

Appendix

I. A variant of the EM algorithm to compute MAP estimates

In the maximum penalized likelihood approach, we find the parameters which maximize $l(\Theta, x^*) = \log p(\Theta, x^*) = \log p(x^*|\Theta) + \log p(\Theta)$ with respect to Θ, where the first term is the log-likelihood and the second term is the logarithm of the prior parameter distribution. As mentioned in section IV, we choose the prior from the conjugate prior distribution family of the likelihood to yield a tractable posterior distribution. The prior distribution is a product of a Dirichlet-, a Normal- and a Wishart-distribution:

$$p(\Theta) = D(\kappa|\gamma)\prod_{i=1}^{n} p(\mu_i, \Sigma_i), \text{ with } p(\mu_i, \Sigma_i) = N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i, \beta_i),$$

$$\text{where} \qquad D(\kappa|\gamma) = b(\gamma)\textstyle\prod_{i=1}^{n} \kappa_i^{\gamma_i - 1}, \text{ with } \kappa_i \geq 0 \text{ and } \sum_{i=1}^{n} \kappa_i = 1 \tag{9}$$

$$N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i) = (2\pi)^{-\frac{d}{2}}|\eta_i^{-1}\Sigma_i|^{-1/2}\exp\left[-\frac{\eta_i}{2}(\mu_i - \nu_i)^t\Sigma_i^{-1}(\mu_i - \nu_i)\right] \tag{10}$$

$$Wi(\Sigma_i^{-1}|\alpha_i, \beta_i) = c(\alpha_i, \beta_i)|\Sigma_i^{-1}|^{\alpha_i - (d+1)/2}\exp\left[-tr(\beta_i\Sigma_i^{-1})\right], \tag{11}$$

where $\alpha > (d-1)/2$. $b(\gamma)$ and $c(\alpha_i, \beta_i)$ are normalizing factors. $tr(.)$ is the trace operator.

In the E-step of the EM algorithm, we determine the expected complete-data posterior (the unprimed parameters are the current estimates, the primed parameters are the new estimates):

$$Q(\Theta'|\Theta) = \sum_{k=1}^{m}\sum_{i=1}^{n} h_i^k\left[\log\kappa_i' + \log N(x^k|\mu_i', \Sigma_i')\right] + \log D(\kappa'|\gamma)$$

$$+\sum_{i=1}^{n}[\log N(\mu_i'|\nu_i,\eta_i^{-1}\Sigma_i') + \log Wi(\Sigma_i^{-1'}|\alpha_i,\beta_i)].$$

h_i^k is the posterior probability that x^k was generated by Gaussian i and is computed according to (1).

In the M-step, we maximize $Q(\Theta'|\Theta)$ with respect to Θ'. This task may be decomposed into the following two optimization problems:

$$\kappa' = \arg\max_{\kappa} \sum_{k=1}^{m}\sum_{i=1}^{n} h_i^k \log\kappa_i + \log D(\kappa|\gamma) \tag{12}$$

$$\begin{aligned}(\mu_i',\Sigma_i') &= \arg\max_{\mu_i,\Sigma_i} \sum_{k=1}^{m}\sum_{i=1}^{n} h_i^k \log N(x^k|\mu_i,\Sigma_i) \\ &\quad +\sum_{i=1}^{n}[\log N(\mu_i|\nu_i,\eta_i^{-1}\Sigma_i) + \log Wi(\Sigma_i^{-1}|\alpha_i,\beta_i)].\end{aligned} \tag{13}$$

Both are solved analytically by setting the derivatives with respect to the parameters to zero. In equation (12), we have to consider the constraint that $\sum_i \kappa_i = 1$, which is done using Lagrange multipliers. This yields the Lagrange function

$$L(\kappa',\lambda) = \sum_{i=1}^{n} h_i^k \log\kappa_i' + \log D(\kappa'|\gamma) - \lambda\left[\sum_{i=1}^{n}\kappa_i' - 1\right].$$

We thus have to solve

$$\begin{aligned}\frac{\partial}{\partial\kappa_i'}L = \sum_{i=1}^{n} h_i^k\kappa_i'^{-1} + \underbrace{(\gamma_i-1)\kappa_i'^{-1}}_{\frac{\partial}{\partial\kappa_i'}\log D(\kappa'|\gamma)} - \lambda &= 0 \\ \frac{\partial}{\partial\lambda}L = -\left[\sum_{i=1}^{n}\kappa_i' - 1\right] &= 0.\end{aligned}$$

Substituting and resolving for κ_i' yields (5).

To solve (13), we first have to compute the derivatives of the involved terms with respect to the elements of μ_i' and $\Sigma_i^{-1'}$. We will denote the corresponding gradients as $\nabla_{\mu_i'}$ and $\nabla_{\Sigma_i^{-1'}}$. Setting these to zero yields

$$\begin{aligned}\sum_{k=1}^{m} h_i^k\Sigma_i'^{-1}(x^k-\mu_i)\underbrace{-\eta_i\Sigma_i'^{-1}(\mu_i'-\nu_i)}_{\nabla_{\mu_i'}\log N(\mu_i'|\nu_i,\eta_i^{-1}\Sigma_i')} &= 0 \\ \frac{1}{2}\sum_{k=1}^{m} h_i^k\left[\Sigma_i' - (x^k-\mu_i)(x^k-\mu_i)^t\right] +\end{aligned}$$

$$\underbrace{+\frac{1}{2}\left[\Sigma_i' - \eta_i(\mu_i'-\nu_i)(\mu_i'-\nu_i)^t\right]}_{\nabla_{\Sigma_i'^{-1}} \log N(\mu_i'|\nu_i,\eta_i^{-1}\Sigma_i')} \underbrace{+(\alpha_i-(d+1)/2)\Sigma_i' - \beta_i}_{\nabla_{\Sigma_i'^{-1}} \log Wi(\Sigma_i'^{-1}|\alpha_i,\beta_i)} \;=\; 0.$$

To determine the gradients of $\log N(\mu_i'|\nu_i,\eta_i^{-1}\Sigma_i')$ and $\log Wi(\Sigma_i'^{-1}|\alpha_i,\beta_i)$, one has to compute the derivatives of several matrix operators, one of which is the logarithm of a determinant. It is helpful to know that

$$\left(\frac{\partial}{\partial a_{ij}} \log|A|\right)_{1\le i\le d, 1\le j\le d} = A^{-1} \qquad \text{for symmetric matrices } A \in \mathcal{R}^{d\times d}.$$

After some algebra, one gets the update equations (6) and (7) in section IV.

II. The posterior probability distribution

In this section we present a formal derivation of the posterior distribution $p(\Theta|x^*)$ used in section V. The likelihood $p(x^*|\Theta)$ may be written as

$$\begin{aligned} p(x^*|\Theta) &= \prod_{k=1}^{m} p(x^k|\Theta) = \prod_{k=1}^{m}\sum_{i=1}^{n} \kappa_i N(x^k|\mu_i,\Sigma_i) \\ &= \sum_{\rho\in\rho^*} \left[\prod_{i=1}^{n} \kappa_i^{|\rho_i|}\right] \cdot p(x^{\rho_1}|\mu_1,\Sigma_1)\cdot \ldots \cdot p(x^{\rho_n}|\mu_n,\Sigma_n). \end{aligned} \tag{14}$$

ρ^* denotes the set of all possible partitionings $\rho = (\rho_1,\ldots,\rho_n)$ of the set $\{1,\ldots,m\}$ into n disjunct subsets $\rho_1,\ldots,\rho_n$ with $\cup_{i=1}^n \rho_i = \{1,\ldots,m\}$. Furthermore, let $x^{\rho_i} = \{x^k \in x^* | k \in \rho_i\}$. $p(x^{\rho_i}|\mu_i,\Sigma_i)$ is the likelihood of μ_i,Σ_i given x^{ρ_i} and under the assumption that all the data-points in x^{ρ_i} were actually generated by Gaussian i.

If the prior distribution is given, we determine the posterior distribution via Bayes theorem:

$$p(\Theta|x^*) \;=\; \frac{p(x^*|\Theta)p(\Theta)}{p(x^*)} = \sum_{\rho\in\rho^*} D(\kappa|\gamma^\rho)\cdot p(\mu_1,\Sigma_1|x^{\rho_1})\cdot\ldots\cdot p(\mu_n,\Sigma_n|x^{\rho_n}).$$

To see this, we use the fact that $p(x^*) = p(x^{\rho_1})\cdot\ldots\cdot p(x^{\rho_n})$ for any ρ. γ^ρ is defined as $\gamma^\rho = (\gamma_1+|\rho_1|,\ldots,\gamma_n+|\rho_n|)$. The overall posterior is thus a sum of products of posteriors for the individual Gaussians $p(\mu_i,\Sigma_i|x^{\rho_i})$. To derive a closed formulation of the latter, we first write down the likelihood for the individual Gaussians:

$$p(x^{\rho_i}|\mu_i,\Sigma_i) = (2\pi)^{-\frac{|\rho_i|\cdot d}{2}} |\Sigma_i|^{-\frac{|\rho_i|}{2}} \exp\left[-\frac{|\rho_i|}{2}(\overline{x}_i-\mu_i)^t\Sigma_i^{-1}(\overline{x}_i-\mu_i) - \frac{1}{2}tr(S_i\Sigma_i^{-1})\right],$$

where $\overline{x}_i = \frac{1}{|\rho_i|}\sum_{k\in\rho_i} x^k$ and $S_i = \sum_{k\in\rho_i}(x^k - \overline{x}_i)(x^k - \overline{x}_i)^t = \sum_{k\in\rho_i} x^k x^{k^t} - |\rho_i|\overline{x}_i\overline{x}_i^t$.

In the next step, we multiply with the prior $p(\mu_i, \Sigma_i) = N(\mu_i|\nu_i, \eta_i^{-1}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i, \beta_i)$ to get the joint density $p(x^{\rho_i}, \mu_i, \Sigma_i)$:

$$\begin{aligned} p(x^{\rho_i}, \mu_i, \Sigma_i) &= (2\pi)^{-\frac{(|\rho_i|+1)\cdot d}{2}} \eta_i^{\frac{1}{2}} |\Sigma_i|^{-\frac{|\rho_i|+1}{2}} \exp\left[-\frac{|\rho_i|}{2}(\overline{x}_i - \mu_i)^t\Sigma_i^{-1}(\overline{x}_i - \mu_i) - \frac{\eta_i}{2}(\nu_i - \mu_i)^t\Sigma_i^{-1}(\nu_i - \mu_i)\right] \\ &\quad \cdot c(\alpha_i, \beta_i)|\Sigma_i^{-1}|^{\alpha_i - \frac{d+1}{2}} \exp\left[-tr((\beta_i + \frac{1}{2}S)\Sigma_i^{-1})\right] \\ &= c'(\alpha_i, \beta_i, \eta_i, \rho_i) \cdot N(\mu_i|\nu_i^\rho, \eta_i^{\rho^{-1}}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i^\rho, \beta_i^\rho), \end{aligned}$$

where

$$\begin{aligned} \alpha_i^\rho &= \alpha_i + \frac{|\rho_i|}{2} \\ \beta_i^\rho &= \beta_i + \frac{1}{2}S_i + \frac{1}{2}\frac{\eta_i|\rho_i|}{\eta_i + |\rho_i|}(\nu_i - \overline{x}_i)(\nu_i - \overline{x}_i)^t \\ \eta_i^\rho &= \eta_i + |\rho_i| \\ \nu_i^\rho &= \frac{1}{\eta_i + |\rho_i|}(\eta_i\nu_i + |\rho_i|\overline{x}_i). \end{aligned}$$

$c'(\alpha_i, \beta_i, \eta_i, \rho_i)$ is some term independent of μ_i and Σ_i which summarizes suitable normalizing factors. From the density property of $N(\mu_i|\nu_i^\rho, \eta_i^{\rho^{-1}}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i^\rho, \beta_i^\rho)$, it follows that $p(x^{\rho_i}) = c'(\alpha_i, \beta_i, \eta_i, \rho_i)$, such that

$$p(\mu_i, \Sigma_i|x^{\rho_i}) = N(\mu_i|\nu_i^\rho, \eta_i^{\rho^{-1}}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i^\rho, \beta_i^\rho)$$

and

$$p(\Theta|x^*) = \sum_{\rho\in\rho^*} D(\kappa|\gamma^\rho)\prod_{i=1}^n N(\mu_i|\nu_i^\rho, \eta_i^{\rho^{-1}}\Sigma_i)Wi(\Sigma_i^{-1}|\alpha_i^\rho, \beta_i^\rho).$$

For a given z^*, and thus for a given ρ, we obtain formula (8) in section V.

III. The toy data generation

In this section we describe the algorithm that was used to generate the two artificial data sets in sections VI-A and VI-B. To obtain samples with the circular structure illustrated in figure 3, we first generate samples $x^{k''}$ on the unit circle. The computationally simplest way to do this is by generating samples $x^{k'''}$ according to a multivariate normal distribution and subsequently projecting them onto the unit circle:

$$\begin{aligned} x^{k'''} &\sim N(x|0, I^d), \\ x^{k''} &= x^{k'''}/||x^{k'''}||_2. \end{aligned}$$

Each data point is then multiplied with another normally distributed random number to obtain the dispersion of the data around the unit circle:

$$\begin{aligned} \varepsilon^k &\sim N(x|0,\sigma^2), \\ x^{k\prime} &= (1+\varepsilon^k)x^{k\prime\prime}. \end{aligned}$$

Finally, we add a (class-dependent) translation constant to the first component of the random vector:

$$x_i^k = \begin{cases} x_1^{k\prime} + \mathit{offset} & \text{if } i = 1 \\ x_i^{k\prime} & \text{otherwise.} \end{cases}$$

The values of the parameters involved are:

- **Toy problem I:** $\sigma = 0.2, \mathit{offset} = \pm 0.5$.
- **Toy problem II:** $\sigma = 0.2, \mathit{offset} = \pm 0.3$.

References

[1] S. J. Nowlan, *Soft Competitive Adaption: Neural Network Learning Algorithms based on Fitting Statistical Mixtures*, Ph.D. thesis, School of Computer Science, Carnegie Mellon University, Pittsburgh, 1991.

[2] Z. Ghahramani and M. I. Jordan, "Supervised learning from incomplete data via an EM approach," in *Advances in Neural Information Processing Systems 6*. 1994, Morgan Kaufmann.

[3] V. Tresp, S. Ahmad, and R. Neuneier, "Training neural networks with deficient data," in *Advances in Neural Information Processing Systems 6*, Jack D. Cowan, Gerald Tesauro, and Joshua Alspector, Eds. 1994, pp. 128–135, Morgan Kaufman.

[4] D. A. Cohn, Z. Ghahramani, and M. I. Jordan, "Active learning with statistical models," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. 1995, pp. 705–712, The MIT Press.

[5] D. Ormoneit, "Estimation of probability densities using neural networks," M.S. thesis, Institut für Informatik, Technische Universität München, 1993.

[6] M. P. Perrone and L. N. Cooper, "When networks disagree: Ensemble methods for hybrid neural networks," in *Artificial Neural Networks for Speech and Vision*, R. J. Mammone, Ed. 1993, pp. 126–142, Chapman & Hall.

[7] L. Breiman, "Bagging predictors," *Machine Learning*, vol. 24, pp. 123–140, 1996.

[8] K. Pearson, "Contribution to the mathematical theory of evolution," *Philosophical Transactions of the Royal Society A*, vol. 185, pp. 71–110, 1894.

[9] A. P. Dempster, N. M. Laird, and D. B. Rubin, "Maximum likelihood from incomplete data via the EM algorithm," *Journal of the Royal Statistical Society B*, vol. 39, pp. 1–38, 1977.

[10] V. Hasselblad, "Estimation of parameters for a mixture of normal distributions," *Technometrics*, vol. 8, pp. 431–444, 1966.

[11] N. E. Day, "Estimating the components of a mixture of normal distributions," *Biometrica*, vol. 56, no. 3, pp. 463–474, 1969.

[12] J. H. Wolfe, "Pattern clustering by multivariate mixture analysis," *Multivariate Behavioral Research*, vol. 5, pp. 329–350, 1970.

[13] R. O. Duda and P. E. Hart, *Pattern Classification and Scene Analysis*, Wiley, 1973.

[14] R. A. Redner and H. F. Walker, "Mixture densities, maximum likelihood and the EM algorithm," *SIAM Review*, vol. 26, pp. 195–239, 1984.

[15] K. Roeder and L. Wasserman, "Practical Bayesian density estimation using mixtures of normals," *Journal of the American Statistical Assiciation*, vol. 92, pp. 894–902, 1997.

[16] J. Diebolt and C. P. Robert, "Estimation of finite mixture distributions through Bayesian sampling," *Journal of the Royal Statistical Society B*, vol. 56, no. 2, pp. 363–375, 1994.

[17] S. Richardson and P. J. Green, "On Bayesian analysis of mixtures with an unknown number of components," *Journal of the Royal Statistical Society B*, vol. 59, pp. 731–792, 1997.

[18] P. J. Green, "On the use of the EM algorithm for penalized likelihood estimation," *Journal of the Royal Statistical Society B*, vol. 52, no. 3, pp. 443–452, 1990.

[19] D. H. Wolpert, "Stacked generalization," *Neural Networks*, vol. 5, pp. 241–259, 1992.

[20] H. Drucker, R. Schapire, and P. Simard, "Improving performance in neural networks using a boosting algorithm," in *Advances in Neural Information Processing Systems 5*, 1993.

[21] Y. Freund and R. Schapire, "A decision-theoretic generalization of on-line learning and an application to boosting," *Journal of Computer and System Sciences*, vol. 55, no. 1, pp. 119–139, 1997.

[22] L. Breiman, "Bias, variance, and arcing classifiers," Tech. Rep., UC Berkeley, 1996.

[23] A. Krogh and J. Vedelsby, "Neural network ensembles, cross validation, and active learning," in *Advances in Neural Information Processing Systems 7*. 1995, Morgan Kaufman.

[24] V. Tresp and M. Taniguchi, "Combining estimators using non-constant weighting functions," in *Advances in Neural Information Processing Systems 7*, G. Tesauro, D. Touretzky, and T. Leen, Eds. 1995, pp. 419–426, Morgan Kaufman.

[25] M. Taniguchi and V. Tresp, "Averaging regularized estimators," *Neural Computation*, vol. 9, no. 5, pp. 1163–1178, 1997.

[26] M. P. Perrone, *Improving Regression Estimates: Averaging Methods for Variance Reduction with Extensions to General Convex Measure Optimization*, Ph.D. thesis, Brown University, 1993.

[27] C. P. Robert, *The Bayesian Choice*, Springer-Verlag, 1994.

[28] J. M. Bernardo and A. F. M. Smith, *Bayesian Theory*, Wiley & Sons, 1994.

[29] D. Heckerman and D. Geiger, "Likelihood and parameter priors for Bayesian networks," Tech. Rep. MSR-TR-95-54, Microsoft Research, 1995.

[30] T. Hastie and R. Tibshirani, "Discriminant analysis by Gaussian mixtures," Tech. Rep., AT&T Bell Labs and University of Toronto, 1994.

[31] N. Kambhatla and T. K. Leen, "Classifying with Gaussian mixtures and clusters," in *Advances in Neural Information Processing Systems 7*. 1995, Morgan Kaufman.

[32] P. Cheeseman, J. Kelly, M. Self, J. Stutz, W. Taylor, and D. Freeman, "AutoClass: A Bayesian classification system," in *Proceedings of the Fifth International Workshop on Machine Learning*. 1988, pp. 54–64, Morgan Kaufmann.

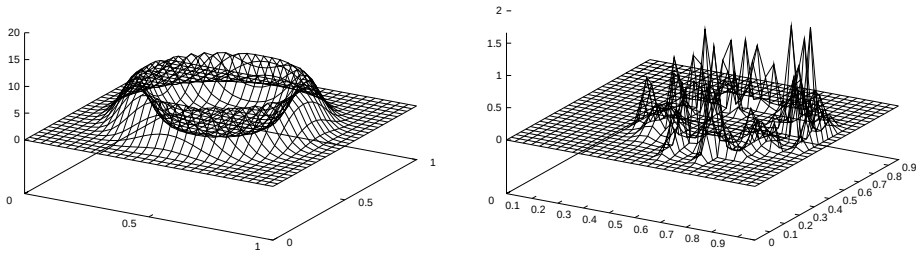

Fig. 1. *The true probability density (left) and the unregularized density estimate (right). We used a Gaussian mixture with 40 units and determined maximum likelihood parameters with the EM algorithm. The training set consisted of 100 observations.*

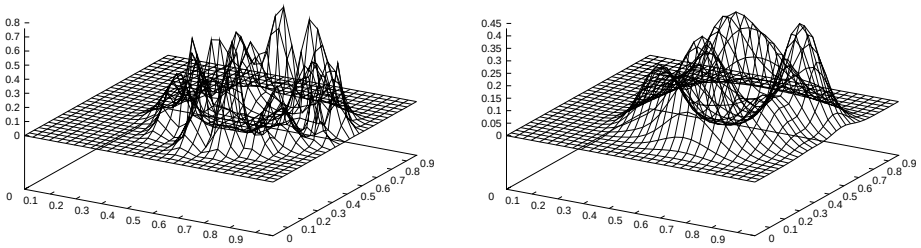

Fig. 2. *Regularized density estimates based on a Gaussian mixture with 40 units (left: $\omega_\Sigma = 0.05$, right: $\omega_\Sigma = 0.1$). The training set consisted of 100 observations.*

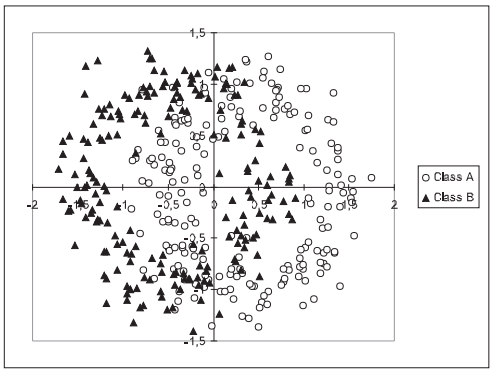

Fig. 3. *Distribution of features in the toy classification task (representative sample, 200 observations per class).*

Algorithm	**Log-Likelihood**				**Accuracy**
	Training		**Test**		
	A	B	A	B	
Max. Likelihood	-42.0 (48.5)	-12.2 (64.0)	-255.8 (44.7)	-245.1 (21.6)	79.03 % (2.6)
Averaging					
Simple Averaging	420.4 (168.4)	771.1 (221.1)	-203.9 (13.5)	-205.6 (13.9)	80.43 % (2)
Subset Averaging	667.5 (268.0)	994.0 (217.6)	-192.8 (9.7)	-193.4 (10.4)	80.73 % (1.8)
Bagging	1697.5 (332.1)	1894.3 (153.7)	-194.6 (9.0)	-197.9 (10.8)	81.2 % (1.4)
Penalized Likelihood					
$\omega_\Sigma = 0.02$	-138.3 (10.3)	-138.2 (9.1)	-186.4 (11.4)	-187.0 (11.3)	82.28 % (1.4)
$\omega_\Sigma = 0.05$	-144.4 (9.1)	-145.1 (9.3)	-177.6 (7.6)	-177.3 (7.9)	81.88 % (1.7)
$\omega_\Sigma = 0.1$	-151.0 (9.7)	-152.6 (8.7)	-173.6 (6.9)	-172.1 (6.5)	82.28 % (1.9)
$\omega_\Sigma = 0.2$	-157.4 (7.1)	-158.9 (6.2)	-171.7 (4.5)	-171.3 (4.2)	82.2 % (2.2)
$\omega_\Sigma = 0.5$	-169.9 (11.4)	-175.5 (10.1)	-181.3 (14.1)	-184.0 (15.9)	79.3 % (4.3)
$\omega_\Sigma = 1.0$	-185.9 (6.9)	-189.0 (5.4)	-196.0 (12.1)	-198.0 (11.1)	73.15 % (5.8)
$\omega_\Sigma = 2.0$	-197.9 (5.6)	-196.9 (4.2)	-207.4 (7.2)	-206.0 (8.3)	68.2 % (4.6)
$\omega_\Sigma = 5.0$	-208.4 (6.1)	-207.2 (4.1)	-216.0 (2.9)	-214.9 (2.5)	64.3 % (1.7)
$\omega_\Sigma = 10.0$	-218.0 (1.5)	-217.8 (1.3)	-220.0 (2.6)	-219.7 (2.4)	63.28 % (1.5)
Bayesian					
$\omega_\Sigma = 0.02$	-171.8 (22.0)	-181.5 (21.5)	-241.3 (24.5)	-243.6 (25.5)	75.95 % (3)
$\omega_\Sigma = 0.05$	-155.4 (14.3)	-166.3 (19.8)	-204.2 (8.5)	-208.0 (12.9)	80.83 % (2.1)
$\omega_\Sigma = 0.1$	-179.8 (16.8)	-179.6 (13.8)	-203.7 (8.8)	-200.9 (5.5)	82.7 % (1.3)
$\omega_\Sigma = 0.5$	-234.5 (9.9)	-242.2 (10.7)	-240.8 (8.4)	-247.4 (10.5)	77.5 % (2.9)
$\omega_\Sigma = 1.0$	-268.8 (15.2)	-264.5 (15.7)	-272.7 (14.9)	-268.3 (15.1)	70.75 % (3.7)
$\omega_\Sigma = 2.0$	-280.3 (13.1)	-279.4 (15.2)	-282.9 (12.7)	-281.8 (15.2)	66.0 % (4.6)
$\omega_\Sigma = 5.0$	-281.1 (10.7)	-277.8 (11.5)	-282.3 (11.0)	-279.2 (11.4)	65.98 % (3.7)
$\omega_\Sigma = 10.0$	-273.8 (6.4)	-273.6 (6.5)	-274.8 (6.5)	-274.6 (6.6)	64.55 % (2.1)

TABLE I

PERFORMANCES IN THE TOY CLASSIFICATION PROBLEM I.

Algorithm	**Log-Likelihood**				
	Training		**Test**		**Accuracy**
	A	B	A	B	
Max. Likelihood	-224.8 (33.9)	-214.1 (44.6)	-1020.5 (78.2)	-1010.6 (72.5)	72.59 % (0.2)
Averaging					
Simple Averaging	-15.7 (36.6)	-10.5 (45.9)	-772.1 (39.8)	-761.2 (49.3)	76.55 % (0.2)
Subset Averaging	-17.8 (40.2)	-3.4 (52.9)	-771.7 (40.3)	-761.3 (48.9)	76.39 % (0.2)
Bagging	-13.8 (37.4)	-5.4 (46.8)	-768.9 (41.2)	-761.6 (47.3)	76.65 % (0.2)
Penalized Likelihood					
$\omega_\Sigma = 0.01$	-230.6 (28.2)	-226.2 (38.3)	-968.0 (63.6)	-964.1 (62.5)	72.32 % (0.3)
$\omega_\Sigma = 0.1$	-344.6 (26.4)	-336.4 (40.2)	-760.9 (37.9)	-750.7 (37.8)	75.24 % (0.2)
$\omega_\Sigma = 0.2$	-453.4 (26.0)	-447.6 (39.1)	-676.4 (33.2)	-663.3 (33.8)	77.44 % (0.2)
$\omega_\Sigma = 0.5$	-548.0 (22.1)	-546.7 (30.1)	-613.4 (32.2)	-600.3 (28.1)	80.18 % (0.2)
$\omega_\Sigma = 1.0$	-571.2 (24.1)	-575.0 (30.6)	-595.5 (32.0)	-578.4 (29.4)	81.41 % (0.2)
$\omega_\Sigma = 2.0$	-574.1 (23.9)	-578.3 (31.1)	-596.4 (30.3)	-579.8 (28.0)	81.53 % (0.2)
$\omega_\Sigma = 5.0$	-588.2 (23.3)	-592.2 (30.3)	-607.5 (26.7)	-593.0 (24.5)	81.55 % (0.2)
$\omega_\Sigma = 10.0$	-624.5 (21.8)	-628.3 (28.4)	-640.4 (22.7)	-628.3 (20.7)	81.56 % (0.2)
$\omega_\Sigma = 100.0$	-1186.1 (9.4)	-1187.8 (12.2)	-1191.1 (8.2)	-1186.9 (7.5)	81.70 % (0.2)
Bayesian					
$\omega_\Sigma = 0.01$	-1365.3 (205.0)	-1448.7 (237.9)	-2104.4 (116.8)	-2150.9 (142.7)	59.99 % (0.5)
$\omega_\Sigma = 0.1$	-1342.6 (435.0)	-1110.7 (326.7)	-1720.2 (297.0)	-1567.7 (204.1)	64.76 % (0.9)
$\omega_\Sigma = 0.2$	-1659.0 (428.1)	-1853.8 (491.7)	-1831.1 (364.9)	-1983.0 (408.8)	59.16 % (0.9)
$\omega_\Sigma = 0.5$	-2753.1 (691.8)	-2773.0 (575.9)	-2780.0 (673.3)	-2795.5 (551.8)	55.81 % (0.8)
$\omega_\Sigma = 1.0$	-3470.1 (502.6)	-3489.6 (566.4)	-3475.1 (504.2)	-3497.4 (558.9)	50.86 % (0.1)
$\omega_\Sigma = 2.0$	-3069.3 (473.7)	-2980.6 (495.7)	-3078.8 (471.9)	-2991.2 (497.6)	52.22 % (0.5)
$\omega_\Sigma = 5.0$	-2380.4 (307.5)	-2482.0 (347.6)	-2385.2 (308.5)	-2482.6 (348.3)	54.37 % (0.7)

TABLE II

PERFORMANCES IN THE TOY CLASSIFICATION PROBLEM II.

Algorithm	**Log-Likelihood**				
	Training		**Test**		**Accuracy**
	A	B	A	B	
Max. Likelihood	-490.4 (185.4)	-724.3 (102.8)	-307.3 (75.9)	-468.6 (60.1)	64.02 % (5.9)
Averaging					
Simple Averaging	867.1 (340.9)	54.4 (264.0)	-46.1 (66.2)	-228.9 (144.2)	74.16 % (3.1)
Subset Averaging	1148.5 (252.6)	460.9 (272.1)	-61.4 (64.2)	-316.3 (100.7)	73.13 % (4.3)
Bagging	2166.1 (289.0)	1409.2 (408.3)	54.3 (79.6)	-120.1 (132.8)	77.85 % (3.3)
Penalized Likelihood					
$\omega_\Sigma = 0.05$	-502.6 (12.3)	-713.2 (29.1)	-338.1 (31.4)	-468.8 (43.2)	67.15 % (3.1)
$\omega_\Sigma = 0.1$	-515.2 (13.7)	-722.3 (20.3)	-320.4 (23.0)	-469.1 (43.1)	67.06 % (4.6)
$\omega_\Sigma = 0.2$	-532.0 (13.2)	-732.0 (23.9)	-315.7 (19.7)	-463.1 (44.6)	66.12 % (4.1)
$\omega_\Sigma = 0.5$	-563.8 (15.3)	-758.5 (21.6)	-313.8 (18.6)	-453.8 (36.6)	62.06 % (4.0)
$\omega_\Sigma = 1.0$	-591.8 (10.4)	-789.6 (22.3)	-310.7 (14.4)	-450.7 (34.5)	61.68 % (3.0)
$\omega_\Sigma = 2.0$	-618.6 (10.8)	-826.2 (21.2)	-309.0 (11.4)	-443.8 (31.2)	59.95 % (2.4)
$\omega_\Sigma = 5.0$	-652.6 (9.1)	-868.0 (20.1)	-311.5 (10.6)	-448.0 (30.5)	57.43 % (3.1)
$\omega_\Sigma = 10.0$	-676.4 (9.9)	-897.6 (22.9)	-318.9 (8.2)	-452.2 (29.7)	57.8 % (2.9)
$\omega_\Sigma = 20.0$	-692.5 (8.4)	-930.5 (22.2)	-325.8 (7.1)	-458.8 (28.1)	58.74 % (2.6)
$\omega_\Sigma = 50.0$	-714.8 (6.3)	-963.9 (20.5)	-337.5 (6.7)	-471.5 (27.2)	58.08 % (3.1)
Bayesian					
$\omega_\Sigma = 0.02$	-927.6 (125.5)	-1312.7 (192.9)	-526.1 (51.5)	-705.5 (85.3)	62.1 % (9.6)
$\omega_\Sigma = 0.05$	-816.9 (85.5)	-1286.7 (133.2)	-474.2 (35.8)	-710.8 (77.5)	59.91 % (8.1)
$\omega_\Sigma = 0.1$	-757.6 (68.2)	-1160.2 (113.5)	-442.3 (24.9)	-639.7 (52.3)	66.78 % (6.9)
$\omega_\Sigma = 0.2$	-813.6 (60.7)	-1141.4 (100.2)	-446.1 (29.1)	-612.8 (45.8)	61.31 % (8.1)
$\omega_\Sigma = 0.5$	-894.4 (72.0)	-1190.7 (132.7)	-446.0 (27.8)	-603.2 (53.6)	59.72 % (5.8)
$\omega_\Sigma = 1.0$	-993.6 (87.2)	-1306.7 (216.1)	-472.7 (632.8)	-632.8 (49.4)	57.38 % (7.1)
$\omega_\Sigma = 2.0$	-1012.4 (90.8)	-1316.2 (145.1)	-472.9 (34.4)	-626.2 (54.5)	54.72 % (6.2)
$\omega_\Sigma = 5.0$	-944.5 (51.3)	-1322.7 (101.0)	-435.5 (22.0)	-613.1 (43.9)	54.49 % (8.2)
$\omega_\Sigma = 10.0$	-948.5 (70.5)	-1238.7 (82.6)	-432.7 (30.6)	-577.3 (36.0)	56.03 % (5.6)
$\omega_\Sigma = 20$	-894.6 (31.5)	-1211.9 (67.1)	-407.1 (13.9)	-565.4 (28.0)	55.28 % (6.8)
$\omega_\Sigma = 50.0$	-843.9 (17.3)	-1163.3 (40.9)	-385.1 (8.3)	-543.1 (24.3)	53.83 % (4.7)

TABLE III

PERFORMANCES IN THE LIVER DISORDER CLASSIFICATION PROBLEM.