The Group Latent Variable Approach to Probit Binary Classifications

Yun Seog Yeun and Nam-Joon Kim, Member, IEEE

Abstract—This paper considers the binary classification with the probit model under the expectation-maximization (EM) algorithm. Usually, in the Bayesian approach of the probit model, the latent variables are introduced to handle with the intractable problem. For each training sample, there is a corresponding latent variable. However, the EM algorithm requires matrix inversions which demand for the expensive computational cost when the number of training samples is large. To overcome this problem, we employ the group latent-variable approach where for each of the training samples there are corresponding multiple latent variables instead of just one. The major advantage of this approach, which is originated from Bayesian backfitting, is that there are no requirements for matrix inversions in the EM algorithm for the probit model. In this paper, to obtain sparse classifiers the Laplacian prior is employed and the method to control the degree of sparseness is presented. Although the sparse classifiers the Laplacian prior is employed and the method to control the degree of sparseness is presented. Although the sparse classifiers the Laplacian prior is employed and the method to control the degree of sparseness is presented. Although the sparse classifiers the Laplacian prior is employed and the method to control the degree of sparseness is presented. Although the sparse classifiers the Laplacian prior is employed and the method to control the degree of sparseness is presented.

Index Terms—Bayesian backfitting, binary classification, expectation–maximization (EM) algorithm, Laplacian prior, latent variable, probit model, sparseness.

I. INTRODUCTION

A. Motivation

In the problem of binary classifications we are given a training set \( D = \{ x_i, y_i \}_{i=1}^n \), where \( x_i = (x_{i,k})_{k=1}^d \in \mathbb{R}^d \) and \( y_i \in \{ +1, -1 \} \). With this training set, our task is to infer a function \( f(x) \) that accurately predicts the class of an unseen input vector. Usually the form of \( f(x) \) can be expressed as the combination of kernel functions

\[
 f(x) = b^T h(x)
\]

where \( b^T = (b_1)^n_{i=1} \) are parameters and \( h(x)^T = (h_i(x))^n_{i=1} \) are kernel functions. Note that the number of kernel functions in (1) is equal to that of training samples in kernel machines.

This paper considers the Gaussian kernel as shown in

\[
 h_i(x) = \exp \left( -\sum_{k=1}^d \theta_i^k (x_k - x_{i,k})^2 \right)
\]

where \( \theta_i = (\theta_{i,1}, \ldots, \theta_{i,d}) \) are kernel parameters. As can be observed in (2), in total we need to estimate \( n \times d \) different kernel parameters. This is the most general formulation, considering the fact that usually kernel machines use only one kernel parameter or \( d \) different kernel parameters.

In the probit model, the kernel classifier takes the following form:

\[
 P(y = 1|x) = \Phi(b^T h(x))
\]

where \( \Phi(z) \) is the normal cumulative distribution function

\[
 \Phi(z) = \int_{-\infty}^z N(x|0,1) = \frac{1}{\sqrt{2\pi}} \int_{-\infty}^z \exp \left( \frac{-x^2}{2} \right) dx.
\]

If we take the prior density of \( b, p(b) \), then the posterior density of \( b \) is given by

\[
 p(b|D) \propto p(b) \prod_{i=1}^n \Phi(y_i b^T h(x_i)).
\]

Note that \( 1 - \Phi(z) = \Phi(-z) \) and \( y_i \in \{ +1, -1 \} \). Unfortunately, the above posterior is largely intractable. To overcome this problem, Albert and Chib [1] introduced \( n \) independent hidden or latent variables \( z_1, \ldots, z_n \) in the following way:

\[
 p(z_i|b, D) \propto u(y_i z_i) N(z_i|b^T h(x_i), 1)
\]

where \( u() \) is the unit step function. According to (6), the conditional distribution of \( z_i \) is the truncated Gaussian with mean \( b^T h(x_i) \). That is, (6) is the left-truncated normal distribution, if \( y_i = 1 \) and the right-truncated one, if \( y_i = -1 \). With the latent variable \( z_i \), the posterior density of \( b \) is given by

\[
 p(b|z_i, D) \propto p(b) \prod_{i=1}^n \left[ u(y_i z_i) N(z_i|b^T h(x_i), 1) \right].
\]

Note that if we integrate out \( z_1, \ldots, z_n \), (7) becomes (5). As shown in [2] and [3], by applying the expectation–maximization (EM) algorithm to (7), the estimate of \( b \) can be readily obtained, but for each iteration the inversion of the \( n \times n \) matrix is required. Therefore, the computational cost is prohibitively expensive if the size of the training set is large, even if the kernel parameters \( \theta_i \) are not estimated. In addition, there might be a possible numerical instability due to matrix inversions.

In Bayesian backfitting [4]–[6], the problem of matrix inversions is circumvented by introducing a vector of latent variables \( z_i^T = (z_{i,1}, \ldots, z_{i,n}) \) associated with the input vector
\( x_i \), where \( z_{i,m} \) is the normal random variable with mean \( b_m h_m(x_i) \) and variance \( \tau_{2m} \). In this paper, we employ this group latent-variable approach to the probit model. Therefore, the conditional distribution of \( z_i \) can be considered as the following way:

\[
p(z_i | b, D) \propto u(y_i \sqrt{1^T z_i})N(z_i | b^T h(x_i), \Psi_z) \tag{8}
\]

where \((b^T h(x_i))^T = (b_1 h_1(x_i), \ldots, b_n h_n(x_i))\) and \( \Psi_z = \text{diag} (\tau_1, \ldots, \tau_m) \)

As will be revealed in the following section, it can be viewed as the special case of a multivariate skew-normal distribution [7]. If we integrate out \( z_i \), (8) becomes \( \Phi(y_i b^T h(x_i)/\sqrt{1^T \tau_z}) \), which is the probability of \( y_i, P(y_i | x_i) \). In addition, the posterior density of \( b \) can be given by

\[
p(b | z_i, D) \propto p(b) \prod_{i=1}^{n} \{u \left( y_i \sqrt{1^T z_i} \right) N(z_i | b^T h(x_i), \Psi_z) \}. \tag{9}
\]

As shown in the following section, the EM algorithm can be easily applied to (9) to obtain the estimate of \( b \) without any matrix inversions. Thus, the computational burden can be significantly reduced even if the size of learning samples is large. In addition, it is possible to estimate \( \theta_i \) with the reasonable computational cost.

We use a Laplacian prior on \( b \) to promote sparseness in estimating \( b \), and present the method of choosing associated hyperparameters controlling the degree of sparseness.

### B. Related Works

Our approach is related to other classifiers such as the support vector machine (SVM) [8], [9], the relevance vector machine (RVM) [10], [11], and the generalized LASSO [12] of which learning algorithms try to produce sparse solutions. However, [2] and [3] have much more significant influence on this paper because our paper follows the similar track of [2] and [3] by employing both the EM algorithm and the Laplacian prior. The major differences are that we adopt the group latent-variable approach to avoid the matrix inversion under the EM algorithm, and present the different treatment of hyperparameters which govern sparseness of solutions. In addition, needless to say, our work is inspired by Bayesian backfitting [4]--[6] where the group latent-variable approach is applied. However, this paper does not belong to Bayesian backfitting. If we follow the Bayesian backfitting formulation with the probit link function, we should directly handle with \( \Phi(1^T z_i) \), apparently which poses us a difficult problem.

There are other research works regarding the probit models which yield sparse solutions. For instance, to prompt sparse solutions the Laplacian prior on \( b \) is employed in the hierarchical way and furthermore the use of the Jeffreys’ prior is encouraged in [2]. At first it seems that the Jeffreys’ prior is attractive because it has no associated parameter, but in certain problems it leads too sparse solutions which are not beneficial at all [14], [22]. In [14], instead of the EM algorithm a variational Bayesian probit classification is presented with the Laplacian prior and a sparsity-inducing mechanism is discussed. In probabilistic classification vector machines (PCVM) a truncated student-\( t \) prior is used, and the probit link function is used with the EM algorithm [21]. More recently, a generalized Gaussian scale mixture prior is proposed for probit classifiers [22]. In addition to those mentioned, other works related to this paper can be found in [23]--[34].

### C. Organization of This Paper

Sections II and III discuss the distribution of \( z_i \) conditioned on \( y_i \), and present the EM algorithm for estimating \( b \) and \( \theta_i \). Section IV presents the method of deciding the value of hyperparameters related to the Laplacian prior. Section V gives an illustrative example. In Section VI, the experimental results of various benchmark data sets are compared with other methods such as RVM, SVM, and generalized LASSO. Finally, conclusion is offered in Section VII.

### II. PROBIT LINK AND THE GROUP OF LATENT VARIABLES

The EM algorithm requires the distribution of the vector of latent variables \( z_i \) conditioned on \( y_i \), which is shown in

\[
p(z_i | b, D) = u \left( y_i \sqrt{1^T z_i} \right) N(z_i | b^T h(x_i), \Psi_z) / C_z \tag{10}
\]

where \( C_z \) is the normalizing constant. We can view (10) as a multivariate skew-normal distribution [7], as described in the following.

Introduce a random variable \( u \) whose distribution is \( N(u | 0, \rho) \) and consider \( y_i \sqrt{1^T z_i} > u \). Then, the conditional density of \( z_i \) should be

\[
p(z_i | b, D) = N(z_i | b^T h(x_i), \Psi_z) \int_{-\infty}^{y_i \sqrt{1^T z_i}} N(u | 0, \rho) du / C_{sn}
\]

\[
= N(z_i | b^T h(x_i), \Psi_z) \Phi(y_i \sqrt{1^T z_i} / \sqrt{\rho}) / C_{sn} \tag{11}
\]

where \( C_{sn} \) is the normalizing constant. This is a well-known skew-normal distribution. Since \(-u + y_i \sqrt{1^T z_i} \) is the normal distribution with mean \( y_i b^T h(x_i) \) and variance \( y_i^2 \sqrt{1^T \tau_z} \) where \( \tau_z = (\tau_1, \ldots, \tau_m) \) (note that \( y_i^2 = 1 \))

\[
C_{sn} = P \left( -u + y_i \sqrt{1^T z_i} > 0 \right) = \Phi \left( y_i b^T h(x_i) / \sqrt{\rho + 1^T \tau_z} \right) \tag{12}
\]

If \( \rho \to 0 \), \( C_{sn} = \Phi(y_i b^T h(x_i) / \sqrt{1^T \tau_z}) \), and \( \Phi(y_i \sqrt{1^T z_i}) = u(y_i \sqrt{1^T z_i}) \). Now it is clear that (10) is a skew-normal distribution. Furthermore we can see that

\[
P(y_i | b) = C_z = \Phi \left( y_i b^T h(x_i) / \sqrt{1^T \tau_z} \right) \tag{13}
\]

Fig. 1 shows an example of a bivariate skew-normal distribution with the unit step function. In this figure, the density function of \((z_{i,1}, z_{i,2})\) is given by

\[
p \left( z_{i,1}, z_{i,2} \right) = u \left( y_i \left( z_{i,1} + z_{i,2} \right) \right)
\times N \left( \left( \begin{array}{c} z_{i,1} \\ z_{i,2} \end{array} \right) \bigg| \left( \begin{array}{c} 0 \\ 1 \end{array} \right), \left( \begin{array}{cc} 0 & 0 \\ 0 & 2 \end{array} \right) \right) / 0.718 \tag{14}
\]
where \( \gamma_1 = 1 \). Note that the normalizing constant 0.718 is correspondent to \( P(\gamma_1 = 1) \).

The expectation of \( z_{i:m} \), \( \langle z_{i:m} \rangle \) and its covariance \( \langle z_{i:m} z_{i:n} \rangle \) can be obtained by the moment generating function

\[
\langle z_{i:m} \rangle = b_m h_m(x_i) + \frac{\gamma_1 \tau_{z_m}}{\sqrt{1^T \tau_z \gamma_1}} \phi\left(\frac{\gamma_1 b_m h_m(x_i)}{\sqrt{1^T \tau_z}}\right)
\]

\[
\langle z_{i:m} z_{i:n} \rangle = \delta_{mn} \frac{\gamma_1 \tau_{z_m} \tau_{z_n}}{\sqrt{1^T \tau_z \gamma_1}} \phi\left(\frac{\gamma_1 b_m h_m(x_i)}{\sqrt{1^T \tau_z}}\right)
\]

(15)

where \( \delta_{mn} \) is the delta function such that \( \delta_{mn} = 1 \) if \( m = n \), and \( \delta_{mn} = 0 \) if \( m \neq n \). See Appendix A for the details.

### III. EM ALGORITHM

To induce the sparsity of the estimated parameters \( b \), a Laplacian prior can employed for each \( b_i \): \( p(b_i|\gamma_i) = \gamma_i/2\exp(-\gamma_i|b_i|) \). However, there are numerical difficulties due to the discontinuity at the origin. This problem can be overcome by introducing a Gaussian prior \( p(b_i|\tau_i) = N(b_i|0, \tau_i) \) and an exponential distribution \( p(\tau_i|\gamma_i) = \gamma_i/2\exp(-\gamma_i/2\tau_i) \). If we integrate out \( \tau_i \) from \( p(b_i|\tau_i) \times p(\tau_i|\gamma_i) \), it becomes \( p(b_i|\gamma_i) = \sqrt{\gamma_i}/2\exp(-\sqrt{\gamma_i}|b_i|) \) which is a Laplacian prior [2].

We want to estimate the kernel parameters \( \theta_i \) using an optimization technique such as a conjugate gradient method. However, it almost surely leads to the overfitting problems. Therefore, the prior on \( \theta_{i:m} \) is introduced in the similar way of [3], where the sparsity of the kernel parameters is pursued to remove irrelevant features. Since our major interest is to avoid overfitting by imposing the certain condition on \( \theta_{i:m} \), we will not further mention the issue of feature selections in this paper. The prior on \( \theta_{i:m} \) is adopted in the following manner:

\[
p(\theta_{i:m}|\sigma_{i:m}) \propto N(\theta_{i:m}|0, \sigma_{i:m}) \rho(\sigma_{i:m})
\]

(17)

where \( \rho(\sigma_{i:m}) \) is the Jeffreys’ or noninformative prior, \( 1/\sigma \).

The reason why this prior is adopted is to avoid involving additional hyperparameters. If we integrate out \( \sigma_{i:m} \) from \( p(\theta_{i:m}|\sigma_{i:m}) \), it becomes \( p(\theta_{i:m}) \propto 1/\theta_{i:m} \), which is also a noninformative distribution.

In this paper, to simplify the problem, the value of \( \tau_{z_m} \) is pre-determined in such a way that \( \tau_{z_1} = \ldots = \tau_{z_m} = 1/n \) for ensuring \( 1^T \tau_z = 1 \).

Now we want to estimate \( b, \theta \) with the EM algorithm. To do this the posterior distribution of \( b \) and \( \theta \) shown in (18) is required

\[
p(b, \theta|D, z, \tau, \sigma) \propto \prod_{i=1}^{n} u\left(\gamma_i 1^T z_i\right) \times \prod_{m=1}^{n} (2\pi)^{-2} (\tau_{z_m})^{-\frac{1}{2}}
\]

\[
\exp\left\{-\frac{1}{2\tau_{z_m}} \left(\sum_{i=1}^{n} z_{i:m}^2 - 2b_m \sum_{i=1}^{n} h_m(x_i) z_{i:m} + b_m^2 \sum_{i=1}^{n} h_m^2(x_i)\right)\right\}
\]

\[
\times \prod_{m=1}^{n} (2\pi \tau_{m})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2\sigma_m} \sum_{k=1}^{d} \gamma_m^2 \sum_{m=1}^{n} \gamma_m^2 \exp\left\{-\frac{\gamma_m^2}{2} \tau_{m}\right\}
\]

\[
\times \prod_{m=1}^{n} (2\pi \sigma_{m:k})^{-\frac{1}{2}} \exp\left\{-\frac{1}{2\sigma_{m:k}} \sum_{k=1}^{d} \gamma_{m:k}^2 \sum_{m=1}^{n} \gamma_{m:k}^2 \exp\left\{-\frac{\gamma_{m:k}^2}{2} \sigma_{m:k}\right\}
\]

(18)

Under the EM algorithm, we can find the maximum a posterior estimates

\[
(\hat{b}, \hat{\theta}) = \arg \max_{b, \theta}(p(b, \theta|D, z, \tau, \sigma)).
\]

The EM algorithm requires the E-step and the M-step.

#### A. E-Step

In the E-step, to remove latent variables \( (z, \tau, \sigma) \) from \( \log(p(b, \theta|D, z, \tau, \sigma)) \), the expected value of the log-posterior \( <\log(p(b, \theta|D, z, \tau, \sigma))> \) is computed with respect to the density function of each latent variable, which is conditioned on \( D \) and the current estimate of \( (\hat{b}^{(t)}, \hat{\theta}^{(t)}) \). This expectation of the log-posterior, called the \( Q \) function, is given by

\[
Q_m\left(b_m, \theta_m, \hat{b}_m^{(t)}, \hat{\theta}_m^{(t)}\right) = -\frac{1}{2\tau_{z_m}} \left( -2b_m \sum_{i=1}^{n} h_m(x_i) (z_{i:m})^{(t)} + b_m^2 \sum_{i=1}^{n} h_m^2(x_i) \right)
\]

\[
-\frac{1}{2} \left( <\tau_{z_m}^{-1}>^{(t)} \right) b_m^2 - \sum_{k=1}^{d} \frac{1}{2} \left( <\sigma_{m:k}^{-1}>^{(t)} \right) \sigma_{m:k}^2 (m = 1, \ldots n).
\]

(19)
The expectation of \( z_{im} \) in (19) can be computed by (15) with \( \tau^{-1} \) and \( \sigma^{-1} \) given by
\[
\langle \tau^{-1} \rangle_{(l)} = \frac{\int_0^\infty b_m(x_i)\exp(-\frac{1}{2}\tau)N(b_m|0, \tau)d\tau}{\int_0^\infty N(b_m|0, \tau)d\tau} = \sqrt{\gamma_m}
\]
(20)
and
\[
\langle \sigma^{-1} \rangle_{(l)} = \frac{1}{\hat{\theta}_{mk}^2}
\]
(21)

\[\text{B. M-Step}\]

In the M-step, the new estimates are sought in such a way that
\[
(\hat{b}_{m(k)}^{(l+1)}, \hat{\theta}_{mk}^{(l+1)}) = \arg \max_{(b_m, \theta)} Q_m(b_m, \theta; \hat{b}_{m(k)}^{(l)}, \hat{\theta}_{mk}^{(l)})
\]
By the derivatives of \( Q_m \) with respect to \( b_m, \theta \), the update rules can be found as shown in the following:
\[\hat{b}_{m(k)}^{(l+1)} = \frac{\sum_{i=1}^n h_m(x_i)}{\sum_{i=1}^n h_m(x_i) + \tau_m}\]
(22)
The partial derivatives of \( Q_m \) with respect to \( \theta_{mk} \) can be given by
\[
\frac{\partial Q_m}{\partial \theta_{mk}} = \langle \sigma_{mk}^{-1} \rangle_{(l)} + \frac{1}{\tau_m} \hat{b}_{m(k)}^{(l)} \sum_{i=1}^n \frac{\partial h_m(x_i)}{\partial \theta_{mk}}
\]
\[
-\langle \hat{b}_{m(k)}^{(l)} \rangle_{(l)}^2 \sum_{i=1}^n h_m(x_i) \frac{\partial h_m(x_i)}{\partial \theta_{mk}}
\]
(23)
where \( \frac{\partial h_m(x_i)}{\partial \theta_{mk}} = -(x_{i,ik} - x_{mk})^2 h_m(x_i) \).

The standard gradient based method can be used to estimate \( \theta_m \) by maximizing \( Q_m \). Note that the \( Q \) function is completely decomposed into \( n \) independent \( Q_m \) and there is no involvement of any matrix inversions in the EM algorithm.

\[\text{IV. HYPERPARAMETERS}\]

Now, we are in the position of determining the value of the hyperparameter \( \gamma_m \) controlling the sparsity of \( b_m \). Usually, \( \gamma_m \) is chosen by conducting computational extensive cross validations if \( \gamma = \gamma_1 = \cdots = \gamma_n \) [2]. In this paper, we have a different approach to the estimation of \( \gamma_m \).

With the given estimates \( \theta^{(l)} \) the posterior can be rewritten by
\[
p(b_m|\gamma) = \gamma_m^{1/2} \exp\left(-(A_m \gamma_m^{1/2} - B_m \gamma_m)\right) (m = 1, \ldots, n)
\]
(25)
where both \( A_m \) and \( B_m \) are constants which are described in Appendix B. By \( d\gamma_m/d\gamma_m = 0 \), without any search methods or iterations, we can obtain the closed form solution \( \hat{\gamma}_m \) at which the peak of \( l(\gamma_m) \) is located. Usually \( \hat{\gamma}_m \) is a large numeric value, \( b_m \) shows the tendency of becoming zero. The magnitude of \( \hat{\gamma}_1, \ldots, \hat{\gamma}_n \) and its distributed pattern could determine the degree of sparseness. We want to more actively control the sparsity of \( b_m \) by adjusting the magnitude of the estimated hyperparameters. To do this, two parameters \( (\alpha, \beta) \) are introduced into (25)
\[
l(\gamma_m) = \gamma_m^{\alpha} \exp\left(-A_m \gamma_m^{1/2} - (\beta + B_m) \gamma_m\right)
\]
(26)
where \( \alpha > 0 \) and \( \beta > 0 \).

Actually, this is equivalent to impose a prior distribution on \( \gamma_m \), \( p(\gamma_m) \propto \gamma_m^{-\alpha} \exp(-\beta \gamma_m) \), that is similar to a gamma distribution. The closed form solution \( \hat{\gamma}_m \), which is the optimum point of \( l(\gamma_m) \), is given by
\[
\hat{\gamma}_m = \frac{A_m^2 + 8\alpha(\beta + B_m) - A_m \sqrt{A_m^2 + 16\alpha(\beta + B_m)}}{8(\beta + B_m)^2}
\]
(27)
Two parameters \( (\alpha, \beta) \) are determined by the following way in this paper:

1. The value for \( \beta(> 0) \) is chosen as the minimum number ensuring that \( \beta + B_m > 0 \) for all \( m \), which implies that \( l(\gamma_m) \) always has a peak at \( \hat{\gamma}_m \), since \( A_m^2 + 16\alpha(\beta + B_m) > 0 \) and \( \beta + B_m \neq 0 \).

2. To decide \( \alpha \), we need the reference value for \( \gamma_m \), which will be denoted by \( R_m \). As in (14), \( \exp(-\gamma_m) \) is used as a prior for \( \gamma_m \), \( p(\gamma_m) \). With this \( p(\gamma_m) \) we can obtain the distribution of \( b_m \), as shown in the following:
\[
p(b_m) = \int_0^\infty \int_0^\infty p(b_m|\tau_m)p(\tau_m|\gamma_m)p(\gamma_m)d\tau_md\gamma_m = \frac{1}{8} \left(-2|b_m| + \sqrt{\pi}(2 + b_m^2) \exp \left( \frac{b_m^2}{4} \right) \right)
\]
(28)
where \( \text{erf}(.) \) is the error function defined by \( \text{erf}(x) = \frac{2}{\sqrt{\pi}} \int_0^x \exp(-t^2)dt \).

In Fig. 2, the solid line is the plot of (28), and the dotted line is the Laplace distribution
\[
p(b_m|\gamma_m) = \frac{\sqrt{\gamma_m}}{2} \exp(-\sqrt{\gamma_m}|b_m|) \text{ with } \gamma_m = 1.
\]
Algorithm 1: Overall Algorithm for the Probit Model

1: For \( m = 1, \ldots, n \), initialize \( \tilde{\theta}^{(0)}_m, \tilde{\gamma}^{(0)}_m \) with the appropriate value, \( \tau_m = \frac{1}{n} \), and \( \bar{r}_m = 1 \).

2: Repeat until the maximum iteration number \( I_1 \) is reached or there is no significant improvement in \( \hat{B}^{(0)} \):

3: Estimate \( \{< x_{m,i} > \}_{i=1}^{n}, \{< \tau_{m,i} > \}_{i=1}^{n}, \{< \sigma_{m,i}^2 > \}_{i=1}^{n} \) with current estimates \( \{\hat{\gamma}_1, \ldots, \hat{\gamma}_n\} \) (See (15) and (20)-(22)).

4: Estimate \( \tilde{\theta}^{(k)}_m \) \( (m = 1, \ldots, n) \) using the standard gradient-based method (See (23)).

A few iterations (3–5) may be sufficient.

5: Estimate new \( \{\hat{\gamma}_1, \ldots, \hat{\gamma}_n\} \) (See (27)) with given \( \kappa \) after every \( I_2 \) iterations are completed, and update current \( \{\hat{\gamma}_1, \ldots, \hat{\gamma}_n\} \) with newly estimated ones.

As can be observed in this figure, two distributions show somewhat similar behaviors. Therefore, we choose one as the reference value for \( \gamma_m \) (actually, when \( \gamma_m = 0.7 \sim 0.8 \), two distributions look much more alike). Of course, it is free to choose another value, but one that is too large or too small might not be beneficial.

3) To determine the degree of sparseness, we need to specify \( N_{R} \), that is the number of hyperparameters, which should be less than the reference value of the hyperparameter, \( R_{\gamma} \). Let denote \( N_{R}^\gamma \) as the actual number of current hyperparameters, which is less than \( R_{\gamma} \) among \( \{\hat{\gamma}_1, \ldots, \hat{\gamma}_n\} \). Our task is to adjust \( \alpha \) by increasing or decreasing until \( N_{R}^\gamma \leq N_{R} \) within the given tolerance. In this way, the value of all hyperparameters are altered with ensuring the magnitude of about \( N_{R} \) different hyperparameters is less than \( R_{\gamma} \).

Needless to say, it does not imply that the number of nonzero \( b_m \)'s is always \( N_{R} \), but there is a certain possibility for that. In this paper, \( N_{R} \) is given by the \( \kappa \) percentage of learning sample size (\( n \)), and the default value is 10% of \( n \) (\( \kappa \) is 10). The disadvantages of this approach are in that it is not a full plugin method and does not consider the generalization performance in estimating the hyperparameters. On the other hand, sparseness could be controlled by just specifying \( \kappa \). The overall algorithm is summarized by Algorithm 1.

### Table I

<table>
<thead>
<tr>
<th>SVM</th>
<th>Probit Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>Errors[%]</td>
<td>Vectors</td>
</tr>
<tr>
<td>10</td>
<td>0.5</td>
</tr>
<tr>
<td>9.8</td>
<td>9.0</td>
</tr>
<tr>
<td>8.7</td>
<td>8.7</td>
</tr>
<tr>
<td>5</td>
<td>15</td>
</tr>
<tr>
<td>4</td>
<td>5</td>
</tr>
</tbody>
</table>

### Table II

<table>
<thead>
<tr>
<th>Nonzero ( b_m )</th>
<th>Kernel Center: ((x_{m,1}, x_{m,2}))</th>
<th>Kernel Parameter: ((\theta_{m,1}, \theta_{m,2}))</th>
</tr>
</thead>
<tbody>
<tr>
<td>-1.70765</td>
<td>-0.21886</td>
<td>-1.83450</td>
</tr>
<tr>
<td>-3.64285</td>
<td>-2.41938</td>
<td>-0.13449</td>
</tr>
<tr>
<td>2.51992</td>
<td>-0.49869</td>
<td>0.97286</td>
</tr>
<tr>
<td>2.21072</td>
<td>-0.09820</td>
<td>1.31834</td>
</tr>
</tbody>
</table>

### V. Illustrative Example

The results of the probit model are given and discussed with the Ripley’s synthetic data set.\(^1\) There are predefined 250 training samples and 1000 test samples in this problem. All inputs are normalized into zero mean and unit variance. The maximum number of EM cycles (\( I_1 \)) is 50000 and after every 500 EM cycles (\( I_2 \)) hyperparameters \( \{\hat{\gamma}_1, \ldots, \hat{\gamma}_n\} \) are updated. In Table I, the results of the probit models are presented together with those of SVM where for reducing the number of support vectors smoothed separable case approximation (SSCA) is used \([20]\).

Only the best result of SVM giving the minimum prediction error from \([20]\) is shown in this table. Overall the probit models are superior to SVM in the prediction ability on this problem. In addition, the number of nonzero vectors is four in the case of \( \kappa = 0.5 \) while SVM with SSCA gives five. When \( \kappa = 10 \), the number of the nonzero vectors increases to 15 and the prediction capability is slightly deteriorated.

Table II shows the nonzero \( b_m \)'s and kernel parameters of the discriminant function when \( \kappa = 0.5 \). As can be seen in this table, the remarkable feature of the discriminant function is that many kernel parameters become exactly zero due to imposing the noninformative prior on each kernel parameter.

Fig. 3 shows two decision boundaries in the case of \( \kappa = 0.5 \) and \( \kappa = 10 \), respectively. It seems that there are no significant differences in two decision boundaries except for the right part where the behavior of the first one looks as a nearly horizontal line compared with the second which can effects on the prediction ability. In addition, like RVM \([10]\), the nonzero vectors are not located near the boundary region.

---

\(^1\)Available at www.stats.ox.ac.uk/pub/PRNN.
In the case where $\kappa = 0.5$, there are four nonzero vectors, and Fig. 4(a) shows the values of four hyperparameters ($\hat{\gamma}_m$) associated with these nonzero vectors. Since hyperparameters are updated at every 500 EM cycles, the value of the hyperparameters is changed in a stepwise manner. As expected, the hyperparameter approaches either near one or below one. On the other hand, as shown in Fig. 4(b), the value of other hyperparameters approaches into a large numeric value during EM cycles. Needless to say, that means the value of $b_m$ attached to such a hyperparameter eventually becomes zero. Finally the number of nonzero vectors during EM cycles is plotted in Fig. 5. After 5000 EM cycles more than 200 $b_m$s are zero, which agrees with the plot of the hyperparameters shown in Fig. 4(b) where after 5000 EM cycles a lot of hyperparameters become the large numeric value.

VI. EXPERIMENTAL RESULTS

In this section, we compared the proposed probit model with SVM [10], [21], RVM [10], [21], and LOGREG-LASSO [12].

We used several UCI benchmark data sets used by Rätsch et al. in [15]. In Table III, the number of training samples ($n$) and the number of input variables ($d$) are shown for each data set. In addition to the default value of $\kappa$ (10), the results for $\kappa = 1$ are presented. Also in addition, the published results of SVM and RVM from [10] are shown for the purpose of the comparisons. In the similar way presented in [10], the results of both the probit model and RVM are normalized by those of SVM.

Overall it seems that the average prediction errors of the probit model are similar to those of RVM. In the case where $\kappa = 10$, except for titanic data set, the solution of RVM is sparser than those of the probit model. However when $\kappa = 1$,
in term of sparseness the probit model is better than RVM, while there is no significant difference in prediction errors.

As shown in Table IV, we compared the probit model with the LOGREG version of LASSO. The results of LOGREG given in [12] are shown in this table. The sparseness controlling parameter $\kappa$ is 10 in the probit model. Overall, regarding the average prediction errors, LOGREG is slightly better, but it seems that there is no statistical significance. In LOGREG, the width of kernel and the regularization parameter are selected by minimizing the averaged test error on five randomly chosen splits, while in the probit model, there are no such efforts improving the generalization performance by determining the optimal value for $\kappa$. In addition, note that the solutions of LOGRES are sparser than those of the probit model with $\kappa = 10$.

Table V shows the published results of Bayesian trigonometric support vector classifier (BT SVC) [16], RVM [21], and probabilistic classification vector machines (PCVM) [21] over Image and Splice data sets, which include relatively large training samples with many input variables. This table contains the result of the probit model obtained with $\kappa = 1$.

The results of both BTSVC and the probit model are over 20 training/test split while those of PCVM and RVM are over 100 splits (in addition to original 20 splits which are available in the benchmark repository, extra 80 splits are randomly generated) [21].

In BTSVC, the automatic relevance determination [17], [18] is adopted. That is, the $d$ different kernel parameters $\theta_1, \ldots, \theta_d$ are estimated by the gradient-based optimization method. In our approach, total $n \times d$ different kernel parameters
\( \theta_i = \{ \theta_{i,1}, \ldots, \theta_{i,d} \}_{i=1}^n \) are estimated by the gradient-based method. Therefore, it is the time-consuming process for the high-dimensional data set such as Splice data. It takes a few hours to complete 50000 iteration of the EM algorithm for one Splice data set with the Intel i7 950 PC. However, the probit model with \( \kappa = 1 \) gives smaller prediction errors compared with RVM, BTSVC, and PCVM. In addition, the average number of nonzero vectors is less than 10\% of the training sample size.

In the traditional EM algorithm [2] with the given kernel parameter where matrix inversions are required to estimate \( b \), it takes 1040 and 451 s to complete 10 EM cycles for the Image and Splice data set, respectively. On the other hand, without the estimation of kernel parameters the proposed method requires only 3 and 4 s. This clearly shows the advantage of our approach in the EM algorithm. Such a fast method allows estimating 23 400 and 60 000 kernel parameters in the Image and Splice problem, respectively under the EM algorithm. Such a fast method allows for quick investigations of hyperparameters that govern the sparsity of solutions inherited by the EM algorithm; the slow convergence rate and the sensitiveness to the initial value of the parameters in searching for optimum solutions.

### APPENDIX A

The expectation and the covariance of \( z_i \) can be obtained using the moment generating function

\[
M(t) = \int_{-\infty}^{\infty} \exp(t^T z_i) p(z_i|b, D) \, dz_i
\]

\[
= C \int_{-\infty}^{\infty} (2\pi)^{-\frac{n}{2}} |\Psi_z|^{-\frac{d}{2}} \times \exp\left\{ -\frac{1}{2}(z_i - (b^T h(x_i) + \Psi_z t))^T \Psi_z^{-1} (z_i - (b^T h(x_i) + \Psi_z t)) \right\} \times \frac{1}{2} t^T \Psi_z t \Phi(\sqrt{\rho} z_i) \, dz_i
\]

where \( t = (t_1, \ldots, t_d) \), \( C^{-1} = \Phi(y_i \sqrt{\rho} h(x_i))/\sqrt{\rho + t^T \Psi_z t} \). Note that if \( \rho \to 0 \), \( \Phi(y_i \sqrt{\rho} h(x_i))/\sqrt{\rho + t^T \Psi_z t} \to u(y_i 1^T z_i) \).

Let \( r_i = \Psi_z^{-\frac{1}{2}} (z_i - (b^T h(x_i) + \Psi_z t)) \), and then change the variables \( z_i \) with \( r_i \). We can write

\[
M(t) = C \exp \left( (b^T h(x_i))^T t + \frac{1}{2} t^T \Psi_z t \right) \times \int_{-\infty}^{\infty} \Phi\left( \frac{y_i \sqrt{\rho}}{\sqrt{\rho}} (b^T h(x_i) + \tau_z^T t) + \frac{y_i}{\sqrt{\rho}} \Psi_z^T r_i \right) N(r_i|0, 1) \, dr_i
\]

\[
= C \exp \left( (b^T h(x_i))^T t + \frac{1}{2} t^T \Psi_z t \right) \Phi\left( \frac{y_i (b^T h(x_i) + \tau_z^T t)}{\sqrt{\rho} + y_i^2 \tau_z^T \Psi_z} \right).
\]

We use the following lemma [19] to obtain \( \langle \Phi(\cdot) \rangle \) with respect to \( N(r_i|0, 1) \) in (A.2).
Lemma 1: Let \( U \sim N(0, \Omega) \). Then, for any scalar \( v \) and \( w \in \mathbb{R}^n \), we have

\[
< \Phi \left( v + w^T U \right) > = \int_{-\infty}^{\infty} \Phi \left( v + w^T U \right) N(U|0, \Omega) \, dU
= \Phi \left( \frac{v}{\sqrt{1 + w^T \Omega w}} \right).
\]

If \( \rho \to 0 \), finally we obtain the required moment generating function

\[
M(t) = \exp((b^T h(x_i))^T t + \frac{1}{2} t^T \Psi t) \\
\times \Phi \left( \frac{y_i(b^T h(x_i) + \tau_z t)}{\sqrt{1^T \tau_z}} \right) \Phi^{-1} \left( \frac{y_i b^T h(x_i)}{\sqrt{1^T \tau_z}} \right). \tag{A.3}
\]

From (A.3) we can obtain the expectation

\[
< z_{i,m} > = \frac{\partial M(t)}{\partial l_m} \bigg|_{t=0}
= b_m h_m(x_i) + \frac{y_i \tau_m}{\sqrt{1^T \tau_z}} N \left( b^T h(x_i) / \sqrt{1^T \tau_z} \right) \Phi \left( y_i b^T h(x_i) / \sqrt{1^T \tau_z} \right) \tag{A.4}
\]

and the covariance

\[
< z_{i,m} z_{i,m} > = \frac{\partial^2 M(t)}{\partial l_i \partial l_m} \bigg|_{t=0}
= \tau_z \delta_{i,m} - \frac{y_i b^T h(x_i) \tau_z}{\sqrt{1^T \tau_z}} N \left( b^T h(x_i) / \sqrt{1^T \tau_z} \right) \Phi \left( y_i b^T h(x_i) / \sqrt{1^T \tau_z} \right)
- \tau_z \frac{N^2 b^T h(x_i) / \sqrt{1^T \tau_z}}{\sqrt{1^T \tau_z}} \Phi^2 \left( y_i b^T h(x_i) / \sqrt{1^T \tau_z} \right). \tag{A.5}
\]

**APPENDIX B**

With given \( \theta^{(t)} \), if we compute the expected value of \( \log(p(z_i|b, \tau, \gamma, D)) \) [see (24)] with respect to \( p(z_i|b, D) \), collect the terms only related to \( b, \tau, \) and \( \gamma \), and take the exponent of them, then we have

\[
L_m = \gamma_m \frac{\tau_m}{2} \exp \left\{ - \frac{\gamma_m \tau_m}{2} \right\} \exp \left( - \frac{b_m^2}{2 \tau_m} \right)
\times \exp \left\{ - \frac{1}{2 \tau_m} \left( - 2 b_m \sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i) \\
+ b_m^2 \sum_{i=1}^{n} h_m^2(x_i) \right) \right\}. \tag{B.1}
\]

If we integrate out both \( b_m \) and \( \tau_m \) from \( L_m \) then

\[
\int_{0}^{\infty} \int_{0}^{\infty} L_m d \tau_m db_m \propto \sqrt{\tau_m}
\times \exp \left( - \frac{1}{2 \tau_m} \left( - 2 b_m \sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i) \\
+ b_m^2 \sum_{i=1}^{n} h_m^2(x_i) \right) \right)
\]

\[
g(\gamma_m) \tag{B.2}
\]

where

\[
g(\gamma_m) = \exp \left( \frac{2 \sqrt{\tau_m} \sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i)}{\sum_{i=1}^{n} h_m^2(x_i)} \right)
\times \left\{ \begin{array}{c}
1 - \text{erf} \left( \frac{\sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i) + \tau_m \sqrt{\tau_m}}{2 \tau_m \sum_{i=1}^{n} h_m^2(x_i)} \right) \\
1 + \text{erf} \left( \frac{\sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i) - \tau_m \sqrt{\tau_m}}{2 \tau_m \sum_{i=1}^{n} h_m^2(x_i)} \right)
\end{array} \right\}.
\]

Typically \( g(\gamma_m) \) shows a peak at \( \gamma_m = 0 \), and rapidly decreases as \( \gamma_m \) increases. Therefore, \( g(\gamma_m) \) can be approximated in the similar manner of a Laplace approximation. From the Taylor expansions of \( \log(g(\gamma_m)) \) at \( \gamma_m = 0 \), choosing only two terms, which is in the form of \( \gamma_m \) and \( \gamma_m \) taking the exponent and then we have

\[
g(\gamma_m) \equiv \exp \left( - (\gamma_m) \gamma_m / 2 - (\gamma_m) \right). \tag{B.3}
\]

Using (B.3), finally (B.2) can be approximated as shown in the following:

\[
l(\gamma_m) = \gamma_m \frac{1}{2} \exp \left( - A_m \gamma_m / 2 - B_m \gamma_m \right). \tag{B.4}
\]

For the description of \( A_m \) and \( B_m \), let us introduce

\[
c_m = \frac{\tau_m}{2}, \quad d_m = \frac{\sum_{i=1}^{n} h_m^2(x_i)}{\sum_{i=1}^{n} h_m^2(x_i)},
\]

and

\[
e_m = \frac{\sum_{i=1}^{n} < z_{i,m} >^{(t)} h_m(x_i)}{\sqrt{2 \tau_m \sum_{i=1}^{n} h_m^2(x_i)}}.
\]

Then

\[
A_m = \sqrt{\frac{2 c_m}{d_m}} \exp \left( - e_m^2 \right) + d_m \text{erf} \left( e_m \right). \tag{B.5}
\]
and
\[
B_m = -\frac{d_m^2}{2} \frac{e_m}{\pi} + \frac{c_m}{\pi} \exp\left(-2e_m^2\right) + \sqrt{\frac{2c_m}{\pi}}d_m \exp\left(-e_m^2\right)\text{erf}(e_m) + \frac{d_m^2}{2} \text{erf}^2(e_m).
\]  

(B.6)

REFERENCES


