Expectation Propagation of Gaussian Process Classification and Its Application to Gene Expression Analysis

Mingyue Tan
Department of Computer Science
University of British Columbia
mtan@cs.ubc.ca

Abstract

Expectation Propagation (EP) is an approximate Bayesian inference technique which has been applied to Gaussian Process Classification (GPC) [4]. In this paper, we investigate four different likelihood functions of GPC, and present EP algorithms for each of these four models. We compare the performances of these models on synthetic data in circular shape. Comparative study is performed on EP-GPC with SVM and Laplace-GPC. Experimental results show EP-GPC outperforms the other two kernel methods on high dimensional gene expression data. A feature selection technique, Automatic Relevance Determination (ARD), is applied to find the relevance of genes. Experiments show the effectiveness of ARD for all classification models.

1 Introduction

Classification with kernel machines have recently received much attention from the machine learning community. Some popular kernel classification algorithms include Support Vector Machine (SVM) [8], Bayes Point Machine (BPM), and GPC [5]. In this paper, we focus on GPC which is a Bayesian kernel classifier derived from Gaussian process priors over functions. Further more, we focus on he binary classification,i.e. discrimination between classes labeled as -1/+1.

GPCs can be represented as graphical models which have random variables for inputs, latent variables for function values, and class labels. Class labels are completely determined by the latent function values. Several noise model can be used to model the likelihood of class label given the latent value, such as probit function. Automatic Relevance Determination (ARD) parameters can be directly embedded into the covariance function, which can be considered a kernel that simplifies the problem in high dimensional space [5].

Since only class labels are observed, we need to integrate over both hyperparameters and latent values of these functions at the data points. Many approximation techniques have been used to approximate the integrals. For example, Williams used a

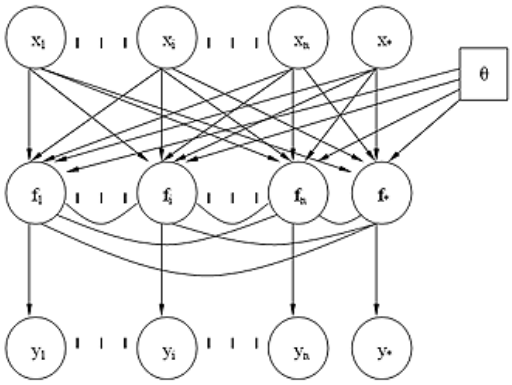


Figure 1: Graphical model for GPCs with n training data points and one test data point. [3]

Laplace approximation to integrate over the latent values and Hybrid Monte Carlo (HMC) to integrate over the hyperparamters [9]. Neal used HMC to integrate over both latent values and hyperparameters [5]. However, Monte Carlo methods are sometimes expensive to use in practice. A successful approach using Expectation Propagation (EP) is introduced by Thomas Minka in [4].

The contributions of this paper include: (1)Investigation of four different likelihood functions, namely step function, probit function, probit function with bias, probit function with Gaussian noise, for GPC and derive EP update rules for probit function with and without Gaussian noise. (2)Implementation of EP-GPC using probit function and probit function with bias. (3)Comparative study on various likelihood functions in the literature of EP for GPC (4) Comparative study of EP with other approximate inference techniques, such as Laplace approximation, as applied to GPC.

The rest of the paper is organized as follows. Section 2 introduces Gaussian process classification. In Section 3, we discuss EP with variational methods for hyperparameter inference. Section 4 presents EM-EP algorithm with four likelihood functions. The experimental results on both synthetic data and real gene expression data are reported in Section 5. We conclude in Section 6.

2 Gaussian Process Classifiers

Consider a data set D of data points x_i with binary class labels $y_i \in \{-1, 1\}$, $D = \{(x_i, y_i)|i = 1, 2, ..., n\}$, $X = \{x_i|i = 1, 2, ...n\}$, $Y = \{y_i|i = 1, 2, ..., n\}$. Given this training data set, we wish to predict the class label for a new data point $\mathbf{x}_*$ by computing the class probability $p(y_*|x_*, D)$.

The main idea of Gaussian process classifier is to assume that the class label y_i is obtained by transforming some real valued latent variable $f(x_i)$ associated with x_i. The graphical model for GPC is shown in Figure 1. This graphical model encodes the assumption that x and y are independent given f. A bayesian framework is described with more details in the following.

2.1 Gaussian process prior

We place a Gaussian process prior on the function $f(\cdot)$, meaning that for any finite set $X = \{x_1, ..., x_m\}$, the random vector $\boldsymbol{f} = [f(x_1), ..., f(x_1)]^T$ is a Gaussian. Without loss of generality, we can assume such a process has a zero mean. The covariance between $f(x_i)$ and $f(x_j)$ can be defined as

$$\Sigma_{ij} = c(x_i, x_j) = v_0 \exp\{-\frac{1}{2}\sum_{m=1}^{h} l_m(x_i^m - x_j^m)^2\} + v_1 + v_2\delta(i,j), \tag{1}$$

The hyperparameter v_0 specifies the overall vertical scale of variation of the latent values, v_1 is the overall bias of the latent values from zero mean, v_2 is the latent noise variance, and l_m is the ARD variable for the m-th feature that controls the contribution of this feature in the modelling.
The prior probability of these latent function values $\{f(x_i)\}$ is a multivariate Gaussian

$$p(\boldsymbol{f}) = \frac{1}{(2\pi)^{\frac{n}{2}}\Sigma^{\frac{1}{2}}}\exp(-\frac{1}{2}\boldsymbol{f}^T\Sigma^{-1}\boldsymbol{f}) \tag{2}$$

2.2 Likelihood for class label

The likelihood $p(D|\boldsymbol{f})$ is the joint probability of observing the sample labels given the latent function values. The likelihood can be evaluated as a product of the likelihood function

$$p(D|\boldsymbol{f}) = \{\prod_{i=1}^{n} p(y_i|f_i)\} \tag{3}$$

The rest of this section presents four likelihood functions for class labels

1. **Step function**([4], [3]):

$$p(y_i|f_i) = \epsilon + (1-2\epsilon)H(y_i f_i) \tag{4}$$

where $H(x) = 1$ if $x > 0$, and otherwise 0. The parameter ϵ models labeling error outliers.

2. **Probit function**([4], [1], [5]):

$$p(y_i|f_i) = \Phi(y_i f_i) \tag{5}$$

where Φ is the cumulative distribution function (c.d.f.) of standard Gaussian distribution $N(0,1)$.

3. **Probit function with bias**([6]):

$$p(y_i|f_i) = \Phi(y_i(f_i + b)) \tag{6}$$

where Φ is same as above, and b is the bias parameter. The reason for this choice is that the integral $\int \Phi(y_i(f_i+b))N(f_i)df_i$ can be done analytically for a Gaussian $N(f_i)$. See [6] for a detailed explanation of this noise function.

4. **Probit function with Gaussian noise**([1]):
In the presence of noise from inputs or targets, we may assume that the latent function values are contaminated by a Gaussian noise which is independent of inputs.

If we use δ to denote the noise, then δ has zero mean and an unknown variance σ^2, i.e. $N(\delta; 0, \sigma^2)$. The likelihood function becomes

$$p(y_i|f_i) = \Phi(\frac{y_i f_i}{\sigma}); \tag{7}$$

2.3 Posterior probability

The posterior probability can be written as

$$p(\boldsymbol{f}|D) = \frac{1}{p(D)} \prod_{i=1}^{n} p(y_i|f_i) p(\boldsymbol{f}) \tag{8}$$

where the prior probability $p(\boldsymbol{f})$ is defined as in (2), and $p(D) = \int p(D|\boldsymbol{f}) p(\boldsymbol{f}) d\boldsymbol{f}$

The kernel parameters in the covariance function (1), and labeling error ϵ in step function, bias term b in probit bias function, or the noise variance σ in probit noise function are all collected into θ, which we will call hyperparameters. The normalization factor $p(D)$ in (8), which should be conditional on the hyperparameters $p(D|\theta)$, also known as evidence for θ is a yardstick for model selection. Section (3) discusses how EP can be used for hyperparameter learning.

2.4 Prediction

Suppose we have found the optimal settings of hyperparameters θ^*, then let us take a test sample x_*, for which the class label y_* is unknown. By the definition of Gaussian process, the latent variable $f(x_*)$ and the $\boldsymbol{f} = [f(x_1), ..., f(x_n)]^T$ have a joint multivariate Gaussian distribution, i.e.

$$\begin{bmatrix} \boldsymbol{f} \\ f_* \end{bmatrix} \sim \mathcal{N}\left[\begin{pmatrix} 0 \\ 0 \end{pmatrix}, \begin{pmatrix} \Sigma & \boldsymbol{k} \\ \boldsymbol{k}^T & \mathcal{K}(x_*, x_*) \end{pmatrix} \right]$$

where $\boldsymbol{k} = [\mathcal{K}(x_*, x_1), \mathcal{K}(x_*, x_2), ..., \mathcal{K}(x_*, x_n)]^T$. The conditional distribution of $f(x_*)$ given $\boldsymbol{f}$ is also a Gaussian:

$$p(f_*|\boldsymbol{f}, D, \boldsymbol{\theta}^*) \propto \exp\left(-\frac{1}{2} \frac{(f(x_*) - \boldsymbol{f}^T \Sigma^{-1} \boldsymbol{k})^2}{\mathcal{K}(x_*, x_*) - \boldsymbol{k}^T \Sigma^{-1} \boldsymbol{k}} \right) \tag{9}$$

The predictive distribution of $P(f(x_*)|D, \theta^*)$ can be computed as

$$p(f_*|D, \boldsymbol{\theta}^*) = \int p(f_*|\boldsymbol{f}, D, \boldsymbol{\theta}^*) p(\boldsymbol{f}|D, \boldsymbol{\theta}^*) d\boldsymbol{f} \tag{10}$$

The second term of the integrand, the posterior distribution $p(\boldsymbol{f}|D, \boldsymbol{\theta}^*)$ can be approximated as a Gaussian by the EP approach discussed in section (3). The predictive distribution (10) then can be simplified as a Gaussian $N(f(x_*); \mu_{x_*}, {\sigma_{x_*}}^2)$ with mean μ_{x_*} and variance ${\sigma_{x_*}}^2$. In the EP approach, we reach

$$\mu_{x_*} = \boldsymbol{k}^T (\Sigma + \Pi^{-1})^{-1} \boldsymbol{m} \quad \text{and} \quad {\sigma_{x_*}}^2 = \mathcal{K}(x_*, x_*) - \boldsymbol{k}^T (\Sigma + \Pi^{-1})^{-1} \boldsymbol{k} \tag{11}$$

The predictive distribution over the class label y_* is

$$p(y_*|\mathbf{x}_*, D, \boldsymbol{\theta}^*) = \int p(y_*|f(x_*), \boldsymbol{\theta}^*) p(f(x_*)|D, \boldsymbol{\theta}^*)\, df(x_*) \tag{12}$$

$$= \begin{cases} 1 - \text{normcdf}(0, \mu_{x_*}, \sigma_{x_*}) & \text{step function} \\ \Phi(\frac{y_* \mu_{x_*}}{\sqrt{1+{\sigma_{x_*}}^2}}) & \text{probit function} \\ \Phi(\frac{y_* (\mu_{x_*}+b)}{\sqrt{1+{\sigma_{x_*}}^2}}) & \text{probit function with bias} \\ \Phi(\frac{y_* \mu_{x_*}}{\sqrt{\sigma^2+{\sigma_{x_*}}^2}}) & \text{probit function with Gaussian noise} \end{cases} \tag{13}$$

where $normcdf(x, \mu, \sigma)$ is the cumulative density of a normal distribution with mean μ and variance σ from $-\infty$ to x.

The class label y_{x_*} can be decided as

$$\arg\max_i p(y_* = i|x_*, D, \boldsymbol{\theta}^*)$$

As for the step function, the class label has a special form of

$$\arg\max_i p(y_* = i|x_*, D, \boldsymbol{\theta}^*) = sgn(\sum_{i=1}^{n} \frac{y_i}{\lambda_i} \frac{(1-2\epsilon)N(z_i; 0, 1)}{\epsilon + (1-2\epsilon)\text{erf}(z_i)} \mathcal{K}(x_i, x_*))$$

where z_i and λ_i for step function are defined in the same way as in [3],

3 EP for Gaussian Process Classifiers

Expectation Propagation (EP) algorithm is an approximation Bayesian inference technique that tries to minimize the KL-divergence between the true posterior and the approximation [4]. We review EP in its general form before describing its application to GPCs.

The EP algorithm has been applied in GPCs along with variational methods for model selection ([6], [3], [1]). In the settings of Gaussian processes, EP attempts to approximate $p(\boldsymbol{f}|D)$ as a product distribution in the form of $q(\boldsymbol{f}) = \prod_{i=1}^{n} \tilde{t}_i(f(x_i))p(\boldsymbol{f})$ where $\tilde{t}_i(f(x_i)) = s_i \exp(-\frac{1}{2} p_i (f(x_i) - m_i)^2)$.
The parameters s_i, m_i, p_i in $\tilde{t}_i$ are successively optimized by minimizing the following Kullback-Leibler divergence,

$$\tilde{t}_i^{\text{new}}(\boldsymbol{f}) = \arg\min_{\tilde{t}_i} \text{KL}(\frac{q(\boldsymbol{f})}{\tilde{t}_i^{\text{old}}} p(y_i|f(x_i)) || \frac{q(\boldsymbol{f})}{\tilde{t}_i^{\text{old}}} \tilde{t}_i) \tag{14}$$

Since q is in the exponential family, this minimization is solved by matching moments of the approximated distribution. EP iterates over i until convergence. A detailed updating scheme for EP-GPC with probit functions (with and without Gaussian noise) can be found in Appendix A. The algorithm is not guaranteed to converge although it did in practice. At equilibrium of $q(\boldsymbol{f})$, we obtain an approximate posterior distribution as

$$p(\boldsymbol{f}|D) \approx N(\boldsymbol{f}; (\Sigma^{-1} + \Pi)^{-1} \Pi \boldsymbol{m}, (\Sigma^{-1} + \Pi)^{-1}) \tag{15}$$

where Π is the diagonal matrix whose ii-th entry is p_i and $\boldsymbol{m} = [m_1, ..., m_n]^T$.

Variational methods can be used to optimize the hyperparameter $\boldsymbol{\theta}$ by maximize the lower bound of the logarithm of the evidence, which has the following form

$$\log p(D|\boldsymbol{\theta}) = \log \int \frac{p(D|\boldsymbol{f})p(\boldsymbol{f})}{q(\boldsymbol{f})} q(\boldsymbol{f}) d\boldsymbol{f} \geq \int q(\boldsymbol{f}) \log \frac{p(D|\boldsymbol{f})p(\boldsymbol{f})}{q(\boldsymbol{f})} d\boldsymbol{f}$$
$$= \int q(\boldsymbol{f}) \log p(D|\boldsymbol{f}) d\boldsymbol{f} + \int q(\boldsymbol{f}) \log p(\boldsymbol{f}) d\boldsymbol{f} - \int q(\boldsymbol{f}) \log q(\boldsymbol{f}) d\boldsymbol{f} = F(\theta) \tag{16}$$

Given the expression of the lower bound $F(\boldsymbol{\theta})$ in terms of $q(\boldsymbol{f})$, the gradients of $F(\boldsymbol{\theta})$ with respect to $\boldsymbol{\theta}$ can be derived by neglecting the possible dependency of $q(\boldsymbol{f})$ on $F(\boldsymbol{\theta})$. The detailed formulation can be found in Appendix B.

4 The EM-EP Algorithm

Kim and Ghahramani [3] proposed a conceptually simple EM-like algorithm to learn the hyperparameters which we refer as EM-EP algorithm. The algorithm works as follows.

1. **E-step** EP iterations are performed given the hyperparameters. $p(\boldsymbol{f}|D)$ is approximated as a Gaussian density $q(\boldsymbol{f})$ given by Equation (15)
See Appendix A for the EP algorithm for the four models

2. **M-step** Given $q(\boldsymbol{f})$ obtained from the E-step, find the hyperparameters which maximize the variational lower bound of $\log p(D|\theta)$.

The E-step and the M-step are alternated until convergence. See Appendix B for a derivation of the M-step. The gradient update rules with respect to the hyperparameters can be found in Appendix B.

5 Experimental Results

One of the goals of this paper is to compare various likelihood functions of GPC using EP. We start this section with two simple synthetic datasets in circular shape to visualize the behavior of these algorithms. Another goal of this paper is to compare EP with other inference techniques, such as Laplace approximation, for GPC. To do this, we test EP-GPC and Laplace-GPC on gene expression data, and the performances of these two algorithms are compared with SVM.

5.1 Implementation

Based on Kim's matlab code of EM-EP algorithm with step likelihood function, we implemented EM-EP algorithm with two other likelihood functions, namely probit function and probit function with bias. Chu [1] publishes his C code of GPC for ordinal regression. By some simple algebra, we can see that in the case of binary classification, the generalized likelihood function in Chu's ordinal regression model becomes the probit function with Gaussian noise, so we use his code to test this likelihood function.
In the first experiment with two dimensional synthetic datasets, we use Kim's matlab code, our matlab code, and Chu's C code to test various likelihood functions. Because of the lower speed of matlab implementation, we only use Chu's C code to test EP for GPC on high dimensional gene expression data.

5.2 Synthetic Data

We use the same data as in [3]. The two-dimensional data set with input features (i.e. dimensions) x_1 and x_2 are random numbers between -1 and 1. The decision rule for class -1 and +1 is $x_1^2 + x_2^2 \geq 0.5$. A total of 1000 data points are generated with 40 of them as the training set and the remaining 960 samples as the test set. Two such type of datasets are used. The two data sets are same excepted that we added labeling errors to two randomly selected data points in the training set. Test results are shown in Figure 2-8.

For the step function, we compare the case where labeling error hyperparameter ϵ is fixed ($\epsilon = 0$) to the one with adapted labeling error hyperparameter on the two datasets. For the dataset without outliers, the model with fixed ϵ achieves the same test error rate as the one with adapted ϵ. Classification results are shown in Figure 2. However, the model with fixed ϵ converges faster as shown in Figure 3. This makes sense because it takes time for parameters, in this case ϵ, with improper initial setting to be adapted. In the presence of outliers, The algorithm with adapted ϵ outperforms the one with fixed ϵ, because fixed ϵ can cause overfitting as shown in Figure 4. The convergence properties of the algorithms are shown in Figure 5.

Three versions of probit functions can not handle the dataset in circular shape well. The algorithm with either version of probit function achieves a test error rate of 12% or so. Classification results with one type of probit function is shown in Figure 6. The algorithms using probit functions fail if the data contain outliers. As shown in Figure 7, the classificaiton performances of the classifiers with probit functions are no better than random guess in presence of outliers. This may suggest that probit function models a general dependency (with tractable noise) of class labels on latent function values, not extreme case, like outlyingness or mislabelling.

5.3 Gene Expression Data

We applied the algorithm using probit function with Gaussian noise to high-dimensional gene expression dataset: colon cancer.

For the colon dataset, the task is to discriminate tumor from normal tissues. The dataset has 22 normal and 40 cancer samples with 2000 features per sample. We randomly split the dataset into 42 training and 20 test samples 10 times and run SVM, EP-GPC, and Laplace-GPC on each partition. Test results of the three algorithms using linear kernels are presented in Table 1 and Table 2. Note that even though Laplace approach reaches a smaller variance in the case where all 2000 features are used, the test error rates on this dataset are both large. It makes sense to use predictive variance as a criteria to evaluate classification performance only if the classification accuracy is reasonably well.

By using some feature selection techniques, such that ARD, we can eliminated irrelevant features. In this case, we use the dataset containing 373 relevant features selected according to optimal ARD parameter values. All of the three algorithms achieve higher accuracies on this smaller dataset, and the two GPCs outperform SVM. EP-GPC and Laplace-GPC have comparative classification accuracies, but EP-GPC approach is better than Laplace-GPC in the sense that the variance of the predictive distribution using EP is smaller.

6 Conclusion and Future Work

Based on the work of [4, 6, 1] on noise models, and EM-EP algorithm proposed in [3], we investigate four likelihood functions and derive some special form of updating rules for EM-EP algorithms in the case of binary classification. Experiments showed step function with labeling error hyperparameter handles outliers well. GPC with EM-EP showed better performances than both SVM and GPC with Laplace-MAP on gene expression data. By incorporating some Bayesian feature selection techniques, such as ARD, we can improve the performance of GPCs on high dimensional data.

GPCs have some advantages over other kernel methods because they are fully statistical models. This suggests that we can improve the performance of GPCs from many different directions. For example, we can incorporate prior information to inform learning of the hyperparameters. We can improve approximate inference and optimization techniques to gain in both accuracy and speed. Also, a good follow-up of this project is to build a better likelihood function. Experiments show step function handles outliers well, and we believe probit functions may handle general noisy data better. As future work, we can do experiments to verify our hypothesis. In real world, Microarray gene expression data are usually both mislabeled and noisy. A robust likelihood function which captures both properties is desired.

Acknowledgements

Thanks to Dr. Kevin Murphy for directing papers on the topics of EP for GPC. Thanks to H. Kim for providing his matlab code of EMEP for GPC with step likelihood function. Thanks to W. Chu for providing the C code of GPC for ordinal regression and gene expression data.

Appendix A. Approximate posterior distribution by EP [1]

The EP algorithm using step likelihood function and probit function with bias are presented in [3] and [6] respectively. We skip these algorithms because of the limited space. The updating scheme of EP algorithm for the other two likelihood functions, namely probit function and probit function with Gaussian noise, are presented below.

1. Initialization (same for all likelihood functions)

- individual mean $m_i = 0 \ \forall i$;
- individual inverse variance $p_i = 0 \ \forall i$;
- individual amplitude $s_i = 1 \ \forall i$
- posterior covariance $A_i = (\Sigma^{-1} + \Pi)^{-1}$, where $\Pi = diag(p_1, p_2, ..., p_n)$;
- posterior mean $\boldsymbol{h} = A\Pi\boldsymbol{m}$, where $\boldsymbol{m} = [m_1, m_2, ..., m_n]^T$.

2. Looping from $i = 1, ..., n$ until all (m_i, p_i, s_i) converge:

(1) Remove $\tilde{t}_i$ from the posterior $q(\boldsymbol{f})$ to get a leave-one-out posterior distribution $q^{\setminus i}(\boldsymbol{f})$ having

[1] A major portion of the fomulae in this Appendix is based [1]'s work on ordinal regression, and we present the special form for binary classification

- variance of f_i: $\lambda_i^{\setminus i} = \frac{A_{ii}}{1-A_{ii}}$
- mean of f_i: $h_i^{\setminus i} = h_i + \lambda_i v_i^{-1}(h_i - m_i)$
- others with $j \neq i : \lambda_i^{\setminus i} = A_{jj}$ and $h_j^{\setminus i} = h_j$

(2) Compute new posterior approximation by incorporating the message $p(y_i|f_i)$ into $q^{\setminus i}(f)$:

If Probit Function with Gaussian noise [2] is used, then
- $\mathcal{Z}_i = \int \mathcal{P}(y_i|f(x_i))\mathcal{N}(f(x_i); h_i^{\setminus i}, \lambda_i^{\setminus i})df(x_i) = \Phi(\tilde{z}_i)$

where $\tilde{z}_i = \frac{y_i h_i^{\setminus i}}{\sqrt{\lambda_i^{\setminus i}+\sigma^2}}$
$\alpha_i = \frac{\partial \log \mathcal{Z}_i}{\partial h_i^{\setminus i}} = -\frac{1}{\sqrt{\lambda_i^{\setminus i}+\sigma^2}}(\frac{\mathcal{N}(\tilde{z}_i;0,1)}{\Phi(\tilde{z}_i)})$
$\beta_i = \frac{\partial \log \mathcal{Z}_i}{\partial \lambda_i^{\setminus i}} = -\frac{1}{2(\lambda_i^{\setminus i}+\sigma^2)}(\frac{y_i \tilde{z}_i \mathcal{N}(\tilde{z}_i;0,1)}{\Phi(\tilde{z}_i)})$

- $v_i = \alpha_i^2 - 2\beta_i$
- $h_i^{new} = h_i^{\setminus i} + \lambda_i^{\setminus i}\alpha_i$
- $p_i^{new} = \frac{v_i}{1-\lambda_i^{\setminus i} v_i}$
- $m_i^{new} = h_i^{\setminus i} + \frac{\alpha_i}{v_i}$
- $s_i^{new} = Z_i\sqrt{\lambda_i^{\setminus i} p_i^{new} + 1}\exp(\frac{\alpha_i^2}{2v_i})$

(3) If $p_i^{new} > 0$, update $\{p_i, m_i, s_i\}$, the posterior mean h and covariance $\mathcal{A}$.

- $\mathcal{A}^{new} = \mathcal{A} - \rho a_i a_i^T$ where $\rho = \frac{p_i^{new}-p_i}{1+(p_i^{new}-p_i)\mathcal{A}_{ii}}$ and a_i is the i-th column of $\mathcal{A}$. (if $p_i^{new} \approx p_i$, skip this sample and this updating.)

- $h^{new} = h + \eta a_i$ where $\eta = \frac{\alpha_i + p_i(h_i - m_i)}{1-\mathcal{A}_{ii}p_i}$

The approximate evidence can be obtained in the same way as for BPMs:
$\mathcal{P}(\mathcal{D} \mid \theta)$ at the EP solution, which can be written as

$$\prod_{i=1}^{n} s_i \frac{\det^{\frac{1}{2}}(\prod^{-1})}{\det^{\frac{1}{2}}(\sum + \prod^{-1})}\exp(\frac{B}{2})$$

where $B = \sum_{ij} \mathcal{A}_{ij}(m_i p_i)(m_j p_j) - \sum_i p_i m_i^2$

Appendix B. Gradient Formulae for Variational Bound

At the equilibrium of $q(\boldsymbol{f})$, the variational bound $\mathcal{F}$ can be analytically calculated as follows:

[2] If probit function is used for likelihood, simply replace every occurrence of σ^2 by 1 in the expressions of $\mathcal{Z}_i$, $\tilde{z}_i$, α_i, and β_i.

$$\mathcal{F}(\theta) = \sum_{i=1}^{n} \int N(f(x_i); h_i, A_{ii}) \ln(p(y_i|f(x_i)))df(x_i) - \frac{1}{2}\ln|I + \Sigma\Pi|$$
$$-\frac{1}{2}\text{trace}((I + \Sigma\Pi)^{-1}) - \frac{1}{2}\boldsymbol{m}^T(\Sigma + \Pi^{-1})^{-1}\Sigma(\Sigma + \Pi^{-1})^{-1}\boldsymbol{m} + \frac{n}{2} \tag{17}$$

Again, we skip the gradient formulae for step function. Refer to [3] if interested. Let κ denote the vector containing the hyperparameters in covariance function, b denote the bias in the probit function with bias, and θ denote the noise variance in probit function with Gaussian noise. Then the gradient of $\mathcal{F}(\theta)$ with respect to the variables $\ln\kappa, \ln b, \ln\sigma$ can be given in the following:

$$\begin{aligned}\frac{\partial\mathcal{F}(\theta)}{\partial\ln\kappa} &= \kappa\int Q(\boldsymbol{f})\frac{\partial\log\mathcal{P}(\boldsymbol{f})}{\partial\kappa}d\boldsymbol{f} \\ &= -\frac{\kappa}{2}\text{trace}(\Sigma^{-1}\frac{\partial\Sigma}{\partial\kappa}) + \frac{\kappa}{2}\boldsymbol{h}^T\Sigma^{-1}\frac{\partial\Sigma}{\partial\kappa}\Sigma^{-1}\boldsymbol{h} + \frac{\kappa}{2}\text{trace}(\Sigma^{-1}\frac{\partial\Sigma}{\partial\kappa}\Sigma^{-1}\mathcal{A}) \\ &= -\frac{\kappa}{2}\text{trace}((\Pi^{-1} + \Sigma)^{-1}\frac{\partial\Sigma}{\partial\kappa}) + \frac{\kappa}{2}\boldsymbol{m}^T(\Pi^{-1} + \Sigma)^{-1}\frac{\partial\Sigma}{\partial\kappa}(\Pi^{-1} + \Sigma)^{-1}\boldsymbol{m}\end{aligned} \tag{18}$$

$$\begin{aligned}\frac{\partial\mathcal{F}(\theta)}{\partial\ln\sigma} &= \sigma\sum_{i=1}^{n}\int\mathcal{N}(f(x_i); h_i, \mathcal{A}_{ii})\frac{\partial\ln\mathcal{P}(y_i \mid f(x_i))}{\partial\sigma}df(x_i) \\ &= \sum_{i=1}^{n} y_i\int\mathcal{N}(f(x_i); \frac{h_i\sigma^2}{\sigma^2 + \mathcal{A}_{ii}}, \frac{\sigma^2\mathcal{A}_{ii}}{\sigma^2 + \mathcal{A}_{ii}})\frac{\frac{-f(x_i)}{\sqrt{2\pi(\sigma^2+\mathcal{A}_{ii})}}\exp(-\frac{(h_i)^2}{2(\sigma^2+\mathcal{A}_{ii})})}{\mathcal{P}(y_i \mid f(x_i))}df(x_i)\end{aligned} \tag{19}$$

$$\frac{\partial\mathcal{F}(\theta)}{\partial\ln b} = \sum_{i=1}^{n}\int\mathcal{N}(f(x_i); h_i, \mathcal{A}_{ii})\frac{(f_i - h_i)}{\mathcal{A}_{ii}}\ln\Phi(y_i(f(x_i) + b))df(x_i) \tag{20}$$

The last two integrals can be approximated using Gaussian quadrature.

References

[1] Chu, W. & Z. Ghahramani. Gaussian processes for ordinal regression. *Technical report, Gatsby Unit, University College London*, 2004. www.gatsby.ucl.ac.uk/ chuwei/paper/gpor.pdf

[2] Qi, Y., T.P.Minka, R.W.Picard, & Z. Ghahramani. Predictive automatic relevance determination by expectation propagation. In *Proceedings of the Twenty-first International Conference on Machine Learning*, pages 671-678, 2004.

[3] Kim, H. & Z. Ghahramani. The EM-EP algorithm for Gaussian process classification. In *Proceedings of the Workshop on Probabilistic Graphical Models for Classification (at ECML)*, 2003

[4] Minka, T.P. A family of algorithm for approximate Bayesian inference. *Ph.D. thesis, Massachusetts Institute of Technology*, Janurary 2001.

[5] Neal, R. M. Monte Carlo implementation of Gaussian process models for Bayesian regression and classification. *Technical Report No. 9702, Department of Statistics, University of Toronto*, 1997.

[6] Seeger, M. Notes on Minka's expectation propagation for Gaussian process classification. *Technical report, University of Edinburgh*, 2002.

[7] Rasmussen, C.R. Gaussian Process in Machine Learning. *Advanced Lectures on Machine Learning*: ML Summer Schools 2003, Canberra, Australia, February 2 - 14, 2003, Tübingen, Germany, August 4 - 16, 2003.

[8] Rasmussen, V. The Nature of Statistical Learning Theory. Springer, New York(1995).

[9] Williams, C.K.I and D. Barber. Bayesian classification with Gaussian processes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 20(12):1342-1351, 1998.

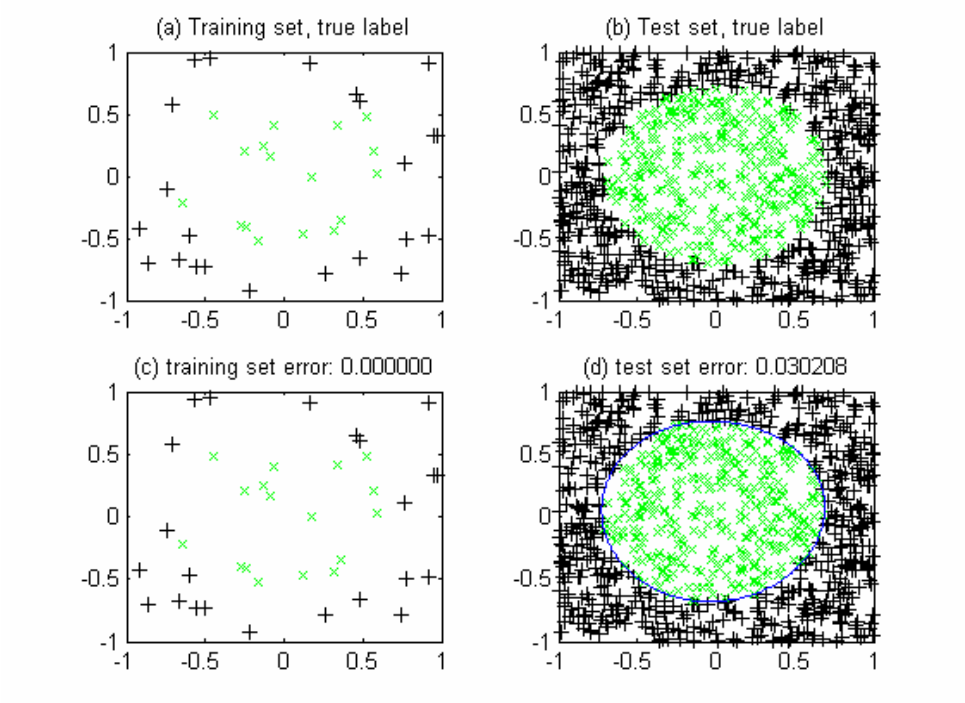
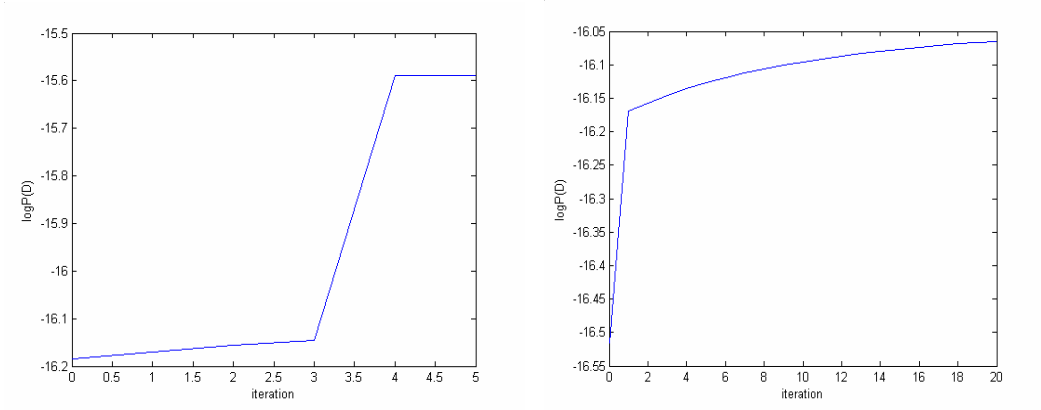


Figure 2: Classification Results on Data without Outliers Using Step Function

(a) Training set, (b) Test set, (c) Classification result on training data using the optimal hyperparameter settings trained with the training set, (d) Classification result on test set



(a) fixed ε (ε= 0) (b) adapted ε (ε= 0.01 initially)

Figure 3: Logarithm of Evidence at Each Iteration on Data without outliers using Step Function

The evidence converges faster with fixed labeling error (ε= 0) than the case where ε was adapted.

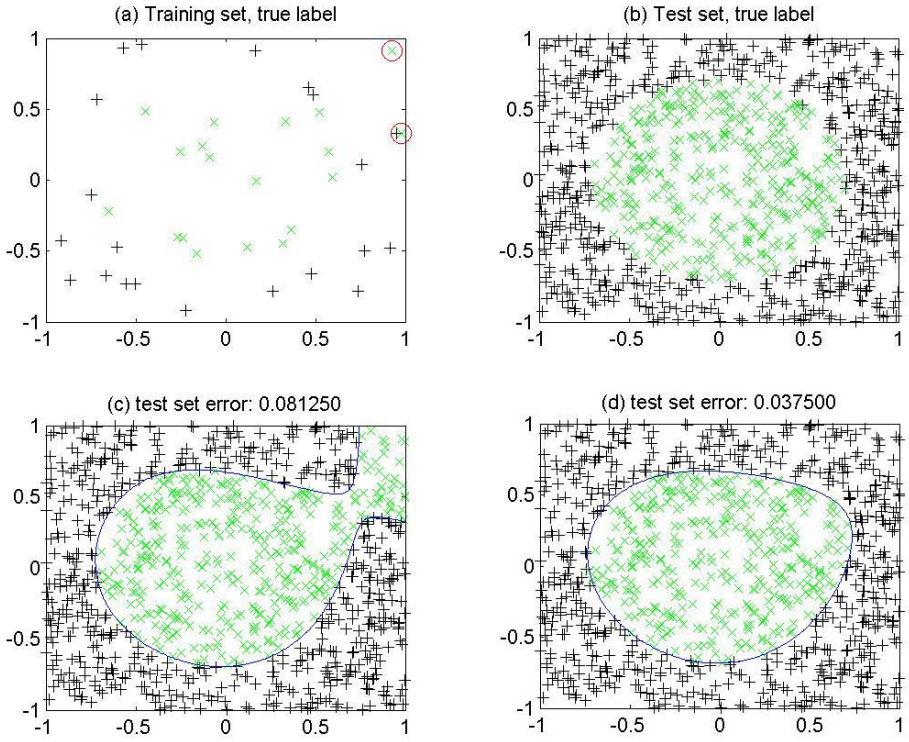
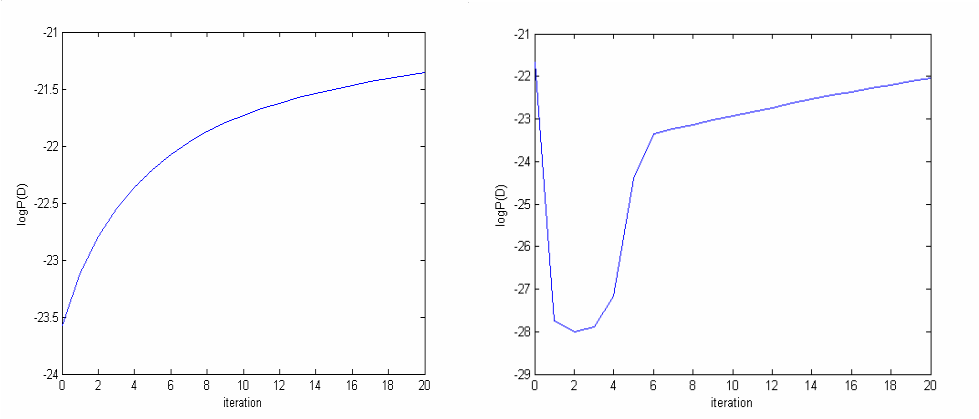


Figure 4: Classification Results on Data with Outliers Using Step Function

(a) Training set with two outliers, i.e. data points that are mislabeled, which are circled red, (b) Test set, (c) Classification result with fixed labeling error hyperparameter (ε= 0), (d) Classification result with adapted labeling error hyperparameter ε



(a) fixed ε (ε= 0) (b) adapted ε (ε= 0.01 initially)

Figure 5: Logarithm of Evidence at Each Iteration on Data with Outliers using Step Function

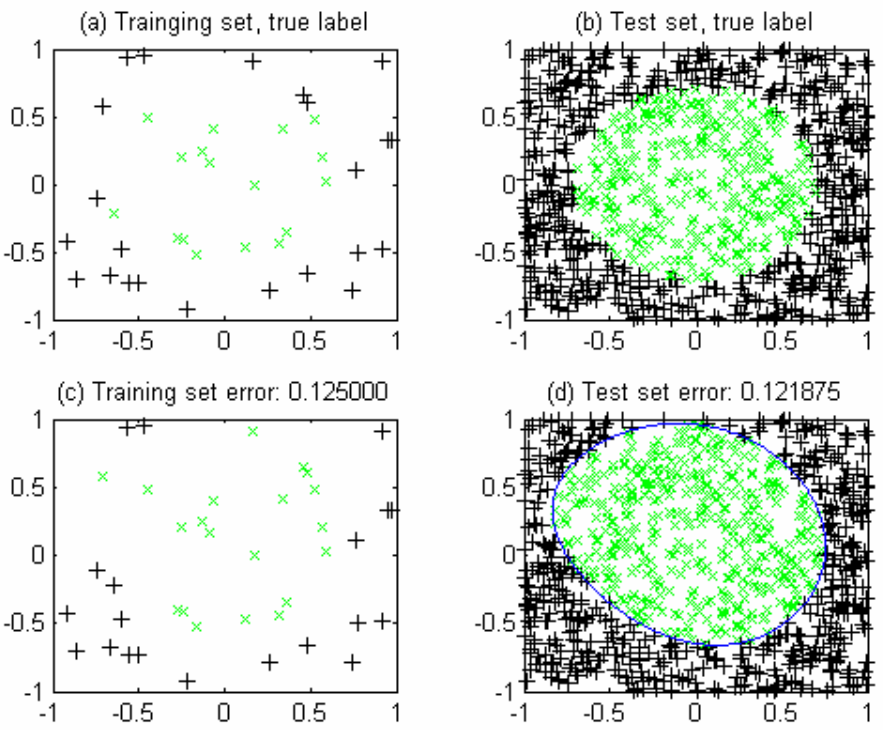


Figure 6: Classification Results on Data without outliers Using Probit functions

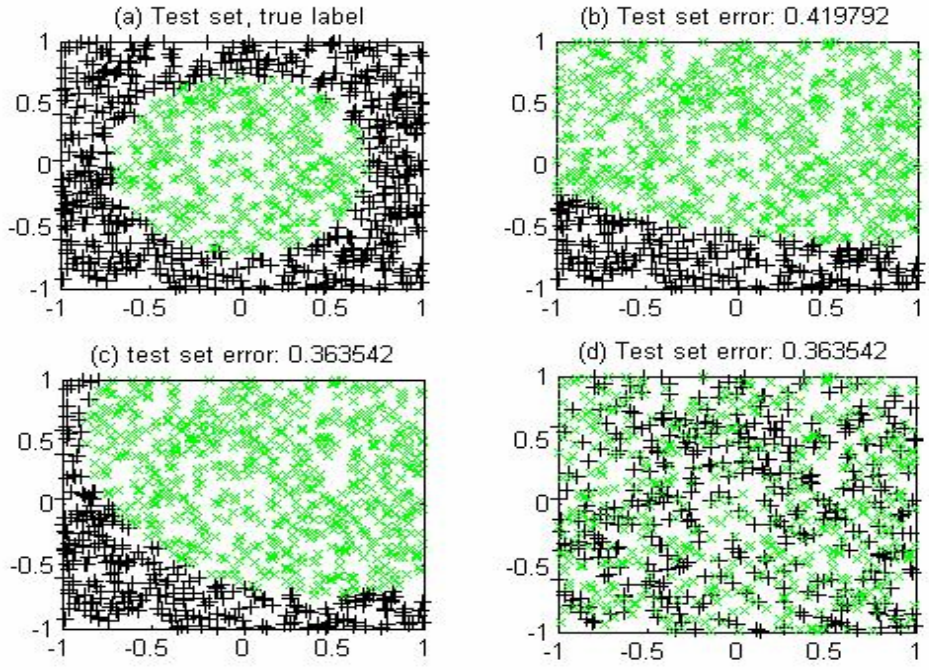


Figure 7: Classification Results on Data with outliers Using Probit functions

Algorithms	Test Error Rate (%)	Predictive Variance
SVM	0.226	-
Laplace-MAP	0.219	3.49
EP	0.19	6.72

Table 1. Test error rates with all 2000 genes

Algorithms	Test Error Rate (%)	Predictive Variance
SVM	0.216	-
Laplace-MAP	0.133	3.16
EP	0.133	4.15

Table 2. Test error rates with 373 relevant genes