Semi-Supervised Classification Using Sparse Gaussian Process Regression

Amrish Patel*
Computer Science and Automation
Indian Institute of Science, India
amrish@csa.iisc.ernet.in

S. Sundararajan
Yahoo! Labs
Bangalore, India
ssrajan@yahoo-inc.com

Shirish Shevade
Computer Science and Automation
Indian Institute of Science, India
shirish@csa.iisc.ernet.in

Abstract

Gaussian Processes (GPs) are promising Bayesian methods for classification and regression problems. They have also been used for semi-supervised learning tasks. In this paper, we propose a new algorithm for solving semi-supervised binary classification problem using sparse GP regression (GPR) models. It is closely related to semi-supervised learning based on support vector regression (SVR) and maximum margin clustering. The proposed algorithm is simple and easy to implement. It gives a sparse solution directly unlike the SVR based algorithm. Also, the hyperparameters are estimated easily without resorting to expensive cross-validation technique. Use of sparse GPR model helps in making the proposed algorithm scalable. Preliminary results on synthetic and real-world data sets demonstrate the efficacy of the new algorithm.

1 Introduction

Supervised learning algorithms require enough labeled training data to learn reasonably accurate classifiers which generalize well. But, in many application domains like bioinformatics or text processing, labeled data are often expensive to get. In such applications, unlabeled data are easily available in abundance. Semi-supervised learning uses large amount of unlabeled data, along with the labeled data, to build better classifiers. Zhu [2005] gives a detailed review of the literature on semi-supervised learning.

In this paper, we consider the semi-supervised learning problem which involves binary classification task and assume that a decision boundary passes through the *low density* region which separates the two classes [Zhu, 2005]. Bennett and Demirez [1998] formulated a semi-supervised support vector machine (SVM) problem and proposed to solve it using mixed integer programming method. The resulting solution labels the unlabeled data so as to maximize the margin.

Xu *et al* [2004] proposed an extension of a maximum margin clustering problem to a semi-supervised learning problem by formulating it as a constrained semi-definite programming (SDP) problem. The SDP solvers are, however, computationally very expensive. Lawrence *et al* [2005] proposed a Gaussian process based algorithm using the *null category noise model* (NCNM) for semi-supervised classification (SSC). This algorithm designs a sparse GP classifier and finds the decision boundary which avoids dense unlabeled data region. However, the effective likelihood associated with this noise model is not log-concave. Therefore, the Gaussian approximation to the noise model can have negative variance. Sindhwani *et al* [2007] proposed a graph-based construction of semi-supervised GP classifier. This method adapts a covariance function, based on the geometry of unlabeled data. The resulting GP classifier is however not sparse. Thus, there is a need to design a simple and sparse GP classifier for SSC which gives comparable/better generalization performance with state-of-the-art GP classifiers.

In this paper, we propose new algorithms for semi-supervised classification, which combine maximum margin clustering idea with support vector regression (SVR) and sparse GP regression (GPR) models. Zhang *et al* [2007] proposed a simple and practical approach for clustering based on maximum margin and SVR. We first propose to extend these ideas to semi-supervised classification. This algorithm does not directly result in a sparse solution, unless re-training is used. Sparse solutions are preferred because of lower computational complexity and ease of interpretation. In many applications involving decision making, one is interested in developing probabilistic models. SVR based algorithm for SSC does not provide probabilistic interpretation of the results on unseen test data. Tuning of hyperparameters is done using expensive techniques like cross-validation. Such techniques are not reliable in SSC where the number of labeled examples is small. With this motivation, we propose to design a sparse GPR model for SSC. The algorithm is simple and easy to implement. Further, the use of sparse GPR model helps in making it scalable. As is the case with standard GP based algorithms, hyperparameter tuning is done easily without resorting to techniques like cross-validation. We evaluated the performance of the proposed algorithm on synthetic and real-world data sets. Comparisons with the SVR based algorithm and the NCNM algorithm indicate that the proposed algorithm is a useful alternative to design sparse classifiers for SSC.

In the next Section, maximum margin clustering and GP based algorithms for semi-supervised classification are re-

*The support of DST, Govt of India is acknowledged.

viewed. The proposed algorithms are presented in Section 3. Section 4 gives experimental results which demonstrate the effectiveness of the proposed sparse GPR based algorithm on various data sets. Section 5 concludes the paper.

2 Background

In semi-supervised classification, we are given a training data set composed of L labeled examples, $D = \{\mathbf{x}_i, y_i\}_{i=1}^{L}$, $\mathbf{x}_i \in \mathbb{R}^d$, $y_i \in \{-1, 1\}$, and U unlabeled examples, $D' = \{\mathbf{x}_i^*\}_{i=1}^{U}$. Traditional classifiers are designed using only the set D. In SSC, our aim is to use D' along with D to determine $y_i^*, i = 1, \ldots, U$ and design a classifier having better generalization performance. We now give a brief overview of maximum margin clustering and some GP based approaches for semi-supervised classification discussed in the literature.

2.1 Maximum Margin Clustering

In clustering problems, the class labels are unknown. Given the input data $\{\mathbf{x}_i^*\}_{i=1}^{U}$, the aim of a clustering algorithm is to assign labels y_i^* to the training data such that *similar* data points have the same label. Use of large margin methods for clustering problems is gaining wide popularity [Xu *et al.*, 2004; Zhang *et al.*, 2007]. Zhang *et al* proposed a practical approach for maximum margin clustering. They observed that an iterative algorithm for maximum margin clustering using support vector machine (SVM) classifier did not perform well because of the problem of poor local minima. So, to improve the performance, they proposed to replace the SVM classifier by SVR with Laplacian loss.

In regression problems, we are given a pair of input-output samples, $\{(\mathbf{x}_i^*, y_i^*)\}_{i=1}^{U}$ where $y_i^* \in \mathbb{R}$. Let the function value at $\mathbf{x}_i^*$ be represented by an unobservable latent variable $f(\mathbf{x}_i^*)$. The goal of SVR problem is to find a function $f(\mathbf{x}^*) = \mathbf{w}^T\phi(\mathbf{x}^*) + b$ which best fits the data. Here, ϕ is a mapping induced by a kernel function k. The primal problem for SVR with Laplacian loss can be written as

$$\begin{array}{ll} \min_{\mathbf{w},b,\xi_i^*} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i^* \\ \text{s.t.} & |y_i^* - (\mathbf{w}^T\phi(\mathbf{x}_i^*) + b)| = \xi_i^* \ \ \forall\ i \end{array} \tag{1}$$

where $C > 0$ is a hyperparameter. Zhang *et al* [2007] used the following SVR formulation to devise an iterative algorithm for maximum margin clustering:

$$\begin{array}{ll} \min_{\mathbf{w},b,y_i^*,\xi_i^*} & \frac{1}{2}\|\mathbf{w}\|^2 + C\sum_i \xi_i^* \\ \text{s.t.} & |y_i^* - (\mathbf{w}^T\phi(\mathbf{x}_i^*) + b)| = \xi_i^* \ \ \forall\ i \\ & y_i^* \in \{-1, 1\}\ \forall\, i, -l \le \mathbf{e}^T\mathbf{y}^* \le l \end{array} \tag{2}$$

where a cluster balance constraint has been added to the formulation (1) to avoid meaningless solutions, $l \ge 0$ is a constant controlling the class imbalance and $\mathbf{e}$ is the vector of ones. As mentioned earlier, y_i^*'s are not known in clustering problems. In the iterative SVR based algorithm for clustering, the idea is to initialize y_i^*'s using a simple clustering algorithm. The dual problem of (2) is then solved and optimal $\mathbf{w}$ is obtained using Karush-Kuhn-Tucker (KKT) conditions. Given this $\mathbf{w}$, the optimal values of b and $\mathbf{y}^*$ can be obtained by solving the following problem:

$$\begin{array}{ll} \min_{b,y_i^*} & |y_i^* - (\mathbf{w}^T\phi(\mathbf{x}_i^*) + b)| \\ \text{s.t.} & y_i^* \in \{-1, 1\}\ \ \forall\ i \\ & -l \le \mathbf{e}^T\mathbf{y}^* \le l \end{array} \tag{3}$$

This formulation (3) can be solved easily by sorting the values of $\mathbf{w}^T\phi(\mathbf{x}_i^*)$ and then setting b such that the cluster balance constraint is satisfied and the objective function value in (3) is minimized. In the iterative SVR algorithm for maximum margin clustering, the dual of (2) and the formulation (3) are solved repeatedly till some stopping condition is satisfied [Zhang *et al.*, 2007].

2.2 Gaussian Processes for Semi-Supervised Classification

We now briefly describe GPs and some algorithms which use GPs for semi-supervised classification.

The learning in GPs involves learning of latent functions f of the input variable. In standard GPs, the latent variables $\{f(\mathbf{x}_i)\}$ are random variables with prior distribution, $P(\mathbf{f}) = \mathcal{N}(\mathbf{f}; \mathbf{0}, \mathbf{K})$, where $\mathbf{f} = (f_1, \ldots, f_L)^T$, $f_i = f(\mathbf{x}_i)$ and $\mathbf{K}$ is a $L \times L$ covariance matrix whose ij-th element is $k(\mathbf{x}_i, \mathbf{x}_j)$, $k(\cdot)$ being the kernel function. We call the parameters associated with the kernel function as kernel hyperparameters. The posterior distribution over $\mathbf{f}$ is $P(\mathbf{f}|\mathbf{y}) \propto P(\mathbf{y}|\mathbf{f})P(\mathbf{f})$. If the output observations y_i's are assumed independent from the input data, $P(\mathbf{y}|\mathbf{f}) = \prod P(y_i|f_i)$ where $P(y_i|f_i)$ is called a noise model. For regression case the noise model can take the form of Gaussian while for the classification case it can be the probit noise model [Rasmussen and Williams, 2006].

Lawrence *et al* [2005] proposed to use the "Null Category Noise Model" (NCNM) for SSC using sparse GP classifiers. This model results in a region in the input space, analogous to the notion of margin in support vector machines, that excludes unlabeled data points. For this purpose, a new category for the output variable y was introduced. With this, $y \in \{-1, 0, 1\}$. Further, a constraint was imposed such that unlabeled data point never comes from the category, $y = 0$. When a data point is labeled, the model acts as a standard binary classification model. On the other hand, for an unlabeled point, the assignment of label to it depends on the mean and variance of $P(f_i|x_i)$. Note that the effective likelihood associated with the proposed noise model is not log-concave. Therefore, the best Gaussian approximation to the noise model can have negative variance. Such cases are handled by setting a negative variance to zero. Also, posterior variance can increase when a point is included. To make the inference tractable, approximation inspired by Assumed Density Filtering (ADF) was proposed. This technique approximates the posterior with a Gaussian by matching the moments between the true posterior and the approximation [Rasmussen and Williams, 2006].

Sindhwani *et al* [2007] proposed to use a graph based construction of semi-supervised GP classifier. This approach is based on the intuition that, for many real world problems, unlabeled examples often identify structures such as a low dimensional manifold, whose knowledge may improve inference. Such geometric properties of unlabeled data are incorporated into the covariance function. Expectation Propagation ideas were used for approximating posterior process and model selection. But, the resulting GP classifier was not sparse.

Thus, there is a need to have a simple and sparse GP classifier for SSC.

3 Simple Algorithms for Semi-Supervised Classification

In this section, we present an SVR based algorithm for semi-supervised classification using the idea of maximum margin clustering. This algorithm is then extended to design a sparse GPR model for semi-supervised classification.

3.1 Semi-Supervised Classification using SVR (SSuSVR)

The problem formulation (2) can be easily extended to solve the SSC problem. In particular, the new problem can be written as

$$\begin{array}{ll}\min_{\mathbf{w},b,y_i^*,\xi_i,\xi_i^*} & \frac{1}{2}\|\mathbf{w}\|^2 + C_L\sum_{i=1}^{L}\xi_i + C_U\sum_{i=1}^{U}\xi_i^* \\ \text{s.t.} & |y_i - (\mathbf{w}^T\phi(\mathbf{x}_i)+b)| = \xi_i,\ i=1,\ldots,L \\ & |y_i^* - (\mathbf{w}^T\phi(\mathbf{x}_i^*)+b)| = \xi_i^*,\ i=1,\ldots,U \\ & y_i^* \in \{-1,1\}\ \forall\, i, -l \le \mathbf{e}^T\mathbf{y}^* \le l \end{array} \tag{4}$$

where C_L and C_U are positive hyperparameters. By fixing y_i^*'s, it is easy to write the dual of the above problem. The optimal $\mathbf{w}$ can be found using the KKT conditions. For this optimal $\mathbf{w}$, the optimal b and y_i^*'s can be obtained by solving the following problem:

$$\begin{array}{ll}\min_{b,y_i^*} & C_L\sum_{i=1}^{L}\xi_i + C_U\sum_{i=1}^{U}\xi_i^* \\ \text{s.t.} & |y_i - (\mathbf{w}^T\phi(\mathbf{x}_i)+b)| = \xi_i,\ i=1,\ldots,L \\ & |y_i^* - (\mathbf{w}^T\phi(\mathbf{x}_i^*)+b)| = \xi_i^*,\ i=1,\ldots,U \\ & y_i^* \in \{-1,1\}\ \forall\, i, -l \le \mathbf{e}^T\mathbf{y}^* \le l \end{array} \tag{5}$$

The above problem can be solved along the lines suggested in subsection 2.1. Thus the algorithm for semi-supervised classification repeatedly finds the solutions of the dual of (4) and (5) until a stopping condition is satisfied. We call this algorithm as SSuSVR (Semi-supervised Classification using SVR) and it is given below:

Algorithm 1 Semi-Supervised Classification using SVR (SSuSVR)

Train a SVM classifier using labeled points and find $\mathbf{y}^*$.
repeat
  Train SVR by solving the dual of (4).
  Compute $\mathbf{w}$ from the KKT conditions.
  Compute the bias b using the method described in 2.1.
  Set $y_i^* = \text{sgn}(\mathbf{w}^T\phi(\mathbf{x}_i^*)+b)\ \ \forall\, i=1,\ldots,U$.
until Algorithm converges or maximum number of iterations are finished

Note that the positive hyperparameters, C_L and C_U are typically determined using cross-validation. In the case of SSC, only a few labeled examples are available and therefore, the cross-validation technique is not guaranteed to give proper values of hyperparameters. Further, the Laplacian loss function used in the formulation (4) does not directly result in a sparse solution of the problem. Also, the resulting model is not probabilistic. These problems can be alleviated by using sparse GPR models for SSC. GPs use a principled approach for hyperparameter determination and also give confidence about prediction. We now discuss the proposed GP based algorithms for semi-supervised classification.

3.2 Semi-Supervised Classification using GPR (SSuGPR)

For Gaussian process regression, we can write $\mathbf{y} = \mathbf{f} + \boldsymbol{\psi}$, where $\boldsymbol{\psi} \sim \mathcal{N}(\boldsymbol{\psi}; b, \sigma_N^2\mathbf{I})$. Consequently, the problem of finding $\mathbf{f}$ and b is equivalent to solving the following optimization problem:

$$\min_{\mathbf{f},b}\ \ \frac{\sigma_N^2}{2}\mathbf{f}^T\mathbf{K}^{-1}\mathbf{f} + \sum_i (y_i - (f_i+b))^2. \tag{6}$$

The kernel hyperparameters are determined by maximizing the marginal likelihood function [Rasmussen and Williams, 2006].

We now describe a GPR based algorithm for semi-supervised classification. Let $N = L+U$, $\mathbf{F} = (\mathbf{f}^T\ \ \mathbf{f}^{*T})^T$ and $\tilde{\mathbf{K}}$ denote the covariance matrix obtained using both the labeled and unlabeled examples. In the case of semi-supervised classification, the formulation (6) can be modified to

$$\begin{array}{ll}\min_{\mathbf{F},b,\mathbf{y}^*} & \frac{\sigma_N^2}{2}\mathbf{F}^T\tilde{\mathbf{K}}^{-1}\mathbf{F} + C_L\sum_{i=1}^{L}\xi_i^2 + C_U\sum_{i=1}^{U}\xi_i^{*2} \\ \text{s.t.} & y_i - (f_i+b) = \xi_i,\ i=1,\ldots,L \\ & y_i^* - (f_i^*+b) = \xi_i^*,\ i=1,\ldots,U \\ & y_i^* \in \{-1,1\}\ \forall\, i, -l \le e^T\mathbf{y}^* \le l \end{array} \tag{7}$$

The above problem is similar to (4) except that it uses the squared error loss instead of the Laplacian loss and uses a different regularizer. Any of these loss functions ensures that the decision boundary passes through the low density region, by penalizing any deviation from $f+b=y$. The squared error loss function has the advantage that it is differentiable everywhere and GP regression formulation can be directly used. Though it is possible to give different weights to labeled and unlabeled examples, we set C_L and C_U to 1 corresponding to standard GP regression. For fixed $\mathbf{y}^*$ and b, the above problem can be solved for $\mathbf{F}$ using standard optimization techniques. Given $\mathbf{f}$ and $\mathbf{f}^*$, $\mathbf{y}^*$ and b can be obtained by solving the following problem:

$$\begin{array}{ll}\min_{\mathbf{y}^*,b} & \sum_{i=1}^{L}\xi_i^2 + \sum_{i=1}^{U}\xi_i^{*2} \\ \text{s.t.} & y_i - (f_i+b) = \xi_i,\ \ i=1,\ldots,L \\ & y_i^* - (f_i^*+b) = \xi_i^*,\ \ i=1,\ldots,U \\ & y_i^* \in \{-1,1\}\ \forall\, i,\ \ -l \le e^T\mathbf{y}^* \le l \end{array} \tag{8}$$

Alternately, one can use the Laplacian loss function and solve the following problem to determine $\mathbf{y}^*$ and b.

$$\begin{array}{ll}\min_{\mathbf{y}^*,b} & \sum_{i=1}^{L}|\xi_i| + \sum_{i=1}^{U}|\xi_i^*| \\ \text{s.t.} & y_i - (f_i+b) = \xi_i,\ \ i=1,\ldots,L \\ & y_i^* - (f_i^*+b) = \xi_i^*,\ \ i=1,\ldots,U \\ & y_i^* \in \{-1,1\}\ \forall\, i,\ \ -l \le e^T\mathbf{y}^* \le l \end{array} \tag{9}$$

Though one can solve the formulation (8), it is simpler to solve the formulation (9). We conducted an experiment to compare the performances of the formulations (8) and (9) and observed that they are close. So, we preferred to solve the formulation (9). The resulting iterative algorithm to design a GPR model for semi-supervised classification is given in Algorithm 2.

Discussion The classifier designed using the proposed SSuGPR algorithm is similar to the probabilistic least squares

Algorithm 2 Semi-Supervised Classification using GPR (SSuGPR)

Initialize σ_N and kernel hyper-parameters.
Train a GP classifier using labeled points and find $\mathbf{y}^*$.
repeat
  Fix $\mathbf{y}^*$, train GPR and predict $\mathbf{f}^*$.
  Calculate the bias b as described in 2.1.
  Set $y_i^* = sgn(f_i^* + b), i = 1, \ldots, U$.
until Algorithm converges or maximum number of iterations are finished

Algorithm 3 Semi-Supervised Classification using sparse GPR (SSuGPS)

Initialize d_{max}, σ_N and kernel hyper-parameters.
Train a GP classifier using labeled points and find $\mathbf{y}^*$.
repeat
  Fix $\mathbf{y}^*$, train IVM and predict $\mathbf{f}^*$.
  Calculate the bias b as described in 2.1.
  Set $y_i^* = sgn(f_i^* + b), i = 1, \ldots, U$.
until Algorithm Converges or maximum number of iterations are finished

classifier [Rasmussen and Williams, 2006]. For a new test sample $\mathbf{x}_*$, the probability that it belongs to class $+1$ can be calculated by first computing the posterior mean f_* and variance σ_*^2. Then, $P(y_* = 1|\mathbf{x}_*) = \Phi(\frac{f_*+b}{\sqrt{1+\sigma_*^2}})$ where $\Phi(\cdot)$ denotes a cumulative Gaussian [Rasmussen and Williams, 2006].

GPR models clearly have advantages over SVR models. Firstly, in addition to prediction, GPR models also give confidence about prediction. Further, the hyperparameters can be adapted directly by maximizing the marginal likelihood, without resorting to cross-validation. Alternatively, they can be adapted by optimizing the predictive distribution based measure, as is done in the probabilistic least squares classifiers [Rasmussen and Williams, 2006].

The computation cost of training a GPR model is $O(N^3)$ which is mainly due to the requirement to invert a matrix. This high cost makes the standard GPs unsuitable for large data sets. Sparse approximate GPR model aims at performing all the operations using a representative data set, called *basis vector set*, from the input space. By fixing the size of this set to d_{max}, it is possible to reduce the training complexity of GPR to $O(Nd_{max}^2)$. Further, the computations of predictive mean and variance can be done using only this representative set, thereby reducing the inference time significantly. There exist several algorithms to design a sparse GP regression model. We used the Informative Vector Machine (IVM) proposed by Lawrence *et al* [2005]. IVM uses a two loop approach for GPR training. The relevant basis vectors are selected in the inner loop using differential entropy score (computed using posterior variance information), while the hyperparameters are adapted by marginal likelihood maximization in the outer loop. IVM is more suitable for large data sets because of inexpensive score evaluation per example in the inner loop.

3.3 Semi-supervised Classification using sparse GPR (SSuGPS)

We now present the algorithm for semi-supervised classification using IVM. Note that any efficient algorithm for sparse GPR design can be used for this purpose.

Unlike the SSuSVR and SSuGPR algorithms, the SSuGPS algorithm results in a sparse model, thereby improving the computational efficiency in terms of both running time and memory requirements. Because of this, the SSuGPS algorithm is scalable and this is demonstrated in the next section.

4 Experiments

In this section, we compare the performances of the proposed SSuGPR and SSuSVR algorithms. Also, the SSuGPS algorithm is compared with the sparse GP algorithm[1] (NCNM) for SSC proposed by Lawrence *et al* [2005]. We performed the experiments on eight real-world data sets for binary classification problem. The data sets are summarized in Table 1. Detailed information about the Waveform, Ionosphere and Letter data sets can be found in UCI repository [Asuncion and Newman, 2007]. The USPS data set was obtained from the site, http://www-i6.informatik.rwth-aachen.de/~keysers/usps.html. The remaining data sets are available at http://ida.first.fraunhofer.de/projects/bench/benchmarks.htm.
For the Splice data set, we only used variables $28 - 34$ instead of original 60 input variables since they are more relevant. For all the experiments, we used Gaussian kernel function defined by $k(\mathbf{x}_i, \mathbf{x}_j) = v \exp\{\frac{-\|\mathbf{x}_i - \mathbf{x}_j\|^2}{2\sigma^2}\}$ where $v, \sigma > 0$. In all the experiments related to the proposed algorithms, C_L and C_U were set to 1, the kernel hyperparameters were initialized using the technique described in [Schölkopf and Smola, 2002] and were adapted by maximizing the marginal likelihood. In the case of SVR based algorithm, C_L and C_U were set to C and the hyperparameters (v, σ, and C) were initialized as suggested in [Zhang *et al.*, 2007]. LIBSVM[2] toolbox was used to train SVR. For the NCNM algorithm, different initial conditions for hyperparameters were tried and the results reported correspond to the best performance.

Data Set	κ	L	U	T
Ionosphere	0.2	36	140	175
Pima	0.2	58	324	384
Splice	0.03	50	450	500
Thyroid	0.25	12	96	107
Waveform	0.2	50	1950	3000
Letter	0.03	10	777	768
Ringnorm	0.03	100	400	500
USPS 3 vs 5	0.1	607		607

Table 1: Summary of the datasets. L, U and T denote the sizes of the labeled, unlabeled and test sets respectively. κ was set based on the difference of proportions of examples belonging to the two classes. l in equation (7) was set to $2\kappa U$.

4.1 Synthetic Data Set

We first consider an illustrative toy problem to demonstrate the decision boundaries obtained using the proposed algo-

[1] The software is available at `http://www.cs.man.ac.uk/~neill/ivm/`.
[2] The software is available at `http://www.csie.ntu.edu.tw/~cjlin/libsvm`.

rithms. The toy 2-dim data set, shown in Figure 1, is a collection of 520 unlabeled examples (small black dots) and 40 labeled examples (solid squares and circles). The decision boundaries obtained using the GP classifier (using only labeled examples) and the SSuGPR algorithm are shown in Figure 1. The plots in Figure 2 demonstrate the decision boundary and contours of the mean of the predictive distribution, obtained using the SSuGPS algorithm. This figure clearly demonstrates how the proposed algorithm utilizes unlabeled data for determining the final decision boundary. Note also that the decision boundary, obtained using the SSuGPS algorithm, passes through the low density region, as desired. However, the generalization performance can deteriorate if the low-density region assumption does not hold.

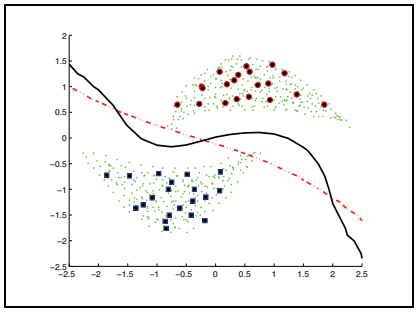

Figure 1: The decision boundaries obtained by the GP Classifier (using only labeled data) and the SSuGPR algorithm for the toy data set are shown by dotted and solid lines respectively.

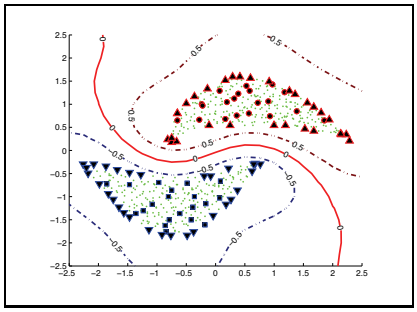

Figure 2: The decision boundary obtained by the SSuGPS algorithm (using $d_{max} = 100$) is shown by a solid line. Contours with prediction values 0.5 and -0.5 are shown by the dashed dotted lines. Labeled examples from class one and two are indicated by dark circles and squares respectively. Unlabeled points are indicated by light circles. $\triangle$ and ∇ show unlabeled points from class one and class two respectively, in the basis vector set.

4.2 Comparison of methods to compute b

To compare the solutions of formulations (8) and (9) we conducted a small experiment on two data sets, Ringnorm and USPS digits "3 vs 5". We ran the SSuGPS algorithm twice, each time using one of the formulations (8) and (9) to compute b and compared the generalization performances. It was observed that the the test set errors (%) for the Ringnorm data set were 5.60 and 6.26, while those for the USPS digits "3 vs 5" data set were 7.15 and 6.86. Since the generalization performances were reasonably close, we preferred to use the formulation (9) in all the experiments, as it is simpler to solve. However, it needs further investigations to compare (8) and (9) in terms of solution simplicity and generalization error.

4.3 Real World Data Sets

All the data sets used in the experiments are fully labeled. So we decided to ignore the labels of some fraction of training data so that the problem can be treated as SSC. In the experiments, each training data set was randomly divided into two parts, labeled and unlabeled. This procedure was repeated for 10 realizations of each data set. d_{max} was set to $.2(L+U)$. The comparative test set error values for different algorithms on various data sets are reported in Table 2. From this table, it is clear that the generalization performance of the SSuGPS algorithm is almost close to that of the SSuGPR algorithm on several data sets. On the data sets like Ionosphere and Splice, the SSuGPS algorithm performed better than the SSuGPR algorithm. This is possibly due to the different set of hyperparameters chosen during training. Note also that these two algorithms performed better than the SSuSVR and NCNM algorithms on majority of the data sets. The SSuSVR algorithm performed better than the other algorithms on the Letter data set while the performances of the NCNM and SSuGPR algorithms were better than the other algorithms on the Pima data set. The high values of standard deviations, observed in some cases, are due to the small sized labeled examples and poor hyperparameter choices made in one or two realizations. The performances of the proposed GP based algorithms were much better than those of the SVM classifiers trained using only labeled data (results not included due to the space constraints), on all the data sets except on the Pima data set.

Data Sets	SSuSVR	SSuGPR	SSuGPS	NCNM
Ionosphere	15.31 ± 3.98	15.20 ± 5.66	$\mathbf{11.03 \pm 4.92}$	19.54 ± 6.11
Pima	30.34 ± 5.05	$\mathbf{28.98 \pm 4.01}$	33.44 ± 5.80	29.45 ± 4.43
Splice	19.32 ± 4.50	18.68 ± 2.87	$\mathbf{17.06 \pm 2.42}$	17.52 ± 3.08
Thyroid	21.03 ± 10.78	16.17 ± 7.72	$\mathbf{15.42 \pm 7.76}$	19.35 ± 4.36
Waveform	19.67 ± 4.47	$\mathbf{12.82 \pm 1.75}$	13.19 ± 1.91	15.30 ± 2.70
Letter	$\mathbf{6.25 \pm 1.76}$	7.88 ± 6.76	8.94 ± 5.75	6.68 ± 2.03
Ringnorm	9.88 ± 5.89	$\mathbf{5.40 \pm 1.74}$	6.26 ± 1.70	17.72 ± 5.40

Table 2: Comparison of the proposed algorithms with the NCNM algorithm on different datasets. Test set errors (%) reported are the averages over 10 trials.

To study the effect of the size of the labeled data on the generalization performance of the classifiers designed using different algorithms, we conducted one experiment on the USPS digits "3 vs 5" data set. The fraction, r, of labeled examples was varied between 0.01 and 0.1. The results are summarized in Table 3. It is clear from the table that the GP based algorithms performed poorly compared to the SSuSVR algorithm when the fraction of labeled examples is small. As the fraction of labeled examples increased, the performances of the GP based algorithms were close to that of the SSuSVR algorithm. Note also that the performances of the SSuGPS and NCNM algorithms are comparable.

4.4 Large Data Set Experiment

To compare the generalization performances of the SSuGPS algorithm with the NCNM algorithm on a large data set, we

r	SSuSVR	SSuGPR	SSuGPS	NCNM
0.01	$\mathbf{9.77 \pm 2.95}$	11.78 ± 3.44	15.40 ± 4.84	19.69 ± 4.68
0.025	$\mathbf{5.07 \pm 0.86}$	5.52 ± 1.31	6.56 ± 1.58	7.15 ± 2.76
0.05	4.48 ± 1.40	$\mathbf{4.32 \pm 0.80}$	5.85 ± 1.33	4.91 ± 0.88
0.1	3.10 ± 0.84	$\mathbf{2.95 \pm 0.77}$	3.51 ± 0.71	3.36 ± 0.94

Table 3: Test set errors (%) on the USPS (digits 3 vs 5) data set.

conducted an experiment on the USPS data set. This data set has 7291 training set examples and 2007 test set examples. For the binary classification problem, we considered each digit vs rest. The problem was set up as SSC by using only a fraction r of labeled examples. r was varied between 0.01 and 0.05. Each experiment was run ten times, randomly selecting the labeled data points. The results using the SSuGPS algorithm are reported in Table 4. The results for the NCNM algorithm, reported in [Lawrence *et al.*, 2005], are given in Table 5. In both the cases, d_{max} was fixed to 500. From these tables, it is clear that for small values of r, the SSuGPS algorithm performed better. The performances of the two algorithms were almost close as r increased. Note that in the experiments, we did not use the same partitions of the data as used in [Lawrence *et al.*, 2005]. But we believe that averaging the results over 10 runs gives a good estimate of the performance of the algorithm. From these experiments we can see that the proposed sparse GPR based algorithm is simple and performs better than the NCNM algorithm.

r	0	1	2	3	4
0.01	3.71±1.39	1.07±0.27	5.68±1.34	5.26±1.31	7.23±5.41
0.025	2.28±0.55	0.82±0.12	3.80±0.55	3.49±0.57	4.09±1.02
0.05	1.48±0.28	0.88±0.53	2.83±0.43	2.67±0.32	4.92±3.54
r	5	6	7	8	9
0.01	6.79±2.08	3.19±0.69	3.42±1.28	5.76±1.14	4.28±0.74
0.025	3.97±0.40	2.11±0.66	1.66±0.41	4.40±0.67	3.53±0.78
0.05	3.24±0.52	1.56±0.26	1.69±0.53	4.03±0.48	2.59±0.51

Table 4: Test set errors (%) on the USPS data set using the SSuGPS algorithm

r	0	1	2	3	4
0.01	17.9±0.00	7.99±6.48	9.87±0.00	8.27±0.00	9.97±0.00
0.025	11.4±8.81	0.98±0.10	9.87±0.00	6.51±2.43	9.97±0.00
0.05	1.72±0.21	1.01±0.10	3.66±0.40	5.35±2.67	7.41±3.50
r	5	6	7	8	9
0.01	7.97±0.00	8.47±0.00	7.32±0.00	8.27±0.00	8.82±0.00
0.025	7.97±0.00	8.47±0.00	7.32±0.00	8.27±0.00	8.82±0.00
0.05	7.11±1.94	1.69±0.15	7.32±0.00	7.42±1.89	7.60±2.72

Table 5: Test set errors (%) on the USPS data set using the NCNM algorithm

To get an idea of the CPU times of different algorithms, we used the USPS data set and considered the binary classification problems of digits "0 vs rest" and digits "1 vs rest". The number of labeled examples used was 364. The number of basis vectors was set to 500. The experiments were done using 2.4 GHz dual core AMD CPU based machine with 4 Gb of main memory running Red Hat Linux. For the digits "0 vs rest", the CPU times of the SSuSVR, SSuGPS and NCNM algorithms were 3982, 804 and 593 seconds respectively. For the digits "1 vs rest", the corresponding times were 3610, 774 and 685 seconds respectively. From these experiments, it is clear that the sparse GP based algorithms are faster than the SVR based algorithm. Though, the SSuGPS algorithm is comparatively slower than the NCNM algorithm, its generalization performance is better than that of the NCNM algorithm. This is evident from Tables 4 and 5. Note that we did not do any detailed study of a timing comparison of different algorithms as it was not the aim of this paper.

5 Conclusion

In this paper, we proposed simple and efficient algorithms for semi-supervised classification using SVR and sparse GPR. The algorithms combined the ideas of maximum margin clustering with regression models. The GPR based algorithms have several advantages over the SVR based algorithm. Further, the sparse GPR based algorithm is scalable. Preliminary experiments on various real-world benchmark data sets demonstrated that the proposed sparse GPR based algorithm is a useful alternative to other sparse GP based algorithms for semi-supervised classification.

References

[Asuncion and Newman, 2007] A. Asuncion and D.J. Newman. UCI machine learning repository, University of California, Irvine, School of Information and Computer Sciences, 2007.

[Bennett and Demiriz, 1998] Kristin P. Bennett and Ayhan Demiriz. Semi-supervised support vector machines. In *Proceedings of the 12th Annual Conference on Neural Information Processing Systems*, pages 368–374, Denver, USA, December 1998. MIT Press.

[Lawrence *et al.*, 2005] Neil D. Lawrence, John C. Platt, and Michael I. Jordan. Extensions of the informative vector machine. In *Proceedings of the Sheffield Machine Learning Workshop*, pages 56–87, New York, 2005. Springer.

[Rasmussen and Williams, 2006] Carl Edward Rasmussen and Christopher K. I. Williams. *Gaussian Processes for Machine Learning*. MIT Press, 2006.

[Schölkopf and Smola, 2002] Bernhard Schölkopf and Alexander Smola. *Learning with Kernels*. MIT Press, Cambridge, MA, 2002.

[Sindhwani *et al.*, 2007] Vikas Sindhwani, Wei Chu, and S. Sathiya Keerthi. Semi-supervised gaussian process classifiers. In *Proceedings of the 20th International Joint Conference on Artificial Intelligence*, pages 1059–1064, Hyderabad, India, January 2007.

[Xu *et al.*, 2004] Linli Xu, James Neufeld, Bryce Larson, and Dale Schuurmans. Maximum margin clustering. In *Advances in Neural Information Processing Systems 17*, Vancouver, Canada, December 2004. MIT Press.

[Zhang *et al.*, 2007] Kai Zhang, Ivor W. Tsang, and James T. Kwok. Maximum margin clustering made practical. In *Proceedings of the 24th International Conference on Machine learning*, pages 1119–1126, Corvalis, USA, June 2007. ACM.

[Zhu, 2005] Xiaojin Zhu. Semi-supervised learning literature survey, Computer Sciences TR 1530, University of Wisconsin-Madison, 2005.