Chapter 1

OPTIMIZATION APPROACHES TO SEMI-SUPERVISED LEARNING

Ayhan Demiriz & Kristin P. Bennett
Department of Decision Sciences and Engineering Systems & Department of Mathematical Sciences, Rensselaer Polytechnic Institute. Troy, NY 12180
demira,bennek@rpi.edu

Abstract We examine mathematical models for semi-supervised support vector machines (S^3VM). Given a training set of labeled data and a working set of unlabeled data, S^3VM constructs a support vector machine using both the training and working sets. We use S^3VM to solve the transductive inference problem posed by Vapnik. In transduction, the task is to estimate the value of a classification function at the given points in the working set. This contrasts with inductive inference which estimates the classification function at all possible values. We propose a general S^3VM model that minimizes both the misclassification error and the function capacity based on all the available data. Depending on how poorly-estimated unlabeled data are penalized, different mathematical models result. We examine several practical algorithms for solving these model. The first approach utilizes the S^3VM model for 1-norm linear support vector machines converted to a mixed-integer program (MIP). A global solution of the MIP is found using a commerical integer programming solver. The second approach uses a noncovex quadratic program. Variations of block-coordinate-descent algorithms are used to find local solutions of this problem. Using this MIP within a local learning algorithm produced the best results. Our experimental study on these statistical learning methods indicates that incorporating working data can improve generalization.

1. INTRODUCTION

The focus of this paper is mathematical programming approaches to semi-supervised learning for classification tasks. The main idea of semi-supervised learning is to construct a classifier using both a training set of labeled data and a working set of unlabeled data. If none of the labels are known then the problem becomes clustering. If some of the labels are known, then the

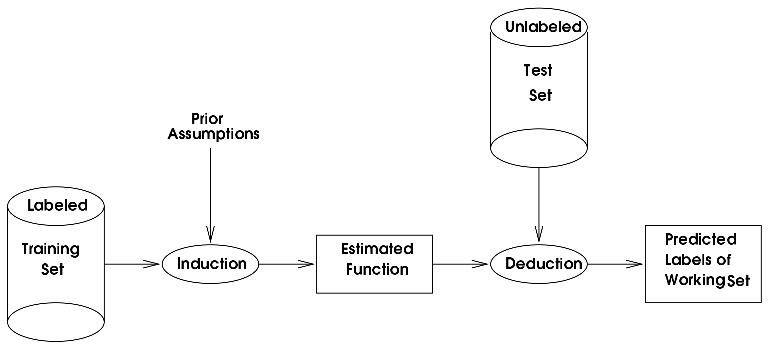

Figure 1.1 Inductive Learning

problem is classification. There are many practical domains in which unlabeled data are abundant but labeled data are expensive to generate and therefore relatively scarce (e.g. medical diagnosis, web search, drug design, and database marketing). When the training data consist of relatively few labeled data points in a high-dimensional space, something must be done to prevent the classification or regression function from overfitting the training data. The key idea is that by exploiting the unlabeled data we hope to be able to provide additional information about the problem that can be used to improve accuracy on data with unknown labels (generalization).

By including the unlabeled data in the testing set, semi-supervised learning can be used to perform transductive learning instead of the more typical inductive learning. In induction, the task is to construct a good discriminant function valid everywhere. This function is fixed and applied to any future test data (Figure 1.1). In transduction, the labeled training data and unlabeled testing data are given, then the discriminant function is constructed based on all the available data. The learning task is to predict the labels of only those specific test data points, not all possible future points. This simpler task can result in theoretically better bounds on the generalization error [30], thus reducing the amount of required labeled data for good generalization (Figure 1.2).

Our semi-supervised support vector machine approach can be illustrated by a simple example. Consider the two-class problem shown in Figure 1.3(a). Since the labeled training sets are linearly separable, there exists an infinite number of possible separating planes that correctly classify the two sets. Intuitively, the best linear classifier is the middle plane shown that separates the two sets with greatest margin. The margin is the sum of distances from the closest points (the support vectors) in each set to the plane or equivalently the distance between the supporting planes for each set. The supporting planes are shown using dotted lines. Statistical Learning Theory proves that for a given misclassification

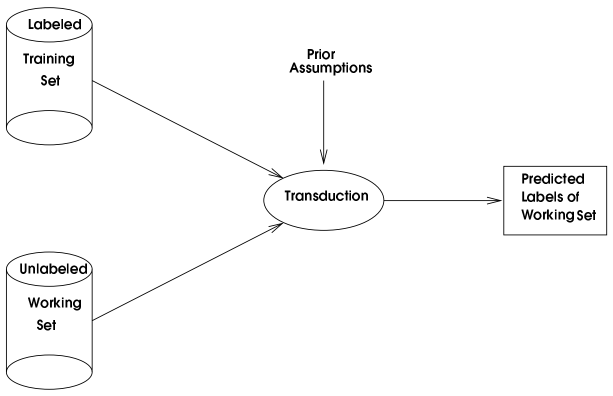

Figure 1.2 Transductive Learning

error, maximizing the margin of separation minimizes a bound on the expected misclassification error on future unseen data [30]. Maximizing the margin reduces the capacity of the function to fit data. Intuitively, a “fat" plane with wide margin has less capacity to fit data than a “skinny" one. In SVM, the optimal plane can be found using quadratic or linear programming depending on the metric used to measure the margin distance [30, 29, 3]. Consider now the additional unlabeled test data shown in Figure 1.3(b). The SVM performs poorly on this particular test set in terms of classification accuracy of the testing data. Note also that the resulting margin for the combined labeled training data and unlabeled testing data is very small. If we construct the SVM margin that correctly classifies the training data and achieves the widest margin based on all the data, the results found by our semi-supervised SVM are significantly improved and the preferable plane is shown in Figure 1.3(c). Results in statistical learning theory show that, for a fixed misclassification error, maximizing the margin based on all the data (train and test) can lead to better bounds on the expected generalization error[30].

For semi-supervised SVM we consider all possible labels of the test data and assign the labels that produce the best SVM with maximum margin based on all the available data, both labeled and unlabeled. For the purpose of this paper we limit our discussion to linear SVM, but these methods can be extended to nonlinear support vector machines using the standard SVM approach of including kernel functions [30, 22]. In Section 2. we review support vector machines. In Section 3. we provide a general framework for viewing the semi-supervised support vector machine problem. Depending on how we penalize unlabeled data appearing in the margin the problem can be formulated as a linear or convex quadratic program with additional equilibrium constraints, mixed-integer constraints, or nonconvex objective terms. In Section 4. we examine

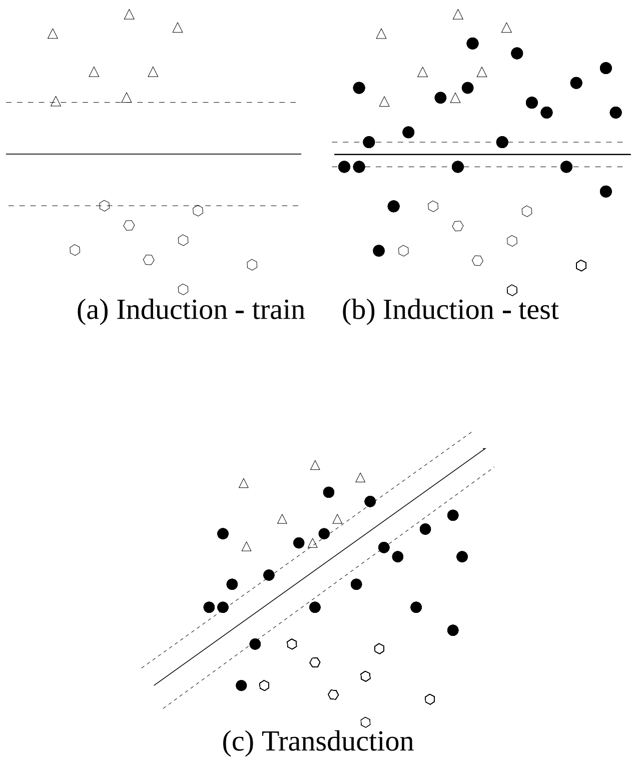

Figure 1.3 Traditional SVM (a, b) versus Semi-Supervised SVM (c)

practical approaches using the linear mixed integer program (MIP) formulation first introduced in [5]. By incorporating the MIP within a local learning approach, performance is greatly enhanced. In Section 5. we examine practical algorithms for a nonconvex quadratic formulation. Finally, we conclude our paper with a brief summary and discussion of optimization issues in semi-supervised learning.

Other researchers have reported favorable results on semi-supervised methods on web-based text classification problems, for example using an EM (Expectation-Maximization) [26, 23], co-training in Bayesian networks [10], and a transductive version of SVM-Light [19]. Cataltepe and Magdon-Ismail [13] propose augmented error, which has components from both labeled and unlabeled data. They provide an analytical solution in the case of linear, noisy targets and linear hypothesis functions. They also show some results for the non-linear case. Theoretical results exist [12] on the relative value of labeled and unlabeled data.

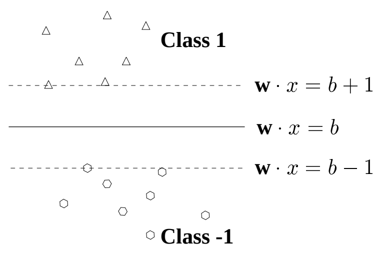

Figure 1.4 Optimal Plane Maximizes Margin

2. REVIEW OF SVM

The underlying problem of interest is to estimate a classification function $f : R^N \rightarrow \{\pm 1\}$ using input-output training data from two classes

$$(x_1, y_1), \ldots, (x_\ell, y_\ell) \in R^n \times \{\pm 1\}. \tag{1.1}$$

The function f should correctly or almost correctly classify unseen examples $(\mathbf{x}, y)$, i.e. $f(\mathbf{x}) = y$ if $(\mathbf{x}, y)$ is generated from the same underlying probability distribution as the training data. In this work we limit discussion to linear classification functions. If the points are linearly separable, then there exist an n-vector $\mathbf{w}$ and scalar b such that

$$\begin{array}{ll} \mathbf{w}\cdot x_i - b \geq 1 & if\ y_i = 1,\ and \\ \mathbf{w}\cdot x_i - b \leq -1 & if\ y_i = -1,\ i = 1, \ldots, \ell \end{array} \tag{1.2}$$

or equivalently

$$y_i[\mathbf{w}\cdot x_i - b] \geq 1,\ i = 1, \ldots, \ell. \tag{1.3}$$

The "optimal" separating plane, $\mathbf{w}\cdot x = b$, is the one that is furthest from the closest points in the two classes. Geometrically this is equivalent to maximizing the separation margin or distance between the two parallel planes $\mathbf{w}\cdot x = b+1$ and $\mathbf{w}\cdot x = b-1$ (see Figure 1.4).

The "margin of separation" in Euclidean distance is $2/\left\|\mathbf{w}\right\|_2$ where $\left\|\mathbf{w}\right\|_2 = \sqrt{\sum_{i=1}^{n} \mathbf{w}_i^2}$ is the 2-norm. To maximize the margin, we minimize $\left\|\mathbf{w}\right\|_2/2$ subject to the constraints (1.3). According to structural risk minimization, for a fixed empirical misclassification rate, larger margins should lead to better generalization and prevent overfitting in high-dimensional attribute spaces[29]. The classifier is called a support vector machine because the solution depends only on the points (called support vectors) located on the two supporting planes $\mathbf{w}\cdot x = b-1$ and $\mathbf{w}\cdot x = b+1$.

In general the classes will not be linearly separable, so the generalized optimal plane problem (1.4) [14, 29] is used. A slack term η_i is added for each point such that if the point is misclassified, $\eta_i \geq 1$. The quadratic programming formulation is (SVM-QP):

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\eta} & C\sum\limits_{i=1}^{\ell}\eta_i + \frac{1}{2}\left\|\mathbf{w}\right\|^2 \\ s.t. & y_i[\mathbf{w}\cdot x_i - b] + \eta_i \geq 1 \\ & \eta_i \geq 0, \;\; i = 1,\ldots,\ell \end{array} \tag{1.4}$$

where $C > 0$ is a fixed penalty parameter. The capacity control provided by the margin maximization can greatly improve generalization [31, 28]. Typically, the following dual form of (1.4) is solved in practice:

$$\begin{array}{ll} \min\limits_{\alpha} & \frac{1}{2}\sum\limits_{i=1}^{\ell}\sum\limits_{j=1}^{\ell} y_i y_j \alpha_i \alpha_j x_i \cdot x_j - \sum\limits_{i=1}^{\ell}\alpha_i \\ s.t. & \sum\limits_{i=1}^{\ell}\alpha_i y_i = 0 \\ & 0 \leq \alpha_i \leq C \quad i = 1,\ldots,\ell \end{array} \tag{1.5}$$

The Robust Linear Programming approach to SVM is identical to SVM-QP except the margin term is changed from the 2-norm $\left\|\mathbf{w}\right\|_2$ to the 1-norm, $\left\|\mathbf{w}\right\|_1 = \sum_{j=1}^{n}|w_j|$. The problem becomes the following robust linear program (SVM-RLP) [6, 11, 4]:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,s,\eta} & C\sum\limits_{i=1}^{\ell}\eta_i + \sum\limits_{j=1}^{n}s_j \\ s.t. & y_i[\mathbf{w}\cdot x_i - b] + \eta_i \geq 1 \\ & \eta_i \geq 0, \;\; i = 1,\ldots,\ell \\ & -s_j <= \mathbf{w}_j <= s_j, \;\; j = 1,\ldots,n. \end{array} \tag{1.6}$$

The RLP formulation is a useful variation of SVM with some nice characteristics. The 1-norm weight reduction still provides capacity control. The results in [21] can be used to show that minimizing $\|\mathbf{w}\|_1$ corresponds to maximizing the separation margin using the infinity norm. Statistical learning theory could potentially be extended to incorporate alternative norms. One major benefit of SVM-RLP over SVM-QP is dimensionality reduction. Both SVM-RLP and SVM-QP minimize the magnitude of the weights **w**. But RLP forces more of the weights to be 0 due to the properties of the 1-norm. This results in dimensionality reduction since variables with 0 weights can be removed from the model. Another benefit of SVM-RLP over SVM-QP is that it can be solved using linear programming instead of quadratic programming.

SVM are easily generalized to nonlinear discriminants through the introduction of kernel functions [30, 22]. The basic idea is that the data are mapped nonlinearly to a higher dimensional space and a linear SVM is constructed in the transformed space corresponding to a nonlinear classifier in the original space. We limit our formulation to the linear classification problem and leave computational studies of these approaches extended with kernels to future work.

3. SEMI-SUPERVISED SVM

The basic idea of semi-supervised support vector machines is that we want the best support-vector machine on the labeled data that has no or very few unlabeled points in the margin. Thus we want to penalize the support vector machine if unlabeled points fall in the margin. Specifically, we define the semi-supervised support vector machine problem (S^{3}VM) as:

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\eta,\xi,z} & C\left[\sum_{i=1}^{\ell}\eta_i + \sum_{j=\ell+1}^{\ell+k} g(\mathbf{w}\cdot x_j - b)\right] + \parallel \mathbf{w} \parallel \\ s.t. & y_i(\mathbf{w}\cdot x_i - b) + \eta_i \geq 1 \quad \eta_i \geq 0 \;\; i = 1,\ldots,\ell \end{array} \tag{1.7}$$

where $C > 0$ is a fixed misclassification penalty parameter and $g(\alpha)$ is the margin penalty function on unlabeled data.

The question then is how to define g. For a hard margin approach in which no unlabeled points are allowed in the margin, the margin penalty function is defined as

$$\begin{array}{rll} g_\infty(\alpha) := & \infty & for\ -1 < \alpha < 1 \\ & 0 & otherwise \end{array} \tag{1.8}$$

If an unlabeled point falls outside the margin, it is considered well-classified and no penalty is incurred.

We can transform the hard margin g_∞ problem into a linear or quadratic program with an additional equilibrium constraint. We start with either SVM formulation, (1.4) or (1.6), and then add two constraints for each point in the working set. One constraint calculates the misclassification error as if the point were in class 1 and the other constraint calculates the misclassification error as if the point were in class -1. We add a constraint that forces one of the two misclassification errors per point to be zero. This produces the following mathematical programming problem with equilibrium constraints:

$$\begin{array}{lrl} \min\limits_{\mathbf{w},b,\eta,\xi,z} & C\left[\sum_{i=1}^{\ell}\eta_i\right] + \parallel \mathbf{w} \parallel & \\ s.t. & y_i(\mathbf{w}\cdot x_i - b) + \eta_i \geq 1 & \eta_i \geq 0 \;\; i = 1,\ldots,\ell \\ & \mathbf{w}\cdot x_j - b + \xi_j \geq 1 & \xi_j \geq 0 \;\; j = \ell+1,\ldots,\ell+k \\ & -(\mathbf{w}\cdot x_j - b) + z_j \geq 1 & z_j \geq 0 \\ & \xi_j \cdot z_j = 0 & j = \ell+1,\ldots,\ell+k \end{array} \tag{1.9}$$

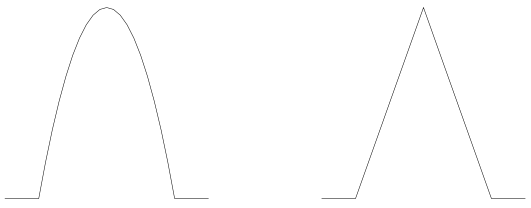
(a) $g_2(\gamma)$ (b) $g_1(\gamma)$

Figure 1.5 Margin Penalty Functions

The requirement that no unlabeled points may fall in the margin may be too strong. A natural relaxation of the problem would be to move the equilibrium constraint into the objective and use it as the margin penalty function g. This results in the following nonconvex quadratic optimization problem:

$$\begin{array}{lll} \min\limits_{\mathbf{w},b,\eta,\xi,z} & C\left[\sum\limits_{i=1}^{\ell}\eta_i + \sum\limits_{j=\ell+1}^{\ell+k}\xi_j \cdot z_j\right] + \parallel \mathbf{w} \parallel & \\ s.t. & y_i(\mathbf{w}\cdot x_i + b) + \eta_i \geq 1 \quad \eta_i \geq 0 \;\; i = 1,\ldots,\ell & \\ & \mathbf{w}\cdot x_j - b + \xi_j \geq 1 \quad \xi_j \geq 0 \;\; j = \ell+1,\ldots,\ell+k & \\ & -(\mathbf{w}\cdot x_j - b) + z_j \geq 1 \quad z_j \geq 0 & \end{array} \tag{1.10}$$

Close examination of this choice of error function shows that it has attractive properties. If the unlabeled point x_j falls outside or on the margin then ξ_j or z_j is 0, and there is no error associated with that point. If the point falls in the margin, then for $\gamma = w \cdot x_j - b$, $\xi_j = 1 - \gamma$ and $z_j = 1 + \gamma$ by construction of the support vector machine. The following piecewise quadratic margin penalty function is produced (see Figure 1.5(a)):

$$g_2(\gamma) := \begin{array}{ll} 1-\gamma^2 & for\ -1 < \gamma < 1 \\ 0 & otherwise. \end{array} \tag{1.11}$$

Another natural choice would be a margin penalty function that calculates the minimum of the two possible misclassification errors. The final class of a point corresponds to the one that results in the smallest error. This is the transductive idea as proposed by Vapnik [30]. It has the advantage that if the correct labels are found, the resulting SVM will be identical to the one produced if the points were known. The minimum error formulation is [5]:

$$\begin{array}{lll} \min\limits_{\mathbf{w},b,\eta,\xi,z} & C\left[\sum\limits_{i=1}^{\ell}\eta_i + \sum\limits_{j=\ell+1}^{\ell+k}\min(\xi_j, z_j)\right] + \|\,\mathbf{w}\,\| & \\ s.t. & y_i(\mathbf{w}\cdot x_i + b) + \eta_i \geq 1 \quad \eta_i \geq 0 \;\; i = 1,\ldots,\ell & \\ & \mathbf{w}\cdot x_j - b + \xi_j \geq 1 \quad \xi_j \geq 0 \;\; j = \ell+1,\ldots,\ell+k & \\ & -(\mathbf{w}\cdot x_j - b) + z_j \geq 1 \quad z_j \geq 0 & \end{array} \tag{1.12}$$

The resulting margin penalty function is shown in Figure 1.5(b)

$$g(\gamma) = g_1(\gamma) := \begin{array}{ll} 1 - |\gamma| & for \; -1 < \gamma < 1 \\ 0 & otherwise \end{array} \tag{1.13}$$

For our experimental study of practical methods for solving these problems we focused on the minimum error formulation (1.12). But this is not to say that other formulations are not possible or preferable. In the next two sections we explore two different approaches to practically solving this problem.

4. MIXED-INTEGER PROGRAMMING FORMULATION

Integer programming can be used to exactly solve S^{3}VM (1.12). The basic idea is to add a 0 or 1 decision variable, d_j, for each point $\mathbf{x}_j$ in the working set. This variable indicates the class of the point. If $d_j = 1$ then the point is in class 1 and if $d_j = 0$ then the point is in class -1. This results in the following mixed integer program (S^{3}VM-MIP):

$$\begin{array}{ll} \min\limits_{\mathbf{w},b,\eta,\xi,z,d} & C\left[\sum\limits_{i=1}^{\ell}\eta_i + \sum\limits_{j=\ell+1}^{\ell+k}(\xi_j + z_j)\right] + \|\,\mathbf{w}\,\| \\ s.t. & y_i(\mathbf{w}\cdot x_i - b) + \eta_i \geq 1 \quad \eta_i \geq 0 \;\; i = 1,\ldots,\ell \\ & \mathbf{w}\cdot x_j - b + \xi_j + M(1-d_j) \geq 1 \quad \xi_j \geq 0 \;\; j = \ell+1,\ldots,\ell+k \\ & -(\mathbf{w}\cdot x_j - b) + z_j + Md_j \geq 1 \quad z_j \geq 0 \;\; d_j = \{0,1\} \end{array} \tag{1.14}$$

The constant $M > 0$ is chosen sufficiently large such that if $d_j = 0$ then $\xi_j = 0$ is feasible for any optimal **w** and b. Likewise if $d_j = 1$ then $z_j = 0$. In this paper we use the 1-norm of **w** in the objective. A globally optimal solution to this problem can be found using CPLEX or other commercial mixed integer programming codes [15] provided computer resources are sufficient for the problem size. Using the mathematical programming modeling language AMPL [16], we were able to express the problem in approximately thirty lines of code plus a data file and solve it using CPLEX. One practical limitation of this approach is the capacity of the MIP solver used. Using CPLEX 4.0 on a Sun Ultra 1 with 700MB RAM we found it was practical to include about 50 unlabeled data points due to the CPU time limitation.

4.1 LOCAL SEMI-SUPERVISED SUPPORT VECTOR MACHINES

To get around the practical restriction on the number of integer variables and thus unlabeled data handled by our MIP solver, we utilized the S^{3}VM-MIP as part of a local learning algorithm. In local learning, a point is classified based on points in its "neighborhood". For example, in the K-Nearest-Neighbor algorithm (K-NN), the K nearest neighbors to a point (by Euclidean distance or some other metric) are found and then the point is assigned the majority class of the K nearest neighbors. Local learning methods are often called memory-based methods, because training examples are kept in "memory" and used to classify new points. Since the local models have fewer training examples, it takes much less computational time to optimize each local S^{3}VM than to train one global one at the expense of many local models. Previous empirical studies have shown that the generalization ability of local methods often exceeds that of global ones since the local models include only the points which are related to the query point (interested unlabeled data) in a given learning task. Many variations exist for both selecting the neighborhoods and determining the output class based on the neighbor. For example, Discriminant Adaptive Nearest Neighbor [17] uses local discriminant analysis to estimate the class within K-NN classification. Lawrence et al. [20] use local neural network models for function approximation. See [1] for a survey of approaches.

4.2 LOCAL S^{3}VM AND EXPERIMENTAL RESULTS

Local S^{3}VM is nothing but an application of S^{3}VM in a local neighborhood of each unlabeled point as determined by the K-NN algorithm using Euclidean distance. This neighborhood includes both labeled and unlabeled examples. In order to have enough labeled examples in each neighborhood, we arbitrarily pick K as 10% of all available data points. Further study is needed on how to best select the neighborhood of a point. We can summarize the method (Local-S^{3}VM) to classify a given unlabeled point in the following steps:

1. Find K-NN for a given unlabeled point.
2. If all the labeled points in the neighborhood are in one class, then label the unlabeled point as in that class and end. Otherwise continue.
3. Solve the S^{3}VM-MIP (1.14) in the neighborhood.
4. Label the point according to the result of S^{3}VM .

There are many advantages to using Local S^{3}VM over using a single global S^{3}VM. In transduction for any data, we need to construct a new model. So the fact that local S^{3}VM must compute a new model for each point is also

Table 1.1 **Dataset Summary Statistics**

Data Set	Dim	Points	Test-size
Bright	14	2462	50*
Cancer	9	699	70
Diagnostic	30	569	57
Dim	14	4192	50*
Heart	13	297	30
Housing	13	506	51
Ionosphere	34	351	35
Musk	166	476	48
Sonar	60	208	21
Pima	8	769	50*

true for any transductive algorithm. Although there are as many models as unlabeled points to solve in Local S^{3}VM , the overall computational time of the algorithm including time to find the local neighborhood is generally less than the global S^{3}VM algorithm. This is because we have fewer unlabeled points in each local model which means we have fewer binary variables in each model. Having fewer binary variables results in less running time for each local model. Another advantage is that the overall classification function by Local S^{3}VM is nonlinear (piecewise linear to be exact) when a linear S^{3}VM is used locally.

Determining nearest neighbors of a point can become problematic for large datasets. One must consider an appropriate metric and method to find K-NN. Since we use datasets which have relatively small dimensions, we use Euclidean distance combined with a partial sort algorithm [25] to find the local neighborhood. As mentioned in the outlines of the algorithm, for each unlabeled point, a related data file is created and the S^{3}VM model is solved using AMPL. Then the output of AMPL is analyzed to find the label of the point.

Our computational study of S^{3}VM consisted of 10 trials using the ten real-world data sets described in Table 1.1 (eight from [24] and the bright and dim galaxy sets from [27]) [1]. The basic properties of the datasets are summarized in Table 1.1. Each dataset is sampled randomly 10 times and each working set is composed of 10% of the data except the Bright, Dim, and Pima datasets in which the size of the working set is set to 50 points and rest of the data are used as the training set. We use the following formula to pick the penalty parameter:

[1]The continuous response variable in Housing dataset was categorized at 21.5

Table 1.2 **Average Error Results for Inductive and Transductive SVM Methods**

Data Set	SVM-RLP	S^3VM	Local SVM	Local S^3VM	3-NN
Bright	0.02	0.018	0.008	<u>0.006</u>	0.028
Cancer	0.036	<u>0.034</u>	0.06	0.059	<u>0.034</u>
Diagnostic	0.035	<u>0.033</u>	0.039	0.039	0.039
Dim	0.064	0.054	<u>0.042</u>	0.044	0.074
Heart	0.173	<u>0.16</u>	0.257	0.253	0.17
Housing	0.155	0.151	<u>0.118</u>	0.124	0.177
Ionosphere	0.109	<u>0.106</u>	0.117	0.109	0.129
Musk	0.173	0.173	0.092	<u>0.085</u>	0.208
Sonar	0.281	0.219	0.181	<u>0.143</u>	0.171
Pima	0.22	0.222	0.22	<u>0.218</u>	0.264

$C = \frac{(1-\lambda)}{\lambda(\ell+k)}$ with $\lambda = 0.001$, ℓ is the size of the training set, and k is the size of the working set. The average working set errors are reported in Table 1.2. The best result from different models is underlined for each dataset.

Columns two and three of Table 1.2 provide a comparison of the inductive linear 1-norm support vector machine (SVM-RLP 1.6) with the transductive linear 1-norm SVM optimized used mixed integer programming (S^{3}VM-MIP 1.14). On all ten datasets, the transductive S^{3}VM-MIP results are either slightly better or not significantly different than the inductive results found using SVM-RLP. Note that all parameters of the formulations are identical; the only difference between the two formulations is the use of unlabeled data for the transductive case. For this formulation, unlabeled data seems to help and never hurt generalization.

Columns 4 and 5 of Table 1.2 compare an inductive version of Local SVM and the transductive version of Local S^{3}VM . In our study, the neighborhoods of points used by both Local SVM and Local S^{3}VM are identical. Thus for each testing set point the optimization problem solved by local S^{3}VM is identical to the one solved by local SVM once the terms involving the unlabeled data are removed. This was done to ensure that the introduction of unlabeled data was the only change in the experiment. But in fact, it means the unlabeled data are being used to determine the effective size of the neighborhood for Local SVM which in itself is a form of transduction. Column 6 of Table 1.2 gives results for the 3-nearest neighbor algorithm. This was done to examine improvements that occur by simply switching to a local algorithm. Local S^{3}VM outperformed or did as well as Local SVM on eight of the ten datasets, once again supporting the transductive hypothesis. The improvements cannot be simply attributed to a local learning strategy since 3-NN did worse than both Local SVM and S^{3}VM on nine of ten datasets.

Overall, Local S^{3}VM was consistently the best or almost the best in our experiments. Either S^{3}VM or the Local S^{3}VM obtained the best results on most of the datasets except Dim and Housing datasets. The results indicate that using the labeled and unlabeled points in a transduction model can improve accuracy. Local S^{3}VM resulted in better accuracy than S^{3}VM on six datasets. One noteworthy point is that in some cases (Sonar, Musk, Housing, Bright) Local S^{3}VM improved accuracy notably. On Cancer, Diagnostic, Heart, and Ionosphere the fact that S^{3}VM performed best indicates that if the neighborhood of Local S^{3}VM is increased, Local S^{3}VM could perform better. The best method of choosing neighborhoods for local methods is still very much an open question. The proposed algorithm in this section takes into consideration only one unlabeled point at a given time. Although there might be many unlabeled points in a given neighborhood, the algorithm returns the results only on the test point of interest. The results for other points are basically discarded. One extension would be keeping these results for a final vote at the end of the algorithm. In this case, we can assign a probability of class membership for a certain point. The results from one point can also be used as starting points to improve the solution time of Local S^{3}VM on nearby points.

5. NONCONVEX QUADRATIC APPROACH

An alternative approach to solving the minimum error S^{3}VM problem (1.12) is to convert it into a nonconvex quadratic program. We adapt the approach used previously to handle disjunctiveness of classification labels within the bilinear separability [7] and global tree optimization problems [9, 2, 7]. Once again a decision variable d_j is introduced for each point such that at optimality if $d_j = 1$ then the predicted class of x_j is 1 and if $d_j = 0$ then the x_j is predicted as class -1. The resulting problem is (S^{3}VM-QP)

$$\begin{array}{ll}
\min\limits_{\mathbf{w},b,\eta,\xi,z,d} & C\left[\sum\limits_{i=1}^{\ell}\eta_i + \sum\limits_{j=\ell+1}^{\ell+k}(d_j\xi_j + (1-d_j)z_j)\right] + \|\,\mathbf{w}\,\| \\
s.t. & y_i(\mathbf{w}\cdot x_i - b) + \eta_i \geq 1 \quad \eta_i \geq 0 \;\; i = 1,\ldots,\ell \\
& \mathbf{w}\cdot x_j - b + \xi_j \geq 1 \quad \xi_j \geq 0 \;\; j = \ell+1,\ldots,\ell+k \\
& -(\mathbf{w}\cdot x_j - b) + z_j \geq 1 \quad z_j \geq 0 \;\; 0 \leq d_j \leq 1
\end{array} \tag{1.15}$$

An intuitively simple approach is to adapt a block coordinate descent algorithm (e.g. [8]) which alternates between fixing d and estimating the SVM weights $\mathbf{w}, b$ and other dependent variables, and optimizing d with the other SVM variables fixed. In [9], it was shown for a class of problems that includes S^{3}VM-QP (1.15), using 2-norm $\|\,\mathbf{w}\,\|_2$ such an approach will converge in a finite number of iterations to a solution satisfying the minimum principal necessary optimality conditions. No linesearch is required. The proof in [9] does require each subproblem be solved to optimality, but this condition can be relaxed to require only a strict decrease in the objective function. On the global tree optimization problem [2, 9] , the block coordinate descent algorithm was found to be very prone to local minima so a tabu search method was used. When applied to transduction, we also found this simple algorithm to be very prone to local minima and thus do not report the results here. To improve the results, we developed a heuristic variation of the block coordinate descent algorithm. We introduce this algorithm in the following section.

5.1 A DESCENT ALGORITHM FOR TRANSDUCTIVE SVM

The essential idea behind our heuristic approach is that we start by heavily penalizing solutions with points falling within the margin and then relax this requirement in order to find solutions with wider margin. Just as in the basic block coordinate descent method, we first estimate the labels $(d_j,\ j = \ell + 1, \ldots, \ell + k)$ based on our current estimate of the SVM, and then solve S^{3}VM-QP with d fixed. Note that in practice and for easy introduction of nonlinearity via kernels we solve the dual of Problem (1.15) which for fixed d reduces to the usual dual SVM problem (Eq. 1.5) tailored for transduction :

$$
\begin{array}{ll}
\min\limits_{\alpha} & \frac{1}{2}\sum\limits_{i=1}^{\ell+k}\sum\limits_{j=1}^{\ell+k} y_i y_j \alpha_i \alpha_j K(x_i, x_j) - \sum\limits_{i=1}^{\ell+k} \alpha_i \\
s.t. & \sum\limits_{i=1}^{\ell+k} \alpha_i y_i = 0 \\
 & 0 \le \alpha_i \le C \quad i = 1, \ldots, \ell + k
\end{array}
\tag{1.16}
$$

where $y_j = 2 * (d_j - \frac{1}{2})$ for $j = \ell, \ldots, \ell + k$. This process is repeated until a local minimum is reached. Then the weight on the misclassification error C is decreased allowing wider margins. In order to escape from local minima, the algorithm switches the labels of unlabeled data close to the separating hyperplane, if necessary. For this purpose, we check the consecutive solutions to track local minima. If 10 consecutive solutions are the same we assign the opposite labels to the points satisfying $|\mathbf{w} \cdot x_{i+1} - b| < S$. Occassionally a local minima is found with all points classified in one class ($\mathbf{w} = 0$). In this case, the algorithm restarts using the same initial conditions except for a reduced margin penalty parameter C for the unlabeled data. We empirically picked $C = \frac{\lambda}{100*(1-\lambda)}$, because it performed well in most cases. To ensure a good starting solution, the initial label assignments are made based on the closest class center for each unlabeled point. The resulting algorithm can be summarized as follows:

Algorithm 5..1 *S^{3}VM-IQP*

```
■ Find class centers from training points
■ Assign labels $d_0$ to working set according to the closest class center
■ Initialization: $i = 0$, $\lambda = 0.9$, $C = \frac{\lambda}{100*(1-\lambda)}$, $counter = 0$, $S = 0.2$.
■ While $i \leq max\_iteration$
    1. Fix $d_i$ and solve Problem 1.15 (or its dual (1.16)) to find ($\mathbf{w}_{i+1}, b_{i+1}, \eta_{i+1}$,
       $\xi_{i+1}, z_{i+1}$).
    2. Fix ($\mathbf{w}_{i+1}, b_{i+1}, \eta_{i+1}, \xi_{i+1}, z_{i+1}$) and solve Problem 1.15 for $d_{i+1}$.
    3. Check convergence criteria
        – If solution is same as the previous one
          then counter=counter +1 and $\lambda = \lambda * 0.9$
          else if there exists no point within margin
          then stop
          else let $counter = 0$
        – if $counter > 10$ then let $counter = 0$ and assign the opposite
          labels to the points satisfying $|\mathbf{w}_{i+1} \cdot x - b_{i+1}| < S$
        – if solution is all-in-one-class then reassign initial conditions
          except $i$ and let $\lambda* = 0.9$
    4. $i = i + 1$
```

As a benchmark for transductive SVM, we report results from SVM-Light proposed by Joachims in [19, 18]. Transductive SVM-Light also can be viewed as a block coordinate descent algorithm that alternates between estimating the class labels and optimizing the SVM based on those labels. Transductive SVM-Light has an inner and an outer loop. The outer loop adjusts the penalty parameters on misclassification errors. Different errors are used for the unlabeled data according to their estimated class labels. After initial inductive iteration, the algorithm starts with low penalty terms for unlabeled data. Two penalty terms (C^*_-, C^*_+) are used in transductive SVM-Light, each for classifying an unlabeled point as a class -1 or a class 1 object respectively. Then it uniformly increases the influence of unlabeled data up to a user-defined penalty level. During this phase, the algorithm tunes these penalty terms in a way to satisfy a user-defined bias in data. The inner loop optimizes the SVM for the given penalties. The inner loop switches the labels of two given points, if such an action reduces the overall error. Like S^{3}VM-IQP, SVM-Light alternates

Table 1.3 **Average Error Results for Transductive and Inductive Methods**

Data Set	SVM-QP	SVM-Light	S^{3}VM-IQP
Heart	$\underline{0.16}$	0.163	0.1966
Housing	0.1804	$\underline{0.1608}$	0.1647
Ionosphere	$\underline{0.0857}$	0.1572	0.0943
Sonar	0.1762	0.2524	$\underline{0.1572}$

the labels to avoid local minima. The primary difference is that SVM-Light changes the signs of at most two points at a time. Another difference is SVM-Light uses different margin penalty parameters for class 1 and class -1 objects. In addition, unlike S^{3}VM-QP, it starts with lower values for margin penalty parameters. Details of SVM-Light and successful results on large datasets can be found in [19]. We use the default parameter options in our experiments with SVM-Light.

5.2 S^{3}VM-IQP RESULTS

In this section we compare S^{3}VM-IQP with SVM-QP (Eq. 1.5) and transductive SVM-Light. We use same datasets as in the previous section. Due to the long computational times for S^{3}VM-IQP and transductive SVM-Light, we limit our experiments to only the Heart, Housing, Ionosphere, and Sonar datasets. Linear kernel functions are used for all methods used in this section. The results given in Table 1.3 show that using unlabeled data in the case of datasets Heart and Ionosphere affects generalization ability slightly but the difference between the best transductive result and SVM-QP (Eq. 1.5) is not statistically significant. In the other two cases (Housing and Sonar), the best transductive method outperforms SVM-QP significantly. On two datasets S^{3}VM-IQP performs significantly better than transductive SVM-Light and in one case (Housing) the difference between two methods is not statistically significant.

As indicated above, the results from both S^{3}VM-IQP and SVM-Light are inconclusive. Both algorithms are much more expensive than their inductive versions. From the results on the Mixed Integer Programming Approaches we know that transduction can improve learning. We speculate that the reason that these improvements were not found using S^{3}VM-IQP and SVM-Light is that the optimization problem is very difficult and that the methods are failing to find the global minima. We know from the prior experiments that there is very little room for improvement on these specific learning tasks. Very few local minima will lead to better generalization. S^{3}VM -MIP and its local version are finding globally optimal solutions that are better. From the results on SVM-

Light reported in [19] we know that on larger problems in text categorization, transductive inference using SVM-Light did lead to significant improvements. So on different learning tasks S^{3}VM-IQP may perform better as well. We speculate that on problems where there are many local minima that improve generalization, it is not as essential that the global minimum be found. Further studies are needed to identify when methods that find good but not globally optimial solutions are sufficient. Note that nonlinear kernels also might result in better generalization.

6. CONCLUSION

We examined mathematical models for semi-supervised support vector machines (S^{3}VM). We proposed a general S^{3}VM model that minimizes both the misclassification error and the function capacity based on all the available data. Three different functions for penalizing unlabeled points falling in the margin were discussed. Our computational investigation focused on the minimum error formulation for the transductive inference problem. We converted this problem to a mixed-integer program that can be exactly solved using commercial integer programming packages. By using the MIP formulation with a local learning algorithm, a powerful scalable transductive inference method was created. Our computational experiments found that the local learning method was the most effective overall. Further studies are needed to determine how to best select neighborhoods and to choose the parameters within the local S^{3}VM-MIP. In addition, very efficient computational methods for the local S^{3}VM-MIP are needed. One possibility is to use the estimated labels and models for one point as a starting point for other points. We also examined a noncovex quadratic optimization approach to S^{3}VM. Our computational studies were less conclusive using this approach. The best optimization approach for solving this problem is still very much an open question.

Acknowledgments

This work was partially supported by NSF IRI-9702306, NSF IIS-9979860 and NSF DMS-9872019. Thanks to the referees and Scott Vandenberg for helpful comments.

References

[1] C. G. Atkeson, A. W. Moore, and S. Schaal. Locally weighted learning. *Artificial Intelligence Review*, 11:11–73, 1997.

[2] K. P. Bennett. Global tree optimizaiton:a non-greedy decision tree algorithm. *Computing Science and Statistics*, 26:156–160, 1994.

[3] K. P. Bennett. Combining support vector and mathematical programming methods for classification. In B. Schölkopf, C. Burges, and A. Smola,

editors, *Advances in Kernel Methods – Support Vector Machines*, pages 307–326, Cambridge, MA, 1999. MIT Press.

[4] K. P. Bennett and E. J. Bredensteiner. Geometry in learning. Web manuscript, Rensselaer Polytechnic Institute, http://www.rpi.edu/∼bennek/geometry2.ps, 1996. Accepted for publication in Geometry at Work, C. Gorini et al, editors, MAA Press.

[5] K. P. Bennett and A. Demiriz. Semi-supervised support vector machines. In D. Cohn M. Kearns, S. Solla, editor, *Advances in Neural Information Processing Systems*, pages 368–374, Cambridge, MA, 1999. MIT Press.

[6] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable sets. *Optimization Methods and Software*, 1:23–34, 1992.

[7] K. P. Bennett and O. L. Mangasarian. Bilinear separation in n-space. *Computational Optimization and Applications*, 4(4):207–227, 1993.

[8] D. P. Berstsekas. *Nonlinear Programming*. Aethena Scientific, Cambridge, MA, 1996.

[9] J. Blue. *A hybrid of tabu search and local descent algorithms with applications in artificial intelligence*. PhD thesis, Rensselaer Polytechnic Institute, Troy, NY, 1998.

[10] A. Blum and T. Mitchell. Combining labeled and unlabeled data with co-training. In *Proceedings of the 1998 Conference on Computational Learning Theory*, Madison WI, 1998. ACM Inc.

[11] E. J. Bredensteiner and K. P. Bennett. Feature minimization within decision trees. *Computational Optimization and Applications*, 10:110–126, 1997.

[12] V. Castelli and T. M. Cover. On the exponential value of labeled samples. *Pattern Recognition Letters*, 16:105–111, 1995.

[13] Z. Cataltepe and M. Magdon-Ismail. Incorporating test inputs into learning. In *Proceedings of the Advances in Neural Information Processing Systems, 10*, Cambridge, MA, 1997. MIT Press.

[14] C. Cortes and V. N. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[15] CPLEX Optimization Incorporated, Incline Village, Nevada. *Using the CPLEX Callable Library*, 1994.

[16] R. Fourer, D. Gay, and B. Kernighan. *AMPL A Modeling Language for Mathematical Programming*. Boyd and Frazer, Danvers, MA, 1993.

[17] T. Hastie and R. Tibshirani. Discriminant adaptive nearest neighbor classification. *IEEE PAMI*, 18:607–616, 1996.

[18] T. Joachims. Text categorization with support vector machines: Learning with many relevant features. In *European Conference on Machine Learning(ECML)*, 1998.

[19] T. Joachims. Transductive inference for text classification using support vector machines. In *International Conference on Machine Learning*, 1999.

[20] S. Lawrence, A. C. Tsoi, and A. D. Back. Function approximation with neural networks and local methods: Bias, variance and smoothness. In Peter Bartlett, Anthony Burkitt, and Robert Williamson, editors, *Australian Conference on Neural Networks, ACNN 96*, pages 16–21. Australian National University, 1996.

[21] O. L. Mangasarian. Arbitrary norm separating plane. *Operations Research Letters*, 24(1-2), 1999.

[22] O. L. Mangasarian. Generalized support vector machines. In A. Smola, P. Bartlett, B. Schölkopf, and D. Schuurmans, editors, *Advances in Large Margin Classifiers*, pages 135–146, Cambridge, MA, 2000. MIT Press. ftp://ftp.cs.wisc.edu/math-prog/tech-reports/98-14.ps.

[23] A. McCallum and K. Nigam. Employing em and pool-based active learning for text classification. In *Proceedings of the 15th International Conference on Machine Learning (ICML-98)*, 1998.

[24] P.M. Murphy and D.W. Aha. *UCI repository of machine learning databases*. Department of Information and Computer Science, University of California, Irvine, California, 1992.

[25] D. R. Musser and A. Saini. *STL Tutorial and Reference Guide: C++ Programming with the Standard Template Library*. Addison-Wesley, 1996.

[26] K. Nigam, A. McCallum, S. Thrun, and T. Mitchell. Learning to classify text from labeled and unlabeled documents. In *Proceedings of the 15th National Conference on Artificial Intelligence (AAAI-98)*, 1998.

[27] S. Odewahn, E. Stockwell, R. Pennington, R Humphreys, and W Zumach. Automated star/galaxy discrimination with neural networks. *Astronomical Journal*, 103(1):318–331, 1992.

[28] V. N. Vapnik. *Estimation of dependencies based on empirical Data*. Springer, New York, 1982. English translation, Russian version 1979.

[29] V. N. Vapnik. *The Nature of Statistical Learning Theory*. Springer Verlag, New York, 1995.

[30] V. N. Vapnik. *Statistical Learning Theory*. Wiley Inter-Science, 1998.

[31] V. N. Vapnik and A. Ja. Chervonenkis. *Theory of Pattern Recognition*. Nauka, Moscow, 1974. In Russian.