An Iterative SVM Approach to Feature Selection and Classification in High-Dimensional Datasets

Dehua Liu\textsuperscript{1,1}, Hui Qian\textsuperscript{1,1,*}, Guang Dai\textsuperscript{1,1}, Zhihua Zhang\textsuperscript{1,1}

College of Computer Science & Technology
Zhejiang University
Hangzhou, China 310027

Abstract

The support vector machine (SVM) is the state-of-the-art classification method, and the doubly regularized SVM (DrSVM) is its important extension based on the elastic net penalty. DrSVM has been successfully applied in handling variable selection and while retaining (or discarding) correlated variables. However, it is challenging to solve this model. In this paper we develop an iterative $\ell_2$-SVM approach to implement DrSVM over high dimensional datasets via an alternating direction method of multipliers (ADMM). Our approach can significantly reduce the computation complexity and further accelerate the convergence speed. Moreover, the corresponding algorithms have global convergence property. Experimental results over the simulated and real-world gene datasets are encouraging.

Keywords: feature selection, SVM, DrSVM, sparse learning

1. Introduction

In this paper we are concerned with a binary classification problem. Assume that we are given a training data set $\mathcal{X} = \{(x_i, y_i), i = 1, \ldots, n\}$, where $x_i \in \mathbb{R}^p$ is the input instance and $y_i \in \{-1, 1\}$ is the corresponding label. The problem tends to seek a decision function so that we can predict the labels of test input instances.

\*Corresponding author

Email addresses: dehualiu0427@gmail.com (D. Liu), qianhui@zju.edu.cn (H. Qian), daiguang116@gmail.com (G. Dai), zhzhang@cs.zju.edu.cn (Z. Zhang)
The support vector machine (SVM) is a classical classification method. It is a large margin classifier in which a hyperplane yields the correct classification when input instances are separable. Usually, a set of slack variables are introduced to deal with the non-separable case. Particularly, for a binary classification problem, ones consider the optimization problem as follows:

\[
\min_{w, b, \xi} \lambda \|w\|_q^q + \sum_{i=1}^{n} \xi_i \\
\text{s. t. } y_i (x_i^T w + b) \geq 1 - \xi_i, \\
\xi_i \geq 0, i = 1, \ldots, n,
\]

where \(w = (w_1, \ldots, w_p)^T\) is the vector of regression coefficients, \(\xi = (\xi_1, \ldots, \xi_n)^T\) is the vector of slack variables, and \(\lambda > 0\) is a regularization parameter (or tuning parameter). Note that in the problem, the original input vector \(x\) can be replaced by \(\phi(x)\), where \(\phi\) is a feature map that transform \(x\) into high-dimensional feature space.

The standard SVM arises when \(q = 2\) and can be solved by transforming it to a quadratic optimization problem. In this case, a direct merit is that SVM can be extended easily to the kernel framework. This formulation makes it unnecessary to find the explicit feature map \(\phi\), and it is also unnecessary to obtain the coefficient vector \(w\).

However, selecting the true underlying original variables is fundamental in some applications such as the microarray data analysis. In such cases, the number of instances is usually small but the dimension is very high.

There are many methods for feature selection under the SVM framework in the literature. For example, Rakotomamonjy [1] investigated the efficiency of feature selection criteria that derived from SVM. They suggested to select the best several features based on ranking criterion. [2, 3] employed a scalar factor to access the relevance of features.

There is another popular approach for achieving the purpose of feature selection in high dimensional space: the use of sparsity-inducing penalties [4]. This approach usually have much simpler model and lower computational complexity compared to the methods mentioned above.

An intuitive and natural approach is to impose the \(\ell_0\)-norm penalty to \(w\), i.e., \(\|w\|_0 = \text{card}\{w_i | w_i \neq 0\}\). However, since the resulting problem is typically NP hard, it is required to use tractable functions to approximate \(\|w\|_0\), e.g., a negative exponential function \(\sum_{i=1}^{p} (1 - e^{-\alpha w_i}) \) with \(|w_i| \leq v_i\) [5], or a nonconvex logarithmic function \(\sum_{i=1}^{p} \ln(\epsilon + |w_i|)\) [6], etc.
A sparse SVM model using the $\ell_1$-norm penalty is another choice for handling feature selection and classification simultaneously, called $\ell_1$-SVM [7, 8, 9]. The $\ell_1$-SVM has some advantages over the classical SVM when there are redundant noise features.

Unfortunately, the $\ell_1$-SVM cannot select correlated features simultaneously. In order to further address this limitation, [10] proposed a doubly regularized SVM (DrSVM) by employing the elastic-net penalty [11], due to its ability in capturing group effect. DrSVM has also appeared in the framework of knowledge based SVM model [12]. DrSVM automatically performs feature selection, moreover, it encourages highly correlated features to be selected (or removed) together. Formally, it is defined as

\[
\min_{w, b, \xi} \frac{\lambda_2}{2} \|w\|_2^2 + \lambda_1 \|w\|_1 + \frac{1}{n} \sum_{i=1}^{n} \xi_i, \tag{1}
\]

s. t. \(y_i(x_i^T w + b)_+ \geq 1 - \xi_i, \quad \xi_i \geq 0, \quad i = 1, \ldots, n,\)

where both $\lambda_1$ and $\lambda_2$ are the regularization parameters. DrSVM can also be equivalently formulated as the minimization of the hinge loss function plus the elastic net penalty. That is

\[
\min_{w, b} \frac{1}{n} \sum_{i=1}^{n} [1 - y_i(x_i^T w + b)]_+ + \frac{\lambda_2}{2} \|w\|_2^2 + \lambda_1 \|w\|_1. \tag{2}
\]

where $[z]_+ = \max\{z, 0\}$. $[1 - y_i(x_i^T w + b)]_+$ is the hinge loss function for SVMs.

The $\ell_1$-norm penalty makes this model have the feature selection property. Theorem 1 in [10] tells us that the $\ell_2$-norm penalty makes it enjoy the group effect property. That is, the larger $\lambda_2$ corresponds to more obvious group effect.

Since both the hinge loss and $\ell_1$ norm are not differentiable, [10] applied a huberized hinge loss to replace the original one, developing a so-called hybrid huberized SVM (HHSVM). As well, they devised an algorithm to calculate a whole solution path for HHSVM. Since the algorithm needs to track disappearance of features along a regularization path, its implementation is not easy. In fact, the nondifferentiability of hinge loss and the $\ell_1$-norm is not an obstacle for solving the DrSVM. Recently, [13] proposed an alternating direction method of multipliers (ADMM) to efficiently solve DrSVM. The main idea is to re-express the
problem (1) as an equivalent form

\[
\min_{\mathbf{w}, \mathbf{b}, \mathbf{u}} \frac{1}{n} \sum_{i=1}^{n} \eta_i + \lambda_1 \| \mathbf{u} \|_1 + \frac{\lambda_2}{2} \| \mathbf{w} \|_2^2
\]

\[
\text{s.t. } \eta = \mathbf{1} - \mathbf{Y}(\mathbf{Xw} + \mathbf{b1}), \quad \mathbf{w} = \mathbf{u},
\]

where \( \eta = (\eta_1, \ldots, \eta_n)^T \), \( \mathbf{1} \) is the \( n \times 1 \) vector of ones, and \( \mathbf{Y} \) is the \( n \times n \) diagonal matrix with \( y_i \) on the \( i \)th diagonal element. \( \mathbf{X} \) is a matrix of size \( n \times p \) where each row of \( \mathbf{X} \) is an input instance vector. Hence, the corresponding Lagrangian function of the problem (3) can be further expressed as follows.

\[
\mathcal{L}_0(\mathbf{w}, \mathbf{b}, \eta, \mathbf{u}, \mathbf{c}, \mathbf{v}) = \frac{1}{n} \sum_{i=1}^{n} \eta_i + \lambda_1 \| \mathbf{u} \|_1 + \frac{\lambda_2}{2} \| \mathbf{w} \|_2^2
\]

\[
+ \langle \mathbf{c}, \mathbf{1} - \mathbf{Y}(\mathbf{Xw} + \mathbf{b1}) - \eta \rangle + \langle \mathbf{v}, \mathbf{w} - \mathbf{u} \rangle,
\]

where \( \mathbf{c} \in \mathbb{R}^n \) and \( \mathbf{v} \in \mathbb{R}^p \) are the dual variables for the constraints. Then, ones solve the following augmented Lagrangian function:

\[
\mathcal{L}(\mathbf{w}, \mathbf{b}, \eta, \mathbf{u}, \mathbf{c}, \mathbf{v}) = \mathcal{L}_0(\mathbf{w}, \mathbf{b}, \eta, \mathbf{u}, \mathbf{c}, \mathbf{v}) + \frac{\rho_1}{2} \| \mathbf{1} - \mathbf{Y}(\mathbf{Xw} + \mathbf{b1}) - \eta \|_2^2
\]

\[
+ \frac{\rho_2}{2} \| \mathbf{w} - \mathbf{u} \|_2^2.
\]

The corresponding solution can be calculated by an iterative optimization procedure (see the details in [13]). However, in each \( k \)th iteration, the high computation complexity is involved by solving \( (\mathbf{w}, \mathbf{b}) \) with the other parameters fixed. That is,

\[
(\mathbf{w}, \mathbf{b}) = \arg\min_{\mathbf{w}, \mathbf{b}} \left\{ \frac{\lambda_2}{2} \| \mathbf{w} \|_2^2 + \langle \mathbf{v}^{(k)}, \mathbf{w} - \mathbf{u}^{(k)} \rangle + \langle \mathbf{c}^{(k)}, \mathbf{1} - \mathbf{Y}(\mathbf{Xw} + \mathbf{b1}) - \eta^{(k)} \rangle \right.
\]

\[
+ \frac{\rho_1}{2} \| \mathbf{1} - \mathbf{Y}(\mathbf{Xw} + \mathbf{b1}) - \eta^{(k)} \|_2^2 + \frac{\rho_2}{2} \| \mathbf{w} - \mathbf{u}^{(k)} \|_2^2 \right\}.
\]

Optimizing function in (4) is equivalent to solving a linear equation of the form \((\mathbf{I}_{p+1} + \mathbf{A})\mathbf{x} = \mathbf{f}\), where \( \mathbf{I}_m \) is the \( m \times m \) identical matrix, \( \mathbf{A} \) is a \((p+1) \times (p+1)\) positive semidefinite matrix with rank at most \((n+1)\), and \( \mathbf{f} \) is a \((p+1) \times 1\) vector. Hence, it can be solved at most \( n+2 \) step by a conjugate gradient method [14], i.e. the complexity of this step is \( O((n+2)p^2) \).

However, according to the complexity of solving \( (\mathbf{w}, \mathbf{b}) \) at every iteration, the total running time still increases fast if \( p \) have a large increase.
In this paper, we discuss the large \( p \) and small \( n \) problem. We find that by reformulating the problem and using the ADMM technique, DrSVM can be optimized by solving the standard SVM at every iteration. Our formulation is different from [13] because their framework can only be solved in the primal space. Our new formulation can be easily solved in the dual space. As a result, the large \( p \) problem is transformed into a sequence of small \( n \) problems. Thus the computation time does not seriously depend on \( p \) any more. Our algorithms not only converge to the global optimal solution but also achieve the convergence within the smaller number of iterations in comparison with that of [13].

The rest parts of the paper are organized as follows. In section 2, we derive three efficient and effective algorithms to calculate the solution of high dimensional DrSVM problems. Then, we present some experimental results in Section 3 to demonstrate the efficiency and effectiveness of our algorithms. Finally, the conclusion is made in Section 4.

2. Algorithms and Convergence Analysis

In this section we develop the efficient algorithms for DrSVM in high-dimensional datasets and then present convergence analysis for the global optimal solution.

2.1. Efficient Algorithms for DrSVM

The formulation in (2) can be rewritten as

\[
\min_{w, b} \sum_{i=1}^{n} [1 - y_{i}(x_{i}^{T} w + b)]_{+} + \frac{\lambda_{2}}{2} \|w\|_{2}^{2} + \lambda_{1}\|u\|_{1} \tag{5}
\]

s.t. \( w = u \).

The corresponding Lagrangian function is then

\[
L_{0}(w, b, u, \mu) = \sum_{i=1}^{n} [1 - y_{i}(x_{i}^{T} w + b)]_{+} + \lambda_{1}\|u\|_{1} + \frac{\lambda_{2}}{2} \|w\|_{2}^{2} + \mu^{T}(w - u). \tag{6}
\]

Furthermore, the augmented Lagrangian is

\[
L(w, b, u, \mu) = \sum_{i=1}^{n} [1 - y_{i}(x_{i}^{T} w + b)]_{+} + \lambda_{1}\|u\|_{1} + \frac{\lambda_{2}}{2} \|w\|_{2}^{2} \tag{7}
\]

\[+\mu^{T}(w - u) + \frac{\rho}{2} \|w - u\|_{2}^{2}.\]
Let $f_1(w, b) = \sum_{i=1}^{n}[1 - y_i(x_i^Tw + b)]_+ + \frac{\lambda_2}{2}\|w\|_2^2$ and $f_2(u) = \lambda_1\|u\|_1$. Then, we can employ the method of multipliers to solve the optimization problem (5), it consists the following two steps at every iteration

\[
(w_{*(k+1)}, b_{*(k+1)}, u_{*(k+1)}) = \arg \min_{w, b, u} L(w, b, u, \mu_{*(k)}),
\]

and

\[
\mu_{*(k+1)} = \rho(w_{*(k+1)} - u_{*(k+1)}) + \mu_{*(k)}.
\]

As we see, for high-dimensional problems, it is not efficient to calculate the optimization problem within the primal step (8). We now attempt to overcome this problem. Let $\xi_i = [1 - y_i(x_i^T w + b)]_+$ and $\xi = (\xi_1, \ldots, \xi_n)^T$. Then the optimization problem (8) is transformed into

\[
\arg \min_{\xi, b, w, u} \left\{ 1^T \xi + \frac{\lambda_2}{2}\|w\|_2^2 + \lambda_1\|u\|_1 + \delta(w, b, \xi | Y(Xw + b1 \geq 1 - \xi) + \delta(w, b, \xi | Y(Xw + b1 \geq 0) + \rho \|w - u\|_2^2 \right\}
\]

where $\delta(x|A) = 0$ if $x \in A$, otherwise $\delta(x|A) = \infty$.

Unfortunately, it is not easy to calculate $(w, \xi, b, u)$ in the problem (10) simultaneously. Alternatively, the problem (10) can be iteratively solved via the two optimization steps as follows

\[
(w_{j+1}^{(k+1)}, b_{j+1}^{(k+1)}, \xi_{j+1}^{(k+1)}) = \arg \min_{w, b, \xi} \left\{ 1^T \xi + \frac{\lambda_2 + \rho}{2}\|w\|_2^2 + \delta(w, b, \xi | Y(Xw + b1 \geq 1 - \xi) + \delta(w, b, \xi | Y(Xw + b1 \geq 0) \right\}
\]

and

\[
u_{j+1}^{(k+1)} = \arg \min_{u} \left\{ \lambda_1\|u\|_1 + \frac{\rho}{2}\|u - (w_{j+1}^{(k+1)} + \mu_{j}^{(k)})\|_2^2 \right\}
\]

In addition, (11) is equivalent to the following optimization problem:

\[
\min_{w, b, \xi} 1^T \xi + \frac{\lambda_2 + \rho}{2}\|w\|_2^2 + w^T(\mu_{j}^{(k)} - \rho u_{j}^{(k+1)}).
\]

s.t. $Y(Xw + b1) \geq 1 - \xi, \xi \geq 0$. 6
For high-dimensional problems with $p > n$, (13) can be efficiently solved by using its dual problem in the following form

$$
\min_\mathbf{a} \quad \frac{\mathbf{a}^T \mathbf{Y} \mathbf{X} \mathbf{X}^T \mathbf{Y} \mathbf{a}}{2(\lambda_2 + \rho)} + \left( \frac{\mathbf{Y} \mathbf{X} (\rho \mathbf{u}_j^{(k+1)} - \mathbf{\mu}_s^{(k)})}{\lambda_2 + \rho} - 1 \right)^T \mathbf{a}
$$

(14)

s.t. $\mathbf{a}^T \mathbf{y} = 0$, and $0 \leq \mathbf{a} \leq 1$.

where $\mathbf{a} = (a_1, \ldots, a_n)^T$. Now we find that it shares the same form as the quadratic programming (QP) in classical SVM. Subsequently, based on the solution $\mathbf{a}_{j+1}^{(k+1)} = ((a_1)_{j+1}^{(k+1)}, \ldots, (a_n)_{j+1}^{(k+1)})^T$ for (14), we can calculate the solution $\mathbf{w}_{j+1}^{(k+1)}$ for (11) as follows:

$$
\mathbf{w}_{j+1}^{(k+1)} = \frac{\mathbf{X}^T \mathbf{Y} \mathbf{a}_{j+1}^{(k+1)} + \rho \mathbf{u}_j^{(k+1)} - \mathbf{\mu}_s^{(k)}}{\lambda_2 + \rho}.
$$

Note that $b_{j+1}^{(k+1)}$ for (11) must satisfy $y_i (\mathbf{x}_i^T \mathbf{w}_{j+1}^{(k+1)} + b_{j+1}^{(k+1)}) = 1$ for such $i$ that $0 < (a_i)_{j+1}^{(k+1)} < 1$. As for (12), it is easily obtained that

$$
\mathbf{u}_{j+1}^{(k+1)} = \text{sign}(\rho \mathbf{w}_{j+1}^{(k+1)} + \mathbf{\mu}_s^{(k)}) \max\{0, |\rho \mathbf{w}_{j+1}^{(k+1)} + \mathbf{\mu}_s^{(k)} - \lambda_1 1|\}/\rho.
$$

Here, $\rho$ can either be fixed or increase monotonically.

Observe that in the iterative optimization scheme between (11) and (12), the calculation of $\mathbf{w}_{j+1}^{(k+1)}$ and $\mathbf{u}_{j+1}^{(k+1)}$ is independent on the calculation of $b_{j+1}^{(k+1)}$ and $\mathbf{\xi}_{j+1}^{(k+1)}$. Further, calculation of $\mathbf{\xi}$ is unnecessary, and $b$ can be calculated outside the inner loop: i.e., we can calculate $b_{s}^{(k)} = \frac{1}{N_\mathcal{M}} \sum_{i \in \mathcal{M}} (y_i - \mathbf{x}_i^T \mathbf{w}_s^{(k)})$, where $\mathcal{M} = \{0 < a_{si}^{(k)} < 1\}$ with the size $N_\mathcal{M}$. We summary the above calculation in Algorithm 1 in which $\rho$ exponentially increases. We call the algorithm iterative SVM 1 (ISVM1).

However, the computational cost of ISVM1 is still high because it requires to solve the quadratic optimization problem within the inner loop many times. In fact, in order to calculate the solution for DrSVM, it is unnecessary to obtain an exact solution for the subproblem (8). Hence, the inner iteration in ISVM1 can be implemented only once. Finally, ISVM1 can be further speeded up, and it is described in ISVM2 with $\rho$ being fixed and ISVM3 with $\rho$ being increased exponentially (see Algorithms 2 and 3).
Algorithm 1 Iterative Support Vector Machine Algorithm 1 (ISVM1)

1: Input: \( \mathcal{X} = \{ (x_i, y_i) \}_{i=1}^n \), \( \lambda_1, \lambda_2, \rho^{(0)}, \alpha, k = 0, w = w^{(0)}, u = u^{(0)}, \) and \( \mu = \mu^{(0)} \).
2: repeat
3: Set \( j = 0, w_0^{(k+1)} = w^{(k)} \) and \( u_0^{(k+1)} = u^{(k)} \);
4: while not convergence do
5: Calculate \( a_{j+1}^{(k+1)} = \arg\min_a \frac{1}{2(\lambda_2 + \rho^{(k)})} a^T Y X X^T Y a + \left( \frac{Y X (\rho^{(k)} u_{j+1}^{(k+1)} - \mu_*)}{\lambda_2 + \rho^{(k)}} - 1 \right)^T a \)
subject to \( a^T y = 0, \) and \( 0 \leq a \leq 1 \);
6: Calculate \( w_{j+1}^{(k+1)} = \frac{x^T y a_{j+1}^{(k+1)} + \rho^{(k)} u_{j+1}^{(k+1)} - \mu_*}{\lambda_2 + \rho^{(k)}} \);
7: Calculate \( u_{j+1}^{(k+1)} = \text{sign}(\rho^{(k)} w_{j+1}^{(k+1)} + \mu_*) \max\{0, |\rho^{(k)} w_{j+1}^{(k+1)} + \mu_* - \lambda_1 1\} / \rho^{(k)} \); \( j = j + 1 \);
8: end while
9: Set \( w_*^{(k+1)} = w_{j+1}^{(k+1)}, u_*^{(k+1)} = u_{j+1}^{(k+1)} \) and \( a_*^{(k+1)} = a_{j+1}^{(k+1)} \);
10: Calculate \( b_*^{(k+1)} = \frac{1}{N_M} \sum_{i \in \mathcal{M}} (y_i - x_i^T w_*^{(k+1)}) \), where \( \mathcal{M} = \{ 0 < a_*^{(k+1)} < 1 \} \) with size \( N_M \).
11: Calculate \( \mu_*^{(k+1)} = \mu_*^{(k)} + \rho^{(k)} (w_*^{(k+1)} - u_*^{(k+1)}) \);
12: Calculate \( \rho^{(k+1)} = \alpha \times \rho^{(k)} \);
13: \( k = k + 1 \);
14: until convergence
15: Output: the solution: \( w_* = w_*^{(k)}, u_* = u_*^{(k)}, b_* = b_*^{(k)} \); the classifier: \( h(x) = \text{sign}(x^T w_* + b_*) \).

2.2. Computational Cost Analysis

In this paper, we only focus on the algorithm ISVM2 and ISVM3. In every loop, the main computational burden is the QP (14), this is a scalar of size \( n \) which does not depend on \( p \). This SVM QP can be solved efficiently by SMO algorithm [15, 16] with the cost \( O(n^3) \). Although the number of iterations is unknown, experiments have shown that usually less than one hundred iterations will achieve convergence. Additionally, we can find from the description of algorithms that
Algorithm 2 ISVM2

1: Input: \(\mathcal{X} = \{(x_i, y_i)\}_{i=1}^{n}, \lambda_1, \lambda_2, \rho, u = (0), \mu = (0), k = 0.\)
2: repeat
3: Calculate \(a^{(k+1)}:\)
   \[
   a^{(k+1)} = \arg\min_a \quad \frac{1}{2(\lambda_2 + \rho)} a^T Y XX^T Y a + \left( \frac{Y X (\rho u^{(k)} - \mu^{(k)})}{\lambda_2 + \rho} - 1 \right)^T a
   \]
   subject to \(a^T y = 0, \text{ and } 0 \leq a \leq 1.\)
4: Calculate \(w^{(k+1)} = \frac{X^T Y a^{(k+1)}}{\lambda_2 + \rho} - \mu^{(k)}\)
5: Calculate \(u^{(k+1)} = \text{sign}(\rho w^{(k+1)} + \mu^{(k)}) \max \{0, |\rho w^{(k+1)} + \mu^{(k)}| - \lambda_1 1\}/\rho\)
6: Calculate \(\mu^{(k+1)} = \mu^{(k)} + \rho (w^{(k+1)} - u^{(k+1)})\)
7: \(k = k + 1.\)
8: until convergence
9: Output: the solution \(a^*, w^*, u^*\) and

\[
b^* = \frac{1}{N_M} \sum_{i \in \mathcal{M}} (y_i - x_i^T w^*), \quad \text{where } \mathcal{M} = \{i | 0 < a_i^* < 1\}, \ N_M \text{ the size of } \mathcal{M}
\]

The classifier is \(h(x) = \text{sign}(y(x^T w^* + b^*))\)

Algorithm 3 ISVM3

1: Let \(\rho\) increase exponentially in Algorithm 2, i.e. Line 6 in Algorithm 2 is replaced with

\[
\mu^{(k+1)} = \mu^{(k)} + \rho^{(k)} (w^{(k+1)} - u^{(k+1)}),
\]
\[
\rho^{(k+1)} = \alpha \times \rho^{(k)} \quad \text{where } \alpha > 1.
\]

the only parts of computational cost depending on \(p\) are the computation of \(XX^T\) and \(Xf\) with \(f\) a vector, the former is implemented only once before the loop, the later is implemented in every iteration with complexity \(O(np)\). Thus the total computational cost of algorithm ISVM2 or ISVM3 is \(O(K(n^3 + np))\) with \(K\) the number of iteration.
2.3. Convergence Analysis

In order to reveal the efficiency and feasibility of our algorithms, we further discuss their convergence analysis in what follows.

**Theorem 1.** For Algorithm ISVM1, any limit point \((w_*, b_*, u_*)\) of \((w^{(k)}_*, b^{(k)}_*, u^{(k)}_*)\) is an optimal solution to (5) and the convergence rate is at least \(O\left(\frac{1}{\rho^{k-1}}\right)\), i.e.,

\[
|f_1(w^{(k)}_*, b^{(k)}_*) + \lambda_1\|u^{(k)}_*\|_1 - p^*| = O\left(\frac{1}{\rho^{k-1}}\right),
\]

where \(p^*\) is the optimal value of the problem (5).

In order to generate the global convergence property of ISVM2, we rewrite the optimization problem (5) as

\[
\min_{w, b, u} f_1(w, b) + f_2(u)
\]

s.t. \(w - u = 0\).

The iteration between \(w, u, b, \mu\) constitutes an ADMM procedure that solves the problem (5). It has global convergence property if the following conditions are satisfied [17]:

(A) The extended real valued functions \(f_1 : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}\) and \(f_2 : \mathbb{R}^n \to \mathbb{R} \cup \{+\infty\}\) are closed, proper, and convex.

(B) The unaugmented Lagrangian (6) has a saddle point (but not necessarily unique), i.e., there exists \((w^*, b^*, \xi^*, u^*, \mu^*)\) such that

\[
L_0(w^*, b^*, \xi^*, u^*, \mu^*) \leq L_0(w^*, b^*, \xi^*, u^*, \mu^*) \leq L_0(w, b, \xi, u, \mu^*).
\]

Obviously, Condition (A) is trivial, and Condition (B) is also easily satisfied [18]. We summarize as the following theorem [17].

**Theorem 2.** The iterative sequence \((w^{(k)}_*, u^{(k)}_*, \mu^{(k)}_*)\) from algorithm ISVM2 satisfies the following properties:

(1) \(u^{(k)}_* - w^{(k)}_* \to 0\), i.e., the iteration approaches feasibility of the problem (5).

(2) The objective function in (5) approaches the optimal value \(p^*\) as \(k \to \infty\).

(3) Dual variable convergence: \(\mu^{(k)}_* \to \mu^*\), \(\mu^*\) is a dual optimal point.
Finally, the following theorem establishes the global convergence of algorithm ISVM3.

**Theorem 3.** Let \((w^\infty, b^\infty, u^\infty)\) be of the limit point of \((w^{(k)}, b^{(k)}, u^{(k)})\) from algorithm ISVM3. If \(\lim_{k \to \infty} \rho^{(k)}(u^{(k+1)} - u^{(k)}) = 0 \) and \(\sum_{k=1}^{\infty} \frac{1}{\rho^{(k)}} < \infty\), then \((w^\infty, b^\infty, u^\infty)\) is an optimal solution of the problem (5).

The proof of theorem 1 and 3 is given in Appendix.

3. Experimental Evaluation

In this section, we only consider algorithms ISVM2 and ISVM3, because ISVM1 needs a inner loop to solve a subproblem exactly which is time costly. Generally, all the parameters \(\rho, \lambda_1, \lambda_2\), and \(\alpha\) can be selected by the cross-validation (CV) technique, while giving rise to the high computational cost. For the sake of simplicity, in our experiments, the parameter \(\rho\) is set to \(\rho = 50\) in ISVM2, parameter \(\lambda_2 \in [1, 50]\) in ISVM2 and ISVM3, \(\alpha\) and \(\rho^{(0)}\) are set as \(\alpha = 1.2\) and \(\rho^{(0)} = 10\) in ISVM3. In addition, since the parameter \(\lambda_1\) have a great impact on sparsity of the model, a proper parameter \(\lambda_1\) is empirically set to \(\lambda_1 = 10 \times \|w_0\|_\infty\) in ISVM2 and ISVM3, where \(w_0\) is the solution of traditional SVM on the training dataset. The experimental results show that this is a proper choice. We find in the experiments that the accuracy is insensitive to the different value of parameters. However, the varying of parameters value has great influence on the speed of the algorithms and sparseness of coefficients \(w\).

Since Theorems 2 and 3 reveal that \(u^{(k)} - w^{(k)} \to 0\) as \(k \to \infty\), the stopping criteria for convergence in our algorithms can be set to \(\|u^{(k)} - w^{(k)}\|_2 \leq \tau\) where \(\tau = 10^{-4}\) in our experiments. Note that this stop condition is not sufficient to guarantee the convergence theoretically, however, it still works well in practice.

Recall that ADMM-DrSVM in [13] can efficiently solve the optimization problem for DrSVM. Moreover, it outperforms HHSVM [19] and other algorithms for DrSVM, significantly reducing the computational time for high-dimensional datasets with competitive classification accuracy. For simplicity, we only compare our algorithms with ADMM-DrSVM.

In order to validate the efficiency of our algorithms, we conduct experiments on the simulation and real-world datasets. Several evaluation criteria for comparison are reported, including average classification accuracy rate, average number of nonzero coefficients, average number of iteration, and average computational time.
3.1. Simulation Data

We are going to implement the experiment on simulation datasets. The two classes ‘+’ and ‘-’ are drawn from the distributions \( \mathcal{N}(\mu_+, \Sigma) \) and \( \mathcal{N}(\mu_-, \Sigma) \) respectively, where \( \mu_+ = (1, ..., 1, 0, ..., 0)^T \) with the first 10 components being 1 and the rest being 0, \( \mu_- = (-1, ..., -1, 0, ..., 0)^T \) with the first 10 components being -1 and the rest being 0, and the covariance matrix is

\[
\Sigma = \begin{pmatrix}
\Sigma_{10 \times 10}^* & 0_{10 \times (p-10)} \\
0_{(p-10) \times 10} & I_{(p-10) \times (p-10)}
\end{pmatrix}.
\]

Here \( \Sigma_{10 \times 10}^* \) is a \( 10 \times 10 \) matrix with all diagonal terms being 1 and the rest being \( r \), and \( 0_{k \times l} \) is the \( k \times l \) zero matrix. It should be noted that the larger \( r \) implies the higher correlation between the first 10 features of the input data. We implement our experiments on \( r = 0.1, 0.5, \) and 0.9, respectively. Obviously, the first 10 variables play a significant role on the classification decision. We report the averaged results of 10 runnings.

Firstly, we compare our algorithms with ADMM-DrSVM based on the number of iterations and computational time, while keeping the test accuracy and sparsity almost the same for all the compared algorithms. Table 1 reports the corresponding results. Table 1 shows that the speeds of both ISVM2 and ISVM3 are almost the same, and they are more efficient in comparison with ADMM-DrSVM.

<table>
<thead>
<tr>
<th></th>
<th>ISVM2</th>
<th>ISVM3</th>
<th>ADMM-DrSVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>m = 50</td>
<td>0.1</td>
<td>0.975</td>
<td>0.821</td>
</tr>
<tr>
<td>p = 2000</td>
<td>0.5</td>
<td>0.873</td>
<td>0.819</td>
</tr>
<tr>
<td>n = 50</td>
<td>0.1</td>
<td>0.967</td>
<td>0.819</td>
</tr>
<tr>
<td>m = 500</td>
<td>0.5</td>
<td>0.875</td>
<td>0.819</td>
</tr>
<tr>
<td>p = 5000</td>
<td>0.9</td>
<td>0.819</td>
<td>0.819</td>
</tr>
<tr>
<td>n = 50</td>
<td>0.1</td>
<td>0.948</td>
<td>0.800</td>
</tr>
<tr>
<td>m = 500</td>
<td>0.5</td>
<td>0.871</td>
<td>0.800</td>
</tr>
<tr>
<td>p = 10000</td>
<td>0.9</td>
<td>0.799</td>
<td>0.800</td>
</tr>
</tbody>
</table>

12
Secondly, we implement the simulation experiments in order to reveal how does the varying of $\lambda_2$ affect the number of true selected features (NTSF). Obviously, the ideal NTSF for our simulation datasets is 10. The corresponding results are given in Table 2. For the sake of brief, we only report the result of ISVM2, because we find that $\lambda_2$ varying in the interval $[1, 50]$ does not affect the NTSF in algorithm ISVM3. From Table 2, we can see that as the value of $\lambda_2$ increase, the NTSF increases, i.e. the group effect becomes more apparent.

<table>
<thead>
<tr>
<th>Data size</th>
<th>r</th>
<th>NTSF for different $\lambda_2$ (algorithm ISVM2)</th>
</tr>
</thead>
<tbody>
<tr>
<td>$n = 50$</td>
<td>0.1</td>
<td>8.8 9.3 9.5 9.5 9.9 9.9 10 10 10</td>
</tr>
<tr>
<td>$m = 500$</td>
<td>0.5</td>
<td>6.5 7.8 8.3 9.4 9.5 9.7 9.9 10 10</td>
</tr>
<tr>
<td>$p = 500$</td>
<td>0.9</td>
<td>8.0 9.2 9.7 9.7 10 10 10 10 10</td>
</tr>
<tr>
<td>$n = 50$</td>
<td>0.1</td>
<td>9.0 9.5 9.8 10 10 10 10 10 10</td>
</tr>
<tr>
<td>$m = 500$</td>
<td>0.5</td>
<td>8.2 9.1 9.7 9.9 9.9 10 10 10 10</td>
</tr>
<tr>
<td>$p = 1000$</td>
<td>0.9</td>
<td>8.9 9.9 9.9 9.9 10 10 10 10 10</td>
</tr>
<tr>
<td>$n = 50$</td>
<td>0.1</td>
<td>9.5 9.9 10 10 10 10 10 10 10</td>
</tr>
<tr>
<td>$m = 500$</td>
<td>0.5</td>
<td>8.6 9.5 9.9 10 10 10 10 10 10</td>
</tr>
<tr>
<td>$p = 2000$</td>
<td>0.9</td>
<td>9.9 10 10 10 10 10 10 10 10</td>
</tr>
</tbody>
</table>

3.2. Microarray Classification and Gene Selection

Typically, a microarray gene expression dataset consists of thousands of genes and a small number of instances, that is, gene selection in this setting is a high-dimension problem with $p \gg n$.

We conduct our experiments on three microarray gene expression datasets\(^1\), including colon, duke breast and leukemia. The description of datasets are given in Table 3. In our experiments, each dataset was standardized. Each dataset is randomly partitioned into two disjoint subsets as the training and test datasets, see Table 3.

Table 4 shows the experimental results, also including the standard SVM classification accuracy rate. According to Table 4, it should be obvious that our algo-

\(^1\)The datasets are available at http://www.csie.ntu.edu.tw/~cjlin/libsvmtools/datasets/.

13
Table 3: Summary of the benchmark datasets: \( C \)— the number of classes; \( n \)— the size of the training dataset; \( m \)— the size of the test data; \( p \)— the dimension of the input vector.

<table>
<thead>
<tr>
<th>Data set</th>
<th>( C )</th>
<th>( n )</th>
<th>( m )</th>
<th>( p )</th>
</tr>
</thead>
<tbody>
<tr>
<td>colon</td>
<td>2</td>
<td>31</td>
<td>31</td>
<td>2000</td>
</tr>
<tr>
<td>duke breast</td>
<td>2</td>
<td>22</td>
<td>22</td>
<td>7129</td>
</tr>
<tr>
<td>leukemia</td>
<td>2</td>
<td>38</td>
<td>34</td>
<td>7129</td>
</tr>
</tbody>
</table>

Rithms is more efficient than ADMM-DrSVM, and that they achieve the convergence within a small number of iterations. Moreover, the classification accuracies of our algorithms are very competitive with the other compared algorithms and the standard SVM. Moreover, the tuning parameter \( \alpha \) in Algorithm 3 efficiently speeds up the convergence of the algorithms.

Table 4: Experimental results for the three methods on the three gene expression datasets: CAR, CT, NT have the same meaning as in Table 1, NNC— the number of nonzero coefficients. Parameter setting: \( \lambda_1 = 10|w_0| \) and \( \lambda_2 = 1 \) for all algorithms (we changed the value of \( \lambda_1 \) in ADMM-DrSVM to improve its performance), \( \rho = 50 \) in ISVM2, and \( \rho^{(0)} = 10 \) and \( \alpha = 1.2 \) in ISVM3.

<table>
<thead>
<tr>
<th>Dataset</th>
<th>ISVM2</th>
<th>ISVM3</th>
<th>ADMM-DrSVM</th>
<th>SVM</th>
</tr>
</thead>
<tbody>
<tr>
<td>colon</td>
<td>0.830</td>
<td>0.65</td>
<td>79  1851</td>
<td>0.819</td>
</tr>
<tr>
<td>duke breast</td>
<td>0.825</td>
<td>0.27</td>
<td>54  6773</td>
<td>0.813</td>
</tr>
<tr>
<td>leukemia</td>
<td>0.824</td>
<td>0.84</td>
<td>53  6738</td>
<td>0.824</td>
</tr>
</tbody>
</table>

We have previously mentioned that although taking values of the tuning parameters in a certain interval does not affect the test accuracy, it does have great influence on the speed. We discuss this issue on the real-world datasets. From the experiments, for both algorithm ISVM2 and ISVM3, we find that a larger \( \lambda_1 \) will increase the number of iterations, leading to higher computational time. The increase of \( \lambda_2 \) will quickly decrease the number of iterations, leading to the lower computational time. We also find that in ISVM3, the appropriate increase of \( \alpha \) can also decrease the computational time to some extent.

We implemented the experiments to analyze the sparsity ability of our algorithms with respect to the different values of the parameters \( \lambda_1, \lambda_2, \) and \( \rho \). Here, the degree of sparsity is evaluated by using the number of zero elements in estimated \( w \) (We mention that in some papers, degree of sparsity is measured as the number of non-zero elements which is opposite to our definition). We find that the
degree of sparsity increases with respect to the parameter $\lambda_1$, and decrease with respect to $\lambda_2$. Additionally, the parameters $\lambda_1$ and $\lambda_2$ in ISVM3 have the similar influence on sparsity. We find that degree of sparsity is insensitive to the different values of $\rho$ in algorithms ISVM2. In addition, we also mention that increase of $\alpha$ in ISVM3 leads to decrease of the degree of sparsity.

4. Conclusion

In this paper we have developed three efficient algorithms for DrSVM in high-dimensional datasets. Our proposed algorithms proceed by solving the QP at every iteration, which largely reduces the scale of the original problem and accelerates the speed when $p \gg n$. We have shown that our algorithms have complexity $O(K(n^3 + np))$, which is much lower than $O(Knp^2)$ given by ADMM-DrSVM algorithm. Moreover, our algorithms are able to achieve global convergence. The experimental results on the simulation and real-world gene datasets illustrate the efficiency of our algorithms. In the future work, we will extend the binary classification setting to the multi-class case.

Appendix

A: Proof of Theorem 1

We denote $[1-y_i(w^T x_i + b)]_+ = \phi(y_i(w^T x_i + b))$, and $(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*) = \arg\min \sum_{i=1}^n \phi(y_i(w^T x_i + b)) + \frac{\lambda_2}{2} \|w\|^2 + \lambda_1 \|w\|_1 + \frac{\rho(k)}{2} \|w - u\|^2 + \langle \mu^{(k)}_*, w - u \rangle$ in the $(k + 1)$th iterative optimization procedure. Since $(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*)$ is the optimal solution for the optimization problem in above, we have that

$$0 \in \sum_{i=1}^n \partial \phi(y_i(w^{(k+1)}_* x_i + b^{(k+1)}_*) )y_i x_i + \lambda_2 \|w^{(k+1)}_*\| + \rho^{(k)}_*(w^{(k+1)}_* - u^{(k+1)}_*) + \mu^{(k)}_*,$$  

$$0 \in \sum_{i=1}^n \partial \phi(y_i(w^{(k+1)}_* x_i + b^{(k+1)}_* ) )y_i;$$  

$$0 \in \partial \|u^{(k+1)}_*\|_1 + \rho^{(k)}_*(u^{(k+1)}_* - w^{(k+1)}_*) - \mu^{(k)}_*.$$

Let

$$\mu^{(k+1)}_* = \mu^{(k)}_* + \rho^{(k)}_*(w^{(k+1)}_* - u^{(k+1)}_*).$$

15
Then, the equations (15) and (17) can be reexpressed as:

$$-\mu^{(k+1)}_* - \lambda_2 w^{(k+1)}_* \in \sum_{i=1}^{n} \partial \phi(y_i (w^{(k+1)}_*^T x_i + b^{(k+1)}_*)) y_i x_i, \quad (19)$$

$$\mu^{(k+1)}_* \in \lambda_1 \partial \| u^{(k+1)}_* \|_1. \quad (20)$$

Obviously, $\partial \phi(\cdot)$ and $\partial \| u \|_1$ are bounded. Then, (19) and (20) tell us that $\mu^{(k+1)}_*$ and $\mu^{(k+1)}_* + \lambda_2 w^{(k+1)}_*$ are bounded. Hence, $w^{(k+1)}_*$ is bounded. If we denote $(w_*, u_*)$ the limit point of $(w^{(k)}_*, u^{(k)}_*)$, from the fact that there exists $M$, such that $\|w^{(k+1)}_* - u^{(k+1)}_*\|_2 \leq \frac{M}{\rho_*}$, we have $w_* = u_*$, so the limit point is feasible. On the one hand, since

$$L(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*, \mu^{(k)}_*, \rho^{(k)}_*) = \min_{w, b, u, \mu, \rho} L(w, b, u, \mu_*, \rho_*)$$

$$\leq \min_{w=U} L(w, b, u, \mu_*, \rho_*)$$

$$= \min \sum_{i=1}^{n} \partial \phi(y_i (w^T x_i + b)) + \frac{\lambda_2}{2} \| w \|_2^2 + \lambda_1 \| w \|_1 = p^*,$$

then we have

$$\sum_{i=1}^{n} \phi(y_i (w^{(k+1)}_*^T x_i + b^{(k+1)}_*)) + \frac{\lambda_2}{2} \| w^{(k+1)}_* \|_2^2 + \lambda_1 \| u^{(k+1)}_* \|_1$$

$$= L(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*, \mu^{(k)}_*, \rho^{(k)}_*) - \langle \mu^{(k)}_*, w^{(k+1)}_* - u^{(k+1)}_* \rangle$$

$$= L(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*, \mu^{(k)}_*, \rho^{(k)}_*) - \langle \mu^{(k)}_*, \frac{\mu^{(k+1)}_* - \mu^{(k)}_*}{\rho^{(k)}_*} \rangle$$

$$= L(w^{(k+1)}_*, b^{(k+1)}_*, u^{(k+1)}_*, \mu^{(k)}_*, \rho^{(k)}_*) - \frac{1}{2 \rho^{(k)}_*} \| \mu^{(k+1)}_* - \mu^{(k)}_* \|_2^2$$

$$\leq p^* - \frac{1}{2 \rho^{(k)}_*} (\| \mu^{(k+1)}_* \|_2^2 - \| \mu^{(k)}_* \|_2^2) = p^* + O(\rho^{(k)}_*). \quad (23)$$

On the other hand, from the boundedness of $w^{(k+1)}_*$, $u^{(k+1)}_*$, and $w^{(k+1)}_*$ -
\( u^{(k+1)} \|_2 = O((\rho_s^{(k)})^{-1}) \), we have
\[
\sum_{i=1}^{n} \phi(y_i(w^{(k+1)}_s x_i + b^{(k+1)})) + \frac{\lambda_2}{2} \|w^{(k+1)}_s\|_2^2 + \lambda_1 \|u^{(k+1)}_s\|_1
\]
\[
= \sum_{i=1}^{n} \phi(y_i(u^{(k+1)}_s x_i + b^{(k+1)})) + \frac{\lambda_2}{2} \|u^{(k+1)}_s\|_2^2 + \lambda_1 \|u^{(k+1)}_s\|_1
\]
\[
+ \frac{\lambda_2}{2} \|w^{(k+1)}_s\|_2^2 - \frac{\lambda_2}{2} \|u^{(k+1)}_s\|_2^2 + \sum_{i=1}^{n} \phi(y_i(w^{(k+1)}_s x_i + b^{(k+1)}))
\]
\[
- \sum_{i=1}^{n} \phi(y_i(u^{(k+1)}_s x_i + b^{(k+1)})) \geq p^* - \sum_{i=1}^{n} |(w^{(k+1)}_s x_i + b^{(k+1)}) - (u^{(k+1)}_s x_i)|
\]
\[
+ \frac{\lambda_2}{2} (w^{(k+1)}_s + u^{(k+1)}_s)^T (w^{(k+1)}_s - u^{(k+1)}_s) \geq p^* + O((\rho_s^{(k)})^{-1}).
\] (24)

From (23) and (24), this prove that \( |f_1(w^{(k)}_s, b^{(k)}) + \lambda_1 \|u^{(k)}_s\|_1 - p^*| = O(\frac{1}{\rho_s^{(k)}}) \).

**B: Proof of Theorem 3**

In the \((k+1)\)th iterative optimization procedure, let
\[
(w^{(k+1)}, b^{(k+1)}) = \arg\min \sum_{i=1}^{n} \phi(y_i(w^T x_i + b)) + \frac{\lambda_2}{2} \|w\|_2^2
\]
\[
+ \frac{\rho^{(k)}}{2} \|w - u^{(k)}\|_2^2 + \langle \mu^{(k)}, w - u^{(k)} \rangle,
\] (25)
\[
u^{(k+1)} = \arg\min \lambda_1 \|u\|_1 + \frac{\rho^{(k)}}{2} \|u - (w^{(k+1)} + \frac{\mu^{(k)}}{\rho^{(k)}})\|_2^2,
\] (26)
and
\[
\mu^{(k+1)} = \mu^{(k)} + \rho^{(k)}(w^{(k+1)} - u^{(k)}).
\] (27)

Since \((w^{(k+1)}, b^{(k+1)})\) and \(u^{(k+1)}\) are the optimal solution for (25) and (26), similarly to the proof of Theorem 2, we have
\[
-\mu^{(k+1)} - \lambda_2 w^{(k+1)} \in \sum_{i=1}^{n} \partial \phi(y_i(w^{(k+1)} T x_i + b^{(k+1)})) y_i x_i,
\] (28)
\[
0 \in \sum_{i=1}^{n} \partial \phi(y_i(w^{(k+1)} T x_i + b^{(k+1)})) y_i,
\] (29)
and
\[
\mu^{(k+1)} \in \lambda_1 \partial \|u^{(k+1)}\|_1.
\] (30)
Obviously, $\mu^{(k+1)}$ is bounded. According to $\hat{\mu}^{(k+1)} - \mu^{(k+1)} = \rho^{(k)}(u^{(k+1)} - u^{(k)})$, $\lim_{k \to \infty} \rho^{(k)}(u^{(k+1)} - u^{(k)}) \to 0$ and the equation (27), it is not difficult to find that $\hat{\mu}^{(k)}$ is bounded and $\|u^{(k+1)} - u^{(k)}\| = o((\rho^{(k)})^{-1})$. In addition, from (28), we can find $w^{(k)}$ is bounded. Hence, $u^{(k)}$ and $b^{(k)}$ is bounded as well.

Since $\rho^{(k)}$ grows exponentially and $u^{(k)}$ is a Cauchy sequence, $u^{(k)}$ converges to a limit $u^{(\infty)}$. In addition, from $\frac{1}{\rho^{(k)}}(\mu^{(k+1)} - \mu^{(k)}) = w^{(k+1)} - u^{(k+1)}$, we can see that the limit point satisfy $w^{\infty} = u^{\infty}$.

Since all terms in the objective function in (5) is convex, together with (28), (29), and (30), we have

$$
\sum_{i=1}^{n} \phi(y_i(w^{(k+1)T}x_i + b^{(k+1)})) + \frac{\lambda_2}{2}\|w^{(k+1)}\|^2 + \lambda_1\|u^{(k+1)}\|_1
\leq \sum_{i=1}^{n} \phi(y_i(w^{*T}x_i + b^{(k+1)})\|w^{(k+1)}\|^2 + \lambda_1\|u^{*}\|_1
\leq \mu^{(k+1)}, w^{(k+1)} - w^{(k+1)}) - \langle \mu^{(k+1)}, u^{(k+1)} - u^{(k+1)} \rangle
= p^* + O((\rho^{(k)})^{-1}) + \langle \mu^{(k+1)}, w^{(k+1)} - w^{(k+1)} \rangle - \langle \mu^{(k+1)}, u^{(k+1)} - u^{(k+1)} \rangle
= p^* + O((\rho^{(k)})^{-1}) + \frac{1}{\rho^{(k)}} \langle \mu^{(k+1)}, \mu^{(k+1)} - \mu^{(k)} \rangle
- \frac{1}{\rho^{(k)}} \langle \mu^{(k+1)}, \mu^{(k)} - \mu^{(k)} \rangle
$$

From the boundedness of $(\mu^{(k)}, \hat{\mu}^{(k)}, \mu^{*}, \hat{\mu}^{*}, w^{(k)}, w^{*})$ and $\rho^{(k)}(u^{(k+1)} - u^{(k)}$) $\to 0$, we find that the limit point $(w^{\infty}, b^{\infty}, u^{\infty})$ must satisfy $f_1(w^{\infty}, b^{\infty}) + \lambda_1\|u^{\infty}\|_1 = p^*$. Then, we obtain

$$f_1(w^{\infty}, b^{\infty}) + \lambda_1\|u^{\infty}\|_1 = p^*.$$

Acknowledgments

This work has been supported in part by the Natural Science Foundations of China (No. 61070239).

References


