Ramp Loss Linear Programming Nonparallel Support Vector Machine

Dalian Liu\textsuperscript{1,2}, Dandan Chen\textsuperscript{3,4}, Yong Shi\textsuperscript{1,4,5,6}, and Yingjie Tian\textsuperscript{4,5}

\textsuperscript{1} School of Computer and Information Technology, Beijing Jiaotong University, Beijing 100044, China
\textsuperscript{2} Department of Basic Course Teaching, Beijing Union University, Beijing 100101, China
\textsuperscript{3} College of Mathematical Sciences, University of Chinese Academy of Sciences, Beijing 100049, China
\textsuperscript{4} Research Center on Fictitious Economy and Data Science, University of Chinese Academy of Sciences, Beijing 100190, China, chendandan1@126.com
\textsuperscript{5} Key Laboratory of Big Data Mining and Knowledge Management, University of Chinese Academy of Sciences, Beijing 100190, China
\textsuperscript{6} College of Information Science and Technology, University of Nebraska at Omaha, Omaha, NE 68182, USA yshi@ucas.ac.cn, tyj@ucas.ac.cn

Abstract
Motivated by the fact that the $l_1$-penalty is piecewise linear, we proposed a ramp loss linear programming nonparallel support vector machine (ramp-LPNPSVM), in which the $l_1$-penalty is applied for the RNPSVM, for binary classification. Since the ramp loss has the piecewise linearity as well, ramp-LPNPSVM is a piecewise linear minimization problem and a local minimum can be effectively found by the Concave Convex Procedure and experimental results on benchmark datasets confirm the effectiveness of the proposed algorithm. Moreover, the $l_1$-penalty can enhance the sparsity.

Keywords: support vector machine, nonparallel, CCCP, linear programming, ramp loss

1 Introduction

As the computationally powerful tools for pattern classification, support vector machines (SVMs) developed fast\textsuperscript{[1, 4, 5, 25, 26]}. Recently, the nonparallel hyperplane SVM is proposed and has attracted many interests, such as the generalized eigenvalue proximal support vector machine (GEPSVM)\textsuperscript{[11]} and the twin support vector machine (TWSVM)\textsuperscript{[7]}. For the binary classification problem, TWSVM seeks two nonparallel proximal hyperplanes such that each hyperplane is closer to one of the two classes and is at least one distance from the other. It is implemented by solving two smaller quadratic programming problems (QPPs) instead of a larger one, which increases the TWSVM training speed by approximately fourfold compared to that of standard SVM. TWSVMs have been studied extensively\textsuperscript{[2, 8, 9, 13, 14, 15, 16, 17, 18, 19, 23, 22]}, in which the nonparallel support vector machine (NPSVM)\textsuperscript{[23]} is superior theoretically and overcomes several drawbacks of the existing TWSVMs.
However, researchers have shown that classical SVMs or TWSVMs are sensitive to the presence of outliers and yield poor generalization performance, since the outliers tend to have the largest margin losses according to the character of the convex loss functions used in them, such as the convex loss functions such as the Hinge loss function and \( \varepsilon \)-insensitive loss function. Therefore, several methods are applied to construct the robust SVM models [3, 6, 10, 12, 20, 21, 24, 29], of which the ramp loss function has been investigated widely in the theoretical literature in order to improve the robustness of SVMs. They constructed a ramp loss support vector machine (RSVM) by taking the Ramp loss instead of the Hinge loss in the classical SVM, the Ramp loss function limits its maximal loss value distinctly and can put definite restrictions on the influences of outliers so that it is much less sensitive to their presence. However, it will also cause the objective of SVMs losing convexity, as a consequence, the concave-convex programming (CCCP) procedure is applied to solves a sequence of convex problems to produce faster and sparser SVMs. For the NPSVM [23], by introducing the ramp loss function and also propose a new non-convex and non-differentiable loss function based on the \( \varepsilon \)-insensitive loss function, a novel ramp loss NPSVM termed as RNPSVM is proposed [11], which can explicitly incorporate noise and outlier suppression in the training process, has less support vectors and the increased sparsity leads to its better scaling properties. The non-convexity of RNPSVM can be efficiently solved by the Concave Convex Procedure.

In this paper, we propose a ramp-LPNPSVM based on our proposed ramp loss NPSVM (RNPSVM) [23], which implies that the algorithm proposed later involves no more quadratic programming problems (QPPs) but linear programming problems (LPPs). Similarly to RNPSVM, the proposed ramp-LPNPSVM enjoys the robustness and sparsity. The problems related to ramp-NPLPSVM leads to a polyhedral concave problem, which minimizes a concave function on one polyhedron. Moreover, ramp-NPLPSVM has the piecewise linear objective functions, which made a sequence of LPPs to be solved efficiently in the CCCP procedure. The rest of this paper is organized as follows. Section 2 briefly dwells on the hinge loss SVM, ramp Loss SVM and ramp loss NPSVM. Section 3 proposes the ramp-LPNPSVM and its corresponding algorithm. Section 4 deals with experimental results and Section 5 contains concluding remarks.

2 Background

In this section, we briefly introduce hinge loss SVM, ramp loss SVM and RNPSVM [11].

2.1 Hinge Loss SVM

Given a training set

\[ T = \{(x_1, y_1), \cdots, (x_l, y_l)\} \]  

where \( x_i \in \mathcal{R}^n \), \( y_i \in \mathcal{Y} = \{1, -1\}, i = 1, \cdots, l \), the Hinge Loss SVM relies on the classical Hinge loss function

\[ H_s(z) = \max(0, s - z) \]  

(2)

to be formulated as the following optimization problem

\[ \min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^{l} H_1(y_i f(x_i)) \]  

(3)

where \( f(x) \) is the decision function with the form of \( f(x) = (w \cdot \Phi(x)) + b \), and \( \Phi(\cdot) \) is the chosen feature map, often implicitly defined by a Mercer kernel \( K(x, x') = (\Phi(x) \cdot \Phi(x')) \) [26]. Hinge loss SVM has the sensitivity to outlier observations and its generalization performance is degraded [27].
2.2 Ramp Loss SVM

Ramp loss SVM increases the robustness of SVM by the ramp loss function [3]

\[
R_s(z) = \begin{cases} 
0, & z > 1 \\
1 - z, & s \leq z \leq 1 \\
1 - s, & z < s
\end{cases}
\] (4)

which makes the loss function flat for scores \( z \) smaller than a predefined value \( s < 1 \). \( R_s(z) \) can be decomposed into the sum of the convex Hinge Loss and a concave loss,

\[
R_s(z) = H_1(z) - H_s(z),
\] (5)

therefore the ramp loss SVM (RSVM) is formulated as a following optimization problem

\[
\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l R_s(y_i f(x_i))
= \frac{1}{2} \|w\|^2 + C \sum_{i=1}^l H_1(y_i f(x_i)) - C \sum_{i=1}^l H_s(y_i f(x_i)),
\] (6)

which can be solved by the Concave-Convex Procedure (CCCP) [28]

Figure 1: [11] The Ramp Loss function (left) can be decomposed into the sum of the convex Hinge Loss (middle) and a concave loss (right).

Figure 2: [11] Ramp \( \varepsilon \)-insensitive Loss function (left) can be decomposed into the sum of the convex \( \varepsilon \)-insensitive Loss (middle) and a Concave loss (right).
2.3 RNPSVM

Given the training set

\[ T = \{(x_1, +1), \ldots, (x_p, +1), (x_{p+1}, -1), \ldots, (x_{p+q}, -1)\} \]  

(7)

RNPSVM seeks two nonparallel hyperplanes \( f_+(x) = (w_+ \cdot \Phi(x)) + b_+ = 0 \) and \( f_-(x) = (w_- \cdot \Phi(x)) + b_- = 0 \) by solving two problems

\[
\min_{w_+, b_+} \frac{1}{2} \|w_+\|^2 + C_1 \sum_{i=1}^{p} R_{e,t}(f_+(x_i)) + C_2 \sum_{j=p+1}^{p+q} R_s(-f_+(x_j))
\]

(8)

and

\[
\min_{w_-, b_-} \frac{1}{2} \|w_-\|^2 + C_3 \sum_{j=p+1}^{p+q} R_{e,t}(f_-(x_j)) + C_4 \sum_{i=1}^{p} R_s(-f_-(x_j))
\]

(9)

where \( C_i > 0, i = 1, \ldots, 4 \) are penalty parameters, and \( R_{e,t} \) is the proposed \( \varepsilon \)-insensitive ramp loss function in [11] (see Fig.2(a)),

\[
R_{e,t}(z) = \begin{cases} 
  t - \varepsilon, & |z| > t \\
  |z| - \varepsilon, & \varepsilon \leq |z| \leq 1 \\
  0, & |z| < \varepsilon 
\end{cases}
\]

(10)

which makes the \( \varepsilon \)-insensitive loss function

\[
I_{\varepsilon}(z) = \max(0, |z| - \varepsilon)
\]

(11)

flat for scores \( z \) larger than a predefined value \( t > \varepsilon \). It is obvious that \( R_{e,t}(z) \) can be decomposed into the sum of the convex \( \varepsilon \)-insensitive loss and a concave loss,

\[
R_{e,t}(z) = I_{\varepsilon}(z) - I_{t}(z)
\]

(12)

From (5) and (12), the problems (8) and (9) of the RNPSVM is reformulated as

\[
\min_{w_+, b_+} \frac{1}{2} \|w_+\|^2 + C_1 \sum_{i=1}^{p} I_{\varepsilon}(f_+(x_i)) + C_2 \sum_{j=p+1}^{p+q} H_s(-f_+(x_j))
\]

(13)

\[
\begin{align*}
&= C_1 \sum_{i=1}^{p} I_{t}(f_+(x_i)) + C_2 \sum_{j=p+1}^{p+q} H_s(-f_+(x_j)), \\
&\text{convex}
\end{align*}
\]

and

\[
\min_{w_-, b_-} \frac{1}{2} \|w_-\|^2 + C_3 \sum_{j=p+1}^{p+q} I_{\varepsilon}(f_-(x_j)) + C_4 \sum_{i=1}^{p} H_s(-f_-(x_j))
\]

(14)

\[
\begin{align*}
&= C_3 \sum_{j=p+1}^{p+q} I_{t}(f_-(x_j)) + C_4 \sum_{i=1}^{p} H_s(-f_-(x_j)), \\
&\text{convex}
\end{align*}
\]

RNPSVM has been proved to explicitly incorporate noise and outlier suppression in the training process, has less support vectors and the increased sparsity leads to its better scaling properties.
3 Ramp Loss Linear Programming NPSVM

In this section, we propose the Ramp Loss Linear Programming NPSVM, termed as ramp-LPNPSVM, for which the $l_1$ regularization terms are applied, and the algorithm proposed later involves a sequence of linear programming problems.

3.1 Linear case

We seek the two nonparallel hyperplanes $f_+(x) = (w_+ \cdot \Phi(x)) + b_+ = 0$ and $f_-(x) = (w_- \cdot \Phi(x)) + b_- = 0$ by solving two problems

$$\min_{w_+ , b_+} \frac{1}{2} \|w_+\|_1 + C_1 \sum_{i=1}^{p} I_+(f_+(x_i)) + C_2 \sum_{j=p+1}^{p+q} H_1(-f_+(x_j))$$

$$= C_1 \sum_{i=1}^{p} I_+(f_+(x_i)) + C_2 \sum_{j=p+1}^{p+q} H_1(-f_+(x_j))$$

and

$$\min_{w_-, b_-} \frac{1}{2} \|w_-\|_1 + C_3 \sum_{j=p+1}^{p+q} I_-(f_-(x_j)) + C_4 \sum_{i=1}^{p} H_1(-f_+(x_j))$$

$$= C_3 \sum_{j=p+1}^{p+q} I_-(f_-(x_j)) + C_4 \sum_{i=1}^{p} H_1(-f_+(x_j))$$

where we only change the $\|w_+\|^2$ and $\|w_-\|^2$ in (15) and (16) into the $l_1$-penalty $\|w_+\|_1$ and $\|w_-\|_1$. We can see that the two problems has the piecewise linear objective functions being composed of a convex part and a concave part. Follow the same idea in [11], for the problem with such objective function, the CCCP algorithm is an efficient iterative procedure that solves a sequence of convex programs. Here we take the first problem as the example, the second is the similar. Let the concave part of the problem (15)

$$P_{cave}(w_+, b_+) = -C_1 \sum_{i=1}^{p} I_+(f_+(x_i)) - C_2 \sum_{j=p+1}^{p+q} H_1(-f_+(x_j))$$

(17)

The CCCP framework for the problem (15) is constructed as in Algorithm 1.

Note that $P_{cave}(w_+, b_+)$ is non-differentiable at some points, for simplification purposes, we introduce the sub-gradient notations

$$\delta_j = -C_2 y_j \frac{\partial H_1(y_j f_+(x_j))}{\partial f_+(x_j)} = \begin{cases} C_2, & \text{if } y_j f_+(x_j) < s \\ 0, & \text{otherwise} \end{cases}$$

(18)

for $j = p + 1, \ldots, p + q$, and

$$\theta_j = -C_1 \frac{\partial I_+(f_+(x_j))}{\partial f_+(x_j)} = \begin{cases} -C_1, & \text{if } f_+(x_j) > t \\ C_1, & \text{if } f_+(x_j) < -t \\ 0, & \text{otherwise} \end{cases}$$

(19)
Algorithm 1 CCCP for the problem (15)

(1) Initialize \((w^0, b^0)\), set \(k = 0\);
(2) Construct and solve the problem

\[
\begin{align*}
\min_{w^+, b^+, \eta^+, \xi^-} & \quad \frac{1}{2} \|w^+\|_1 + C_1 \sum_{i=1}^{p} (\eta_i + \eta_i^*) + C_2 \sum_{j=p+1}^{p+q} \xi_j + P_{cav}(w_i^+, b_i^+) \cdot (w^+ b^+) \\
\text{s.t.} & \quad (w^+ \cdot x_i) + b^+ \leq \varepsilon + \eta_i, \; i = 1, \cdots, p, \\
& \quad -(w^+ \cdot x_i) - b^+ \leq \varepsilon + \eta_i^*, \; i = 1, \cdots, p, \\
& \quad (w^+ \cdot x_j) + b^+ \leq -1 + \xi_j, \; j = p + 1, \cdots, p + q, \\
& \quad \eta_i, \eta_i^* \geq 0, \; \xi_j \geq 0, \; i = 1, \cdots, p, \; j = p + 1, \cdots, p + q
\end{align*}
\]

get the solution \((w^{k+1}, b^{k+1})\);
(3) If \((w^k, b^k)\) not convergence, set \(k = k + 1\), go to step (2).

for \(i = 1, \cdots, p\). And we also introduce the variable vector \(u^+\) such that \(u_i = |w_{+i}|, \; i = 1, \cdots, n\), therefore the problem (20) turns to be a LPP

\[
\begin{align*}
\min_{w^+, u^+, b^+} & \quad \frac{1}{2} \|w^+\|_1 + C_1 \sum_{i=1}^{p} (\eta_i + \eta_i^*) + C_2 \sum_{j=p+1}^{p+q} \xi_j + \sum_{i=1}^{p} \theta_i ((w^+ \cdot x_i) + b^+) \\
& \quad + \sum_{j=p+1}^{p+q} \delta_j y_j ((w^+ \cdot x_i) + b^+) \\
\text{s.t.} & \quad (w^+ \cdot x_i) + b^+ \leq \varepsilon + \eta_i, \; i = 1, \cdots, p, \\
& \quad -(w^+ \cdot x_i) - b^+ \leq \varepsilon + \eta_i^*, \; i = 1, \cdots, p, \\
& \quad (w^+ \cdot x_j) + b^+ \leq -1 + \xi_j, \; j = p + 1, \cdots, p + q, \\
& \quad \eta_i, \eta_i^* \geq 0, \; \xi_j \geq 0, \; i = 1, \cdots, p, \; j = p + 1, \cdots, p + q
\end{align*}
\]

Another LPP can be formulated as follows if two variable vectors \(u^+\) and \(v^+\) are introduced satisfying \(w^+ = u^+ - v^+, |w_{+i}| = u_{+i} + v_{+i}, \; i = 1, \cdots, n\),

\[
\begin{align*}
\min_{u^+, v^+, b^+} & \quad \sum_{i=1}^{n} (u_{+i} + v_{+i}) + C_1 \sum_{i=1}^{p} (\eta_i + \eta_i^*) + C_2 \sum_{j=p+1}^{p+q} \xi_j \\
& \quad + \sum_{i=1}^{p} \theta_i ((u^+ - v^+) \cdot x_i) + b^+) + \sum_{j=p+1}^{p+q} \delta_j y_j ((u^+ - v^+) \cdot x_i) + b^+) \\
\text{s.t.} & \quad ((u^+ - v^+) \cdot x_i) + b^+ \leq \varepsilon + \eta_i, \; i = 1, \cdots, p, \\
& \quad -((u^+ - v^+) \cdot x_i) - b^+ \leq \varepsilon + \eta_i^*, \; i = 1, \cdots, p, \\
& \quad ((u^+ - v^+) \cdot x_j) + b^+ \leq -1 + \xi_j, \; j = p + 1, \cdots, p + q, \\
& \quad \eta_i, \eta_i^* \geq 0, \; \xi_j \geq 0, \; i = 1, \cdots, p, \; j = p + 1, \cdots, p + q
\end{align*}
\]
3.2 Nonlinear case

In [11], we have

\[ w_+ = \sum_{i=1}^{p} (\alpha_i^* - \alpha_i - \theta_i) x_i - \sum_{j=p+1}^{p+q} (\beta_j - \delta_j) x_j \]  

(23)

where \( \alpha, \alpha^*, \beta \geq 0 \) are the corresponding Lagrangian multiplier vectors, therefore we can assume that \( w_+ = \sum_{i=1}^{l} u_{+i} x_i \) and \( \|w_+\|^2 \) is the convex function of \( u_+ = (u_{+1}, \cdots, u_{+l})^T \). If we take some convex function \( f(s_+) \) to replace \( \|w_+\|^2 \), typically some norm or seminorm of \( u_+ \), we will get the generalized formulation. Here we choose the 1-norm of \( u_+ \), at the same time introduce the kernel function \( K(x, x') \) to get the LPPs for the nonlinear case

\[
\begin{align*}
\min_{u_+, s_+, b_+} & \quad \sum_{i=1}^{l} s_{+i} + C_1 \sum_{i=1}^{p} (\eta_i + \eta_i^*) + C_2 \sum_{j=p+1}^{p+q} \xi_j \\
& + \sum_{i=1}^{p} \theta_i (\sum_{k=1}^{l} u_{+k} K(x_k, x_i) + b_+) + \sum_{j=p+1}^{p+q} \delta_j y_j (\sum_{k=1}^{l} u_{+k} K(x_k, x_j) + b_+) \\
\text{s.t.} & \quad \sum_{k=1}^{l} u_{+k} K(x_k, x_i) + b_+ \leq \varepsilon + \eta_i, \quad i = 1, \cdots, p, \\
& - \sum_{k=1}^{l} u_{+k} K(x_k, x_i) - b_+ \leq \varepsilon + \eta_i^*, \quad i = 1, \cdots, p, \\
& \sum_{k=1}^{l} u_{+k} K(x_k, x_j) + b_+ \leq -1 + \xi_j, \quad j = p + 1, \cdots, p + q, \\
& \eta_i, \eta_i^*, \xi_j \geq 0, \quad i = 1, \cdots, p, j = p + 1, \cdots, p + q 
\end{align*}
\]  

(24)

The CCCP framework for the nonlinear case is similar to Algorithm 1 and only the subproblems is different, the decision functions constructed are

\[
\begin{align*}
f_+(x) = \sum_{i=1}^{l} u_{+i} K(x_i, x); \quad f_-(x) = \sum_{i=1}^{l} u_{-i} K(x_i, x); 
\end{align*}
\]  

(25)

for the new point \( x \in \mathbb{R}^n \), it is predicted to the Class by

\[
\text{Class} = \arg \min_{m=+, -,} f_m(x) 
\]  

(26)

4 Experimental Results

In this section, in order to validate the performance of our ramp-LPNPSVM, we compare it with RNPSVM on several publicly available benchmark datasets which are used in [11]. All methods are implemented in MATLAB 2010 on a PC with an Intel Core I5 processor and 2GB RAM. All methods are solved by the optimization toolbox. For each data set, we randomly select the same number of samples from different classes to compose a balanced training set, therefore, based on this set to verify the above methods. This procedure is repeated 5 times, and Table 1 lists the average tenfold cross-validation results of these methods in terms of accuracy (The results of RNPSVM are reported in [11]. The parameters are chosen to be the same as used in RNPSVM, where the parameters \( l, s \) of ramp loss are set...
\[ t \in (\varepsilon, 1), \; s \in (-1, 1). \] The best test accuracies are in boldface. From the results we can find that the ramp-LPNPSVM gets the accuracy as good as RNPSVM, while it runs faster since we used the linear programming toolbox.

<table>
<thead>
<tr>
<th>Data Sets</th>
<th>ramp-LPNPSVM Accuracy</th>
<th>RNPSVM Accuracy</th>
</tr>
</thead>
<tbody>
<tr>
<td>Australian</td>
<td>84.57 ± 3.25</td>
<td>86.81 ± 3.19</td>
</tr>
<tr>
<td>BUPA liver</td>
<td>73.26 ± 2.83</td>
<td>74.65 ± 2.66</td>
</tr>
<tr>
<td>CMC</td>
<td>75.64 ± 3.19</td>
<td>76.32 ± 4.47</td>
</tr>
<tr>
<td>Heart-Statlog</td>
<td>85.71 ± 3.27</td>
<td>87.03 ± 3.41</td>
</tr>
<tr>
<td>Hepatitis</td>
<td>83.05 ± 3.22</td>
<td>85.27 ± 3.18</td>
</tr>
<tr>
<td>Ionosphere</td>
<td>89.28 ± 2.47</td>
<td>90.12 ± 3.04</td>
</tr>
<tr>
<td>Pima Indian</td>
<td>78.36 ± 3.55</td>
<td>79.68 ± 4.53</td>
</tr>
<tr>
<td>Sonar</td>
<td>88.27 ± 3.37</td>
<td>89.69 ± 5.19</td>
</tr>
<tr>
<td>Votes</td>
<td>95.01 ± 2.54</td>
<td>95.97 ± 4.38</td>
</tr>
<tr>
<td>WPBC</td>
<td>84.96 ± 3.61</td>
<td>86.11 ± 3.06</td>
</tr>
</tbody>
</table>

Table 1: The average tenfold cross-validation results on UCI data sets in terms of accuracy

5 Conclusion

In this paper, we have proposed a ramp loss linear programming NPSVM, termed ramp-LPNPSVM, by introducing the \( l_1 \) regularization term to the ramp loss NPSVM, which involves a sequence of linear programming problems in the CCCP procedure. Compared with the RNPSVM, ramp-LPNPSVM not only has the advantages of RNPSVM, but also has the less training time. Experimental results on benchmark datasets confirm the effectiveness of the proposed algorithm. In [6], Johan A.K. Suykens et. al proposed the ramp-LPSVM and pointed out that the problem related to ramp-LPSVM leads to a polyhedral concave problem which is easier to handle, and they established algorithms including DC programming for local minimization and hill detouring for global search. Since ramp-LPNPSVM is similar to ramp-LPSVM, so we will consider their method in the future.

6 Acknowledgments

This work has been partially supported by grants from National Natural Science Foundation of China (No.61472390, No.11271361, No.71331005, No.11226089), the Major International (Regional) Joint Research Project (No.71110107026) and the Beijing Natural Science Foundation (No. 1162005).

References


