

## L1-TWSVM

Lecturer: Shuning Wang, Li Li      swang@tsinghua.edu.cn   li-li@tsinghua.edu.cn

Student: 闫茹钰, 喻望, 张芙作, 郑洁, 朱榕平

## 1 TWSVM

TWSVM (孪生支持向量机) 是 Jayadeva 等人于 2007 年提出的一种改进的双分界面支持向量机, 用于解决二分类问题。与传统的支持向量机 (SVM) 不同, TWSVM 为每一类的数据点单独建立一个分类面, 其优化策略为, 使同一类的数据点尽可能集中的围绕在该类分类面的周围, 并且远离另一类数据的分类面。所以 TWSVM 需要解决两个二次规划问题, 得到两个不平行的分类面, 但是同一类的数据要作为另一个二次规划问题的约束条件, 反之亦然。

TWSVM 需要求解以下两个二次优化问题:

$$\begin{aligned} \min_{\mathbf{w}_1, b_1} \quad & \frac{1}{2} \|\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1\|_2^2 + c_1 \mathbf{e}_2^T \mathbf{q}_1 \\ \text{s.t.} \quad & -(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0 \end{aligned} \quad (1)$$

$$\begin{aligned} \min_{\mathbf{w}_2, b_2} \quad & \frac{1}{2} \|\mathbf{B}\mathbf{w}_2 + \mathbf{e}_2 b_2\|_2^2 + c_2 \mathbf{e}_1^T \mathbf{q}_2 \\ \text{s.t.} \quad & (\mathbf{A}\mathbf{w}_2 + \mathbf{e}_1 b_2) + \mathbf{q}_2 \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0 \end{aligned} \quad (2)$$

其中,  $\mathbf{A}_{m_1 \times n} = (\mathbf{a}_1^{(1)}, \mathbf{a}_2^{(1)}, \dots, \mathbf{a}_{m_1}^{(1)})^T$  表示  $m_1$  个正样本,  $\mathbf{B}_{m_2 \times n} = (\mathbf{b}_1^{(2)}, \mathbf{a}_2^{(2)}, \dots, \mathbf{a}_{m_2}^{(2)})^T$  表示  $m_2$  个负样本,  $\mathbf{e}_1$  和  $\mathbf{e}_2$  表示相应维数的单位变量,  $\|\cdot\|_2$  表示 L2 范数,  $\mathbf{q}_1, \mathbf{q}_2$  是松弛向量,  $c_1, c_2$  是非负惩罚系数, 分别为正样本和负样本的平衡因子, 可以用来解决正负样本个数不同的问题。通过求解以上两个优化问题, 可以分别得到两个不平行的超平面:

$$\mathbf{x}^T \mathbf{w}_1 + b_1 = 0, \quad \mathbf{x}^T \mathbf{w}_2 + b_2 = 0 \quad (3)$$

当有一个新的点  $\mathbf{x}$ , 计算其到两个超平面的垂直距离, 如果它距离超平面  $\mathbf{x}^T \mathbf{w}_1 + b_1 = 0$  的距离小于它到超平面  $\mathbf{x}^T \mathbf{w}_2 + b_2 = 0$  的距离, 则将该点归入正类, 否则它属于负类。

我们也可以得到问题 (1)、(2) 的 Wolfe 对偶问题:

$$\begin{aligned} \max_{\alpha} \quad & \mathbf{e}_2^T \alpha - \frac{1}{2} \alpha^T \mathbf{G} (\mathbf{H}^T \mathbf{H})^{-1} \mathbf{G}^T \alpha \\ \text{s.t.} \quad & 0 \leq \alpha \leq c_1 \mathbf{e}_2 \end{aligned} \quad (4)$$

$$\begin{aligned} \max_{\beta} \quad & \mathbf{e}_1^T \beta - \frac{1}{2} \beta^T \mathbf{H} (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{H}^T \beta \\ \text{s.t.} \quad & 0 \leq \beta \leq c_2 \mathbf{e}_1 \end{aligned} \quad (5)$$

其中  $\alpha \in R^{m_2}$  和  $\beta \in R^{m_1}$  是拉格朗日乘子, 可以利用  $\alpha$  和  $\beta$  得到两个不平行的超平面:

$$\begin{aligned} \mathbf{z}_1 &= (\mathbf{w}_1^T b_1)^T = -(\mathbf{H}^T \mathbf{H})^{-1} \mathbf{G}^T \alpha \\ \mathbf{z}_2 &= (\mathbf{w}_2^T b_2)^T = (\mathbf{G}^T \mathbf{G})^{-1} \mathbf{H}^T \beta \end{aligned} \quad (6)$$

由于逆矩阵  $(\mathbf{H}^T \mathbf{H})^{-1}$  和  $(\mathbf{G}^T \mathbf{G})^{-1}$  可能带来奇异问题, 为防止矩阵奇异, 可以加入一个正则项  $\varepsilon \mathbf{I}$ ,  $\varepsilon$  是一个足够小的正数。这样可以保证  $(\mathbf{H}^T \mathbf{H} + \varepsilon \mathbf{I})^{-1}$  和  $(\mathbf{G}^T \mathbf{G} + \varepsilon \mathbf{I})^{-1}$  正定, 从而不会出现奇异问题。

图 1 是对 TWSVM 的几何解释。

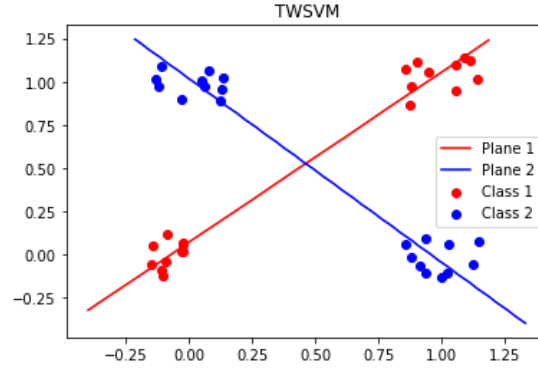


图 1: TWSVM

## 2 L1-TWSVM

TWSVM 有良好的分类性能, 已成为数据分类研究的热点。但 TWSVM 使用对离群值较为敏感的 L2 范数来度量距离, 导致异常观测点可能会对其结果有较大影响。由于 L1 范数是 L2 范数距离的鲁棒替代, (Yan, Ye,

and Yu, 2018) 提出基于 L1 范数的鲁棒分类器。优化问题如下：

$$\begin{aligned} \min_{\mathbf{w}_1, b_1} \quad & \frac{1}{2} \|\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1\|_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1 \\ \text{s.t.} \quad & -(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0 \end{aligned} \quad (7)$$

$$\begin{aligned} \min_{\mathbf{w}_2, b_2} \quad & \frac{1}{2} \|\mathbf{B}\mathbf{w}_2 + \mathbf{e}_2 b_2\|_1 + c_2 \mathbf{e}_1^T \mathbf{q}_2 \\ \text{s.t.} \quad & (\mathbf{A}\mathbf{w}_2 + \mathbf{e}_1 b_2) + \mathbf{q}_2 \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0 \end{aligned} \quad (8)$$

其中  $\|\cdot\|_1$  表示 L1 范数。在最小化目标函数时，每个平面要尽可能靠近两个分类中的一类，并尽可能远离另一类。由于公式 (7)、(8) 中不等式为非凸约束，具有局部最优解，可以求解得到两个不平行的超平面：

$$\mathbf{x}^T \mathbf{w}_1 + b_1 = 0, \mathbf{x}^T \mathbf{w}_2 + b_2 = 0 \quad (9)$$

则原问题可优化为：

$$\min_{\mathbf{w}_1, b_1} \quad \frac{1}{2} \left( \sum_{i=1}^{m_1} \frac{(\mathbf{a}_i^T \mathbf{w}_1 + e_1^i b_1)^2}{d_i} \right) + c_1 \mathbf{e}_2^T \mathbf{q}_1 \quad (10)$$

$$\text{s.t.} \quad -(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0$$

$$\min_{\mathbf{w}_2, b_2} \quad \frac{1}{2} \left( \sum_{j=1}^{m_2} \frac{(\mathbf{b}_j^T \mathbf{w}_2 + e_2^j b_2)^2}{d_j} \right) + c_2 \mathbf{e}_1^T \mathbf{q}_2 \quad (11)$$

$$\text{s.t.} \quad (\mathbf{A}\mathbf{w}_2 + \mathbf{e}_1 b_2) + \mathbf{q}_2 \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0$$

其中  $d_i = |\mathbf{a}_i^T \mathbf{w}_1 + e_1^i b_1| \neq 0$ ,  $d_j = |\mathbf{b}_j^T \mathbf{w}_2 + e_2^j b_2| \neq 0$ ,  $e_1^i, e_2^j$  分别表示  $\mathbf{e}_1$  的第  $i$  个元素和  $\mathbf{e}_2$  的第  $j$  个元素。由于上述两个式子都包含绝对值运算，难以直接求解，本文提出了一种迭代凸优化策略，基本思想为迭代更新增广向量  $\mathbf{z}_1$  直到连续两次迭代式 (10) 的目标值小于一个固定值 (如 0.001)，则  $\mathbf{z}_1$  为局部最优解。记  $\mathbf{z}_1^p$  为第  $p$  次迭代结果，则第  $p+1$  次迭代结果  $\mathbf{z}_1^{(p+1)}$  可等价如下述问题的解：

$$\min_{\mathbf{z}_1} \quad \frac{1}{2} \left( \sum_{i=1}^{m_1} \frac{(\mathbf{h}_i^T \mathbf{z}_1)^2}{d_{1i}} \right) + c_1 \mathbf{e}_2^T \mathbf{q}_1 \quad (12)$$

$$\text{s.t.} \quad -(\mathbf{G}\mathbf{z}_1 + \mathbf{q}_1) \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0$$

$$\min_{\mathbf{z}_2} \quad \frac{1}{2} \left( \sum_{j=1}^{m_2} \frac{(\mathbf{g}_j^T \mathbf{z}_2)^2}{d_{2j}} \right) + c_2 \mathbf{e}_1^T \mathbf{q}_2 \quad (13)$$

$$\text{s.t.} \quad (\mathbf{H}\mathbf{z}_2 + \mathbf{q}_2) \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0$$

其中  $d_{1i} = |\mathbf{h}_i^T \mathbf{z}_1^p|$ ,  $d_{2j} = |\mathbf{g}_j^T \mathbf{z}_2^p|$ ,  $\mathbf{g}_j^T = (\mathbf{b}_j^T \mathbf{e}_2^j)$ , 则公式 (12)、(13) 可改写为

$$\min_{\mathbf{z}_1} \quad \frac{1}{2} \mathbf{z}_1^T \mathbf{H}^T \mathbf{D}_1 \mathbf{H} \mathbf{z}_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1 \quad (14)$$

$$\text{s.t.} \quad -(\mathbf{G}\mathbf{z}_1 + \mathbf{q}_1) \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0$$

$$\min_{\mathbf{z}_2} \quad \frac{1}{2} \mathbf{z}_2^T \mathbf{H}^T \mathbf{D}_2 \mathbf{G} \mathbf{z}_2 + c_2 \mathbf{e}_1^T \mathbf{q}_2 \quad (15)$$

$$\text{s.t.} \quad (\mathbf{H}\mathbf{z}_2 + \mathbf{q}_2) \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0$$

其中  $\mathbf{D}_1 = \text{diag}(1/d_{11}, 1/d_{12}, \dots, 1/d_{1m_1})$ ,  $\mathbf{D}_2 = \text{diag}(1/d_{21}, 1/d_{22}, \dots, 1/d_{2m_2})$  为对角矩阵。则问题 (14)、(15) 等价于:

$$\min_{\mathbf{z}_1} \frac{1}{2} \|\mathbf{H}\mathbf{z}_1\|_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1 \quad (16)$$

$$s.t. \quad -(\mathbf{G}\mathbf{z}_1 + \mathbf{q}_1) \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0$$

$$\min_{\mathbf{z}_2} \frac{1}{2} \|\mathbf{G}\mathbf{z}_2\|_1 + c_2 \mathbf{e}_1^T \mathbf{q}_2 \quad (17)$$

$$s.t. \quad (\mathbf{H}\mathbf{z}_2 + \mathbf{q}_2) \geq \mathbf{e}_1, \mathbf{q}_2 \geq 0$$

公式 (7) 是不等式约束 (非凸) 的凸优化问题, 因此它存在解析解。其拉格朗日函数为:

$$\begin{aligned} L_1(\mathbf{w}_1, b_1, \mathbf{q}_1, \boldsymbol{\alpha}, \boldsymbol{\beta}) = & \frac{1}{2}(\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1)^T \mathbf{D}_1 (\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1) \\ & + c_1 \mathbf{e}_2^T \mathbf{q}_1 - \boldsymbol{\alpha}^T (-(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 - \mathbf{e}_2) - \boldsymbol{\beta}^T \mathbf{q}_1 \end{aligned} \quad (18)$$

其中  $\boldsymbol{\alpha} = (\alpha_1, \alpha_2, \alpha_3, \dots, \alpha_{m_2})^T$ ,  $\boldsymbol{\beta} = (\beta_1, \beta_2, \beta_3, \dots, \beta_{m_1})^T$  为拉格朗日乘子,  $\boldsymbol{\alpha} \geq 0, \boldsymbol{\beta} \geq 0$ , 令  $L_1$  对  $\mathbf{w}_1, b_1, \mathbf{q}_1$  的偏导分别为 0, 可得 Karush-Kuhn-Tucker (KKT) 条件为:

$$\frac{\partial L}{\partial \mathbf{w}_1} = \mathbf{A}^T \mathbf{D}_1 (\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1) + \mathbf{B}^T \boldsymbol{\alpha} = 0 \quad (19)$$

$$\frac{\partial L}{\partial b_1} = \mathbf{e}_1^T \mathbf{D}_1 (\mathbf{A}\mathbf{w}_1 + \mathbf{e}_1 b_1) + \mathbf{e}_2 \boldsymbol{\alpha} = 0 \quad (20)$$

$$\frac{\partial L}{\partial \mathbf{q}_1} = c_1 \mathbf{e}_2 - \boldsymbol{\alpha} - \boldsymbol{\beta} = 0 \quad (21)$$

$$-(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 \geq \mathbf{e}_2, \mathbf{q}_1 \geq 0 \quad (22)$$

$$\boldsymbol{\alpha}^T (-(\mathbf{B}\mathbf{w}_1 + \mathbf{e}_2 b_1) + \mathbf{q}_1 - \mathbf{e}_2) = 0, \boldsymbol{\beta}^T \mathbf{q}_1 = 0 \quad (23)$$

从式 (21) 可推出  $0 \leq \boldsymbol{\alpha} \leq c_1 \mathbf{e}_2$ , 结合公式 (19)、(20) 可得:

$$\mathbf{A}^T \mathbf{e}_1^T \mathbf{D}_1 (\mathbf{A} \mathbf{e}_1) (\mathbf{w}_1 b_1)^T + (\mathbf{B}^T \mathbf{e}_2^T) \boldsymbol{\alpha} = 0 \quad (24)$$

结合之前定义的矩阵  $(\mathbf{H}, \mathbf{G})$  及增广向量  $(\mathbf{z}_1, \mathbf{z}_2)$ , 可以得到

$$\mathbf{H}^T \mathbf{D}_1^p \mathbf{H} \mathbf{z}_1^{(p+1)} + \mathbf{G}^T \boldsymbol{\alpha} = 0 \quad (25)$$

即

$$\mathbf{z}_1^{(p+1)} = -(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H})^{-1} \mathbf{G}^T \boldsymbol{\alpha} \quad (26)$$

由于  $(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H})^{-1}$  为半正定矩阵, 因此可能得到不稳定或不准确的解, 在实际应用中, 本文使用正则化方法解决这个问题。  $(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H} + \varepsilon \mathbf{I})$  为正定矩阵 (其中  $\varepsilon$  为一个小扰动), 不受奇点的影响。则逆矩阵  $(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H})^{-1}$  可由  $(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H} + \varepsilon \mathbf{I})$  代替, 因此  $\mathbf{z}_1^{(p+1)}$  可推导为:

$$\mathbf{z}_1^{(p+1)} = -(\mathbf{H}^T \mathbf{D}_1^p \mathbf{H} + \varepsilon \mathbf{I})^{-1} \mathbf{G}^T \boldsymbol{\alpha} \quad (27)$$

同样的

$$\mathbf{z}_2^{(p+1)} = -(\mathbf{G}^T \mathbf{D}_2^p \mathbf{G} + \varepsilon \mathbf{I})^{-1} \mathbf{H}^T \boldsymbol{\beta} \quad (28)$$

将增广向量  $\mathbf{z}_1^{(p+1)}, \mathbf{z}_2^{(p+1)}$  分别代入拉格朗日函数中。在 KKT 条件下, 原问题 (7)、(8) 转变为 Wolfe 对偶问题:

$$\begin{aligned} \max_{\boldsymbol{\alpha}} \quad & \mathbf{e}_2^T \boldsymbol{\alpha} - \frac{1}{2} \boldsymbol{\alpha}^T \mathbf{G} (\mathbf{H}^T \mathbf{D}_1 \mathbf{H})^{-1} \mathbf{G}^T \boldsymbol{\alpha} \\ \text{s.t.} \quad & 0 \leq \boldsymbol{\alpha} \leq c_1 \mathbf{e}_2 \end{aligned} \quad (29)$$

$$\begin{aligned} \max_{\boldsymbol{\beta}} \quad & \mathbf{e}_1^T \boldsymbol{\beta} - \frac{1}{2} \boldsymbol{\beta}^T \mathbf{H} (\mathbf{G}^T \mathbf{D}_2 \mathbf{G})^{-1} \mathbf{H}^T \boldsymbol{\beta} \\ \text{s.t.} \quad & 0 \leq \boldsymbol{\beta} \leq c_2 \mathbf{e}_1 \end{aligned} \quad (30)$$

通过求解对偶问题可得拉格朗日乘子  $\boldsymbol{\alpha} \in \mathbf{R}^{m_2 \times 1}, \boldsymbol{\beta} \in \mathbf{R}^{m_1 \times 1}$  以及权向量  $\mathbf{w}_1, \mathbf{w}_2$ 、偏差  $b_1, b_2$ , 即获得两个不平行的超平面。对于新加入的点  $\mathbf{x} \in \mathbf{R}^n$ , 根据决策方程  $f(\mathbf{x})$  (选择最近的超平面) 将其分配到对应类别中

$$f(\mathbf{x}) = \arg \min_{i=1,2} (|\mathbf{x}^T \mathbf{w}_i + b_i| / \|\mathbf{w}_i\|) \quad (31)$$

其中  $|\cdot|$  表示取绝对值。式 (7) 中的目标函数为非凸约束的凸问题, 因此  $\mathbf{z}_1^{(p+1)}$  为此问题的局部最优解。而在式 (26) 中,  $\mathbf{D}_1^p$  依赖于  $\mathbf{z}_1^{(p+1)}$ , 因此它是一个未知变量, 可看作 (7) 中目标的隐变量, 可以同相同的迭代算法交替优化求解。我们根据前一次迭代结果  $\mathbf{z}_1^{(p+1)}$  来更新  $\mathbf{D}_1^p$ , 又通过  $\mathbf{D}_1^p$  来改变  $\mathbf{z}_1^{(p+1)}$ , 增加  $p$  直到连续两次迭代结果小于一个固定值。此外, 适当的初始化可有效加快算法收敛速度。本文通过求解公式 (1)、(2) 得到初始解, 仿真结果较优。算法 1 总结了 L1-TWSVM 的迭代过程。

**Input:**  $A \in \mathbf{R}^{m_1 \times n}$  and  $B \in \mathbf{R}^{m_2 \times n}$ ;

Construct the matrices  $H = (A \ e_1)$  and  $G = (B \ e_2)$ ;

Set  $p = 0$ . Initialize  $z^p$ , a standard solution of TWSVM;

**while not converge do**

    Compute  $D_1^p$ ;

    Compute  $z_1^{(p+1)}$  by solving

$$z_1^{(p+1)} = \arg \min_{z_1} \frac{1}{2} z_1^T H^T D_1^p H z_1 + c_1 e_2^T q_1, \text{ s.t. } -G z_1 + q_1 \geq e_2, q_1 \geq 0 \quad (32)$$

$p = p + 1$ ;

**end**

**Output:** The learned solution of  $z_1$ .

**算法 1:** L1-TWSVM

### 3 收敛性证明

**引理 3.1.** 对任意非零向量  $\mathbf{u}, \mathbf{u}^p \in \mathbf{R}^1$ , 有以下不等式:

$$\|\mathbf{u}\|_1 - \frac{\|\mathbf{u}\|_1^2}{2\|\mathbf{u}^p\|_1} \leq \|\mathbf{u}^p\|_1 - \frac{\|\mathbf{u}^p\|_1^2}{2\|\mathbf{u}^p\|_1} \quad (33)$$

证明：

$$\begin{aligned} (\sqrt{\mathbf{v}} - \sqrt{\mathbf{v}^p})^2 &\geq 0 \Rightarrow \mathbf{v} - 2\sqrt{\mathbf{v}\mathbf{v}^p} + \mathbf{v}^p \geq 0 \\ \Rightarrow \sqrt{\mathbf{v}} - \frac{\mathbf{v}}{2\sqrt{\mathbf{v}^p}} &\leq \frac{\sqrt{\mathbf{v}^p}}{2} \Rightarrow \sqrt{\mathbf{v}} - \frac{\mathbf{v}}{2\sqrt{\mathbf{v}^p}} \leq \sqrt{\mathbf{v}^p} - \frac{\mathbf{v}^p}{2\sqrt{\mathbf{v}^p}} \end{aligned} \quad (34)$$

将式 (34) 中的  $\mathbf{v}$  和  $\mathbf{v}^p$  替换为  $\|\mathbf{u}\|_1^2$  和  $\|\mathbf{u}^p\|_1^2$ ，即得到式 (33)。

**定理 3.2.** 算法 1 在每步迭代中都使问题 (14) 的目标值单调递减。

证明：首先，用以下等式重写 (32) 中的问题：

$$\mathbf{z}_1^{(p+1)} = \arg \min_{\mathbf{z}_1} \frac{1}{2} \mathbf{z}_1^T \mathbf{H}^T \mathbf{D}_1^T \mathbf{H} \mathbf{z}_1 + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1) \quad (35)$$

即：

$$\mathbf{z}_1^{(p+1)} = \arg \min_{\mathbf{z}_1} \frac{1}{2} (\mathbf{H} \mathbf{z}_1)^T \mathbf{D}_1^p \mathbf{H} \mathbf{z}_1 + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1) \quad (36)$$

因此，在第  $(p+1)$  步迭代中，有

$$\begin{aligned} &\frac{1}{2} (\mathbf{H} \mathbf{z}_1^{(p+1)})^T \mathbf{D}_1^p (\mathbf{H} \mathbf{z}_1^{(p+1)}) + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1^{(p+1)}) \\ &\leq \frac{1}{2} (\mathbf{H} \mathbf{z}_1^p)^T \mathbf{D}_1^p (\mathbf{H} \mathbf{z}_1^p) + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1^p) \end{aligned} \quad (37)$$

将式 (33) 中的  $\mathbf{u}$  和  $\mathbf{u}^p$  替换为  $\mathbf{H} \mathbf{z}_1^{(p+1)}$  和  $\mathbf{H} \mathbf{z}_1^p$ ，可以得到：

$$\|\mathbf{H} \mathbf{z}_1^{(p+1)}\|_1 - \frac{\|\mathbf{H} \mathbf{z}_1^{(p+1)}\|_1^2}{2\|\mathbf{H} \mathbf{z}_1^p\|_1} \leq \|\mathbf{H} \mathbf{z}_1^p\|_1 - \frac{\|\mathbf{H} \mathbf{z}_1^p\|_1^2}{2\|\mathbf{H} \mathbf{z}_1^p\|_1} \quad (38)$$

因此可得如下不等式：

$$\sum_{i=1}^{m_1} \left( |\mathbf{h}_i^T \mathbf{z}_1^{(p+1)}| - \frac{(\mathbf{h}_i^T \mathbf{z}_1^{(p+1)})^2}{2|\mathbf{h}_i^T \mathbf{z}_1^p|} \right) \leq \sum_{i=1}^{m_1} \left( |\mathbf{h}_i^T \mathbf{z}_1^p| - \frac{(\mathbf{h}_i^T \mathbf{z}_1^p)^2}{2|\mathbf{h}_i^T \mathbf{z}_1^p|} \right) \quad (39)$$

该式可被简化为：

$$\begin{aligned} &\|\mathbf{H} \mathbf{z}_1^{(p+1)}\|_1 - \frac{1}{2} (\mathbf{H} \mathbf{z}_1^{(p+1)})^T \mathbf{D}_1^p (\mathbf{H} \mathbf{z}_1^{(p+1)}) \\ &\leq \|\mathbf{H} \mathbf{z}_1^p\|_1 - \frac{1}{2} (\mathbf{H} \mathbf{z}_1^p)^T \mathbf{D}_1^p (\mathbf{H} \mathbf{z}_1^p) \end{aligned} \quad (40)$$

综合式 (37) 和 (40)，可得：

$$\begin{aligned} &\|\mathbf{H} \mathbf{z}_1^{(p+1)}\|_1 + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1^{(p+1)}) \\ &\leq \|\mathbf{H} \mathbf{z}_1^p\|_1 + c_1 \mathbf{e}_2^T \max(0, \mathbf{e}_2 + \mathbf{G} \mathbf{z}_1^p) \end{aligned} \quad (41)$$

因为 (14) 中的问题恒小于零，因此算法 1 收敛，(41) 中的不等式成立。这表示 (14) 中的目标值随迭代递减，直到算法收敛。

**定理 3.3.** 算法 1 收敛至问题 (14) 的一个局部最优解。

证明：问题 (14) 的拉格朗日函数如下：

$$L_2(\mathbf{z}_1, \mathbf{q}_1) = \frac{1}{2} \|\mathbf{H}\mathbf{z}_1\|_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1 - \alpha^T (-\mathbf{G}\mathbf{z}_1 + \mathbf{q}_1 - \mathbf{e}_2) - \beta^T \mathbf{q}_1 \quad (42)$$

其中， $\alpha$  和  $\beta$  是拉格朗日乘子向量。通过对其求导并取零，可以得到问题 (14) 的 KKT 条件：

$$\mathbf{H}^T \mathbf{D}_1 \mathbf{H} \mathbf{z}_1 + \mathbf{G} \alpha = 0, c_1 \mathbf{e}_2 - \alpha - \beta = 0 \quad (43)$$

在算法 1 的每步迭代中，寻找问题 (32) 中的最优  $\mathbf{z}_1^{(p+1)}$ 。因此，算法 1 的收敛解满足问题的 KKT 条件。接下来，定义算法 1 中问题 (32) 的拉格朗日函数如下：

$$L_3(\mathbf{z}_1, \mathbf{q}_1) = \frac{1}{2} \mathbf{H}^T \mathbf{D}_1 \mathbf{H} \mathbf{z}_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1 - \alpha^T (-\mathbf{G}\mathbf{z}_1 + \mathbf{q}_1 - \mathbf{e}_2) - \beta^T \mathbf{q}_1 \quad (44)$$

同样对其求导并取零，得到：

$$\mathbf{H}^T \mathbf{D}_1 \mathbf{H} \mathbf{z}_1 + \mathbf{G} \alpha = 0, c_1 \mathbf{e}_2 - \alpha - \beta = 0 \quad (45)$$

根据算法 1 中  $\mathbf{D}_1$  的定义，等式 (43) 和 (45) 在算法 1 收敛时成立。这说明算法 1 的收敛解  $\mathbf{z}_1^{(p+1)}$  满足问题 (14) 的 KKT 条件，是问题 (14) 的一个局部最优解。

## 4 数值实验

类似论文原文，我们使用异或问题测试 L1-TWSVM 算法的正确性。由于异或问题比较简单，直接采用本地随机生成的数据集。本次实验分别以点 (0, 0), (0, 1), (1, 0), (1, 1) 为中心，范围  $[\pm 0.15, \pm 0.15]$  内均匀产生 10 个样本点；再手动添加两个离群点 (1.2, -0.3), (-0.3, -0.3)。得到数据散点图如图 2 所示。

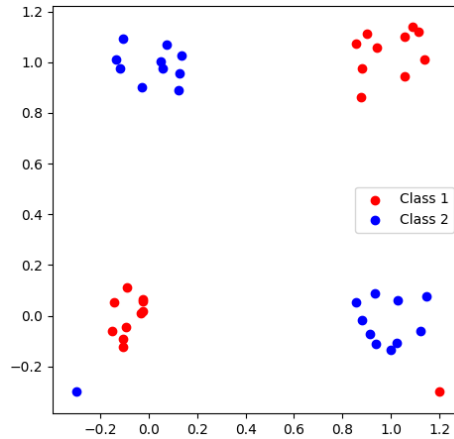


图 2: 数据散点图

实验代码采用 Python 编写，调用 cvxpy (Diamond and Boyd, 2016) 求解凸优化问题，详细代码见附录 A。首先不添加离群点，观察 TWSVM 与 L1-TWSVM 拟合超平面的效果对比，得到输出：

```

Init solution z1: [-0.00934916  0.00960656 -0.00173161]
Init solution z2: [-0.00083704 -0.00064728  0.00173187]
Final solution z1: [-2.1700e-08  2.2172e-08 -4.1194e-09]
Final solution z2: [-2.2670e-09 -1.7231e-09  3.0088e-09]

```

拟合结果如图 3，可见四条拟合直线都存在较大问题。考察优化目标  $\frac{1}{2}\mathbf{z}_1^T \mathbf{H}^T \mathbf{D}_1 \mathbf{H} \mathbf{z}_1 + c_1 \mathbf{e}_2^T \mathbf{q}_1$ ，由于改变超平面系数比例不会影响超平面的法线方向和位置，所以系数尺度越小，目标函数值越小，所以最优解将会具有较小的模长。但是求解问题时由除法操作，在数值计算中，除以较小数将导致较大的误差，因此造成超平面偏离数据点。直观的解决方法是增加  $\|\mathbf{z}_1\|_2 = 1$  的约束，但此时问题将变成非凸。所以最后采用  $\mathbf{z}_1^{(1)} = 1$ ， $\mathbf{z}_1^{(1)}$  即向量首元素，作为新的约束。修改后得到输出：

```

Init solution z1: [ 1. -1.01312805  0.06991737]
Init solution z2: [ 1.  0.93863708 -0.95349977]
Final solution z1: [ 1. -0.99892206  0.04902573]
Final solution z2: [ 1.  0.86698756 -0.90293122]

```

拟合结果如图 4，两个算法结果相近。由于  $l_1$ -范数受较远处点的影响较小，所以蓝色直线没有过点集的重心，这也符合预期。

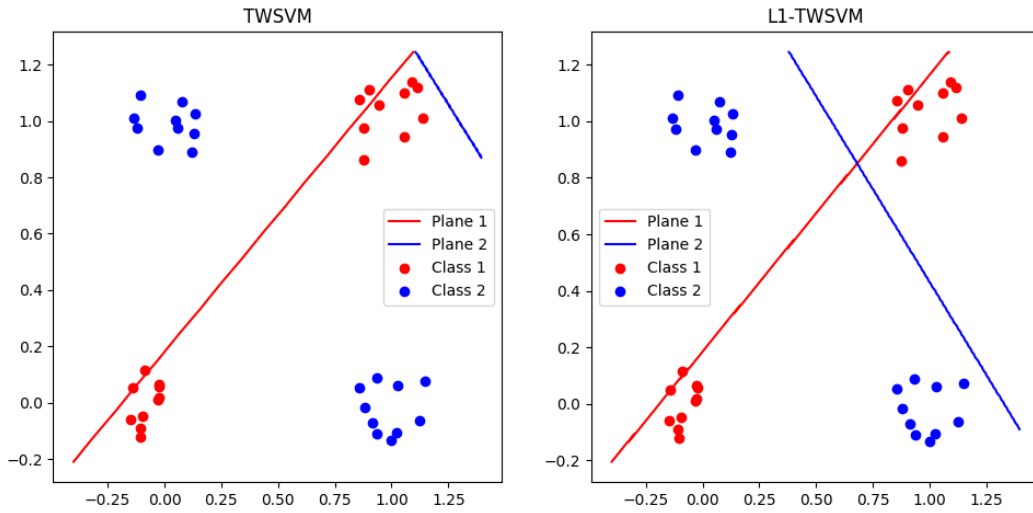


图 3: 不带离群点，无参数规范化

增加离群点再进行实验，拟合结果如图 5。可见离群点使  $l_2$ -范数下的超平面产生较大的偏移，而 L1-TWSVM 拟合的直线受影响较小。实验结果说明，与 TWSVM 相比，L1-TWSVM 确实能够减小离群点对拟合超平面的影响。



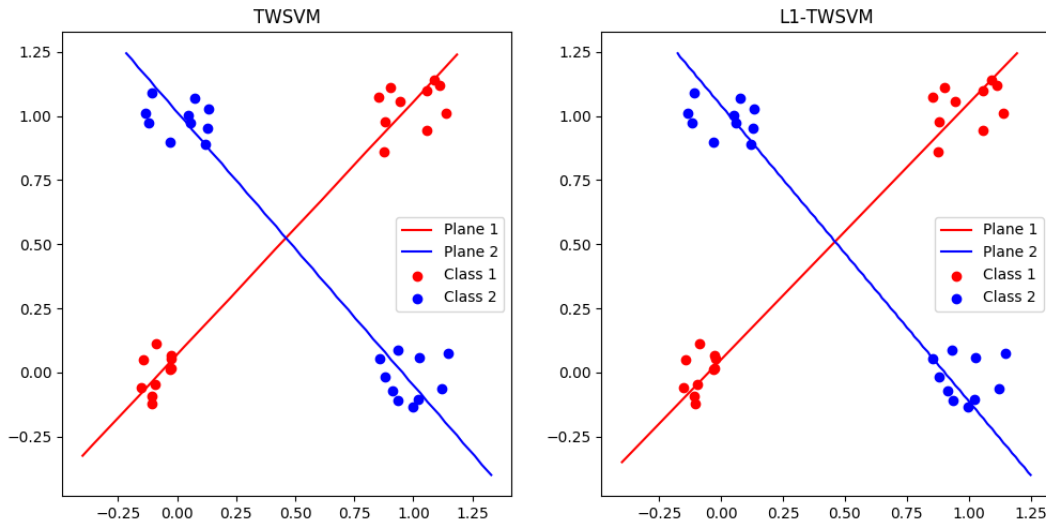


图 4: 不带离群点, 参数规范化

## 5 总结

本次作业本组阅读了 (Yan, Ye, and Yu, 2018), 对其中的算法提出思路, 算法收敛性证明进行了整理。最后使用 Python 进行了数值实验, 验证了算法 L1-TWSVM 的正确性。

## 参考文献

- Diamond, Steven and Stephen Boyd (2016). “CVXPY: A Python-Embedded Modeling Language for Convex Optimization”. In: *Journal of Machine Learning Research* 17.83, pp. 1–5 (cit. on p. 7).
- Yan, He, Qiao-Lin Ye, and Dong-Jun Yu (2018). “Efficient and robust TWSVM classification via a minimum L1-norm distance metric criterion”. In: *Machine Learning*, pp. 1–26 (cit. on pp. 2, 9).

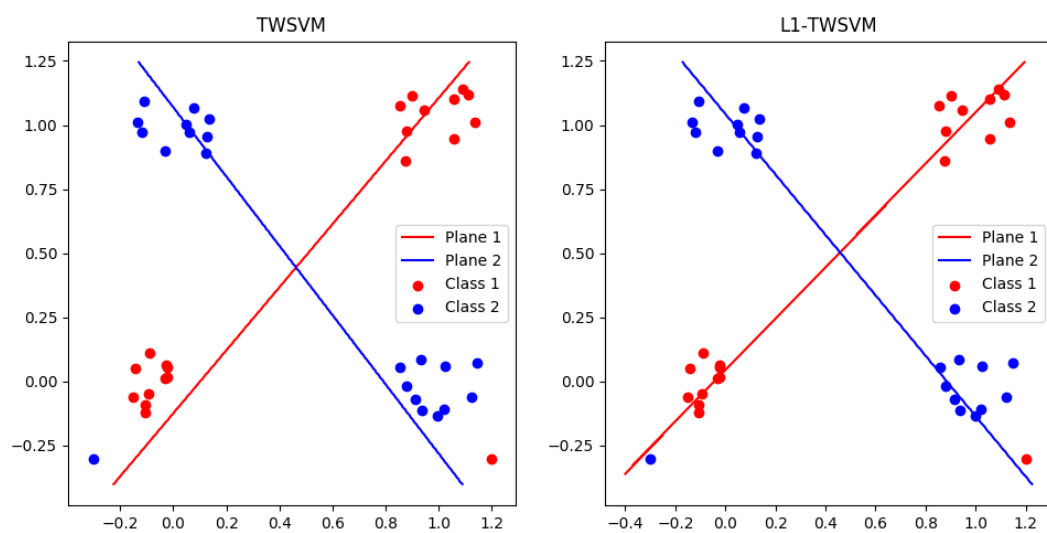


图 5: 带离群点, 参数规范化

## A 实验代码

```

import cvxpy as cp
import numpy as np
import matplotlib.pyplot as plt

# Generate data
np.random.seed(1)
length = 0.3
A = np.vstack([np.array([0, 0]) + length * (-0.5 + np.random.random((10, 2))),
               np.array([1, 1]) + length * (-0.5 + np.random.random((10, 2)))])
B = np.vstack([np.array([1, 0]) + length * (-0.5 + np.random.random((10, 2))),
               np.array([0, 1]) + length * (-0.5 + np.random.random((10, 2)))])
A = np.vstack([A, np.array([1.2, -0.3])])
B = np.vstack([B, np.array([-0.3, -0.3])])
xmin, xmax = -0.4, 1.4
ymin, ymax = -0.4, 1.25

# Parameter Setup
TOL = 1e-3
n = 2
m1 = A.shape[0]
m2 = B.shape[0]
H = np.hstack([A, np.ones((m1, 1))])
G = np.hstack([B, np.ones((m2, 1))])
D1_SR = cp.Parameter((m1, m1))
D2_SR = cp.Parameter((m2, m2))
c1 = cp.Parameter(nonneg=True)
c2 = cp.Parameter(nonneg=True)

# Regularizer Coeff
c1.value = 0.001
c2.value = 0.001

# Construct the problem.
z1 = cp.Variable(n + 1)

```

```

q1 = cp.Variable(m2)
z2 = cp.Variable(n + 1)
q2 = cp.Variable(m1)
init_obj1 = cp.Minimize(cp.sum_squares(H * z1) / 2 + c1 * cp.sum(q1))
init_cons1 = [-G * z1 + q1 >= 1, q1 >= 0]
init_cons1.append(z1[0] == 1)
init_prob1 = cp.Problem(init_obj1, init_cons1)

init_obj2 = cp.Minimize(cp.sum_squares(G * z2) / 2 + c2 * cp.sum(q2))
init_cons2 = [H * z2 + q2 >= 1, q2 >= 0]
init_cons2.append(z2[0] == 1)
init_prob2 = cp.Problem(init_obj2, init_cons2)

objective1 = cp.Minimize(cp.sum_squares(D1_SR * H * z1) / 2 + c1 * cp.sum(q1))
constraints1 = [-G * z1 + q1 >= 1, q1 >= 0]
constraints1.append(z1[0] == 1)
prob1 = cp.Problem(objective1, constraints1)

objective2 = cp.Minimize(cp.sum_squares(D2_SR * G * z2) / 2 + c2 * cp.sum(q2))
constraints2 = [H * z2 + q2 >= 1, q2 >= 0]
constraints2.append(z2[0] == 1)
prob2 = cp.Problem(objective2, constraints2)

# Obtain the initial solution
init_prob1.solve()
z1_last = z1.value
print('Init solution z1:', z1_last)

init_prob2.solve()
z2_last = z2.value
print('Init solution z2:', z2_last)

# Plot points and planes
w1 = z1_last / np.linalg.norm(z1_last)
w2 = z2_last / np.linalg.norm(z2_last)
plane1 = np.array([[x, y] for x in np.arange(xmin, xmax, 0.005)
                    for y in np.arange(ymin, ymax, 0.005)

```

```

        if abs(w1.dot(np.array([x, y, 1]))) < 0.001])
plane2 = np.array([[x, y] for x in np.arange(xmin, xmax, 0.005)
                    for y in np.arange(ymin, ymax, 0.005)
                    if abs(w2.dot(np.array([x, y, 1]))) < 0.001])

plt.subplot(121)
plt.scatter(A[:, 0], A[:, 1], color='r', label='Class 1')
plt.scatter(B[:, 0], B[:, 1], color='b', label='Class 2')
plt.plot(plane1[:, 0], plane1[:, 1], color='r', label='Plane 1')
plt.plot(plane2[:, 0], plane2[:, 1], color='b', label='Plane 2')
plt.title('TWSVM')
plt.legend()

# Iterative procedure
while True:
    D1_SR.value = np.diag(1 / np.sqrt(np.abs(np.dot(H, z1_last))))
    prob1.solve()
    z1_cur = z1.value
    if np.linalg.norm(z1_cur - z1_last) < TOL:
        break
    else:
        z1_last = z1_cur.copy()

while True:
    D2_SR.value = np.diag(1 / np.sqrt(np.abs(np.dot(G, z2_last))))
    prob2.solve()
    z2_cur = z2.value
    if np.linalg.norm(z2_cur - z2_last) < TOL:
        break
    else:
        z2_last = z2_cur.copy()

print('Final solution z1:', z1_cur)
print('Final solution z2:', z2_cur)

z1_cur /= np.linalg.norm(z1_cur)
z2_cur /= np.linalg.norm(z2_cur)
plane1 = np.array([[x, y] for x in np.arange(xmin, xmax, 0.005)

```

```
        for y in np.arange(ymin, ymax, 0.005)
            if abs(z1_cur.dot(np.array([x, y, 1]))) < 0.002])
plane2 = np.array([[x, y] for x in np.arange(xmin, xmax, 0.005)
                    for y in np.arange(ymin, ymax, 0.005)
                    if abs(z2_cur.dot(np.array([x, y, 1]))) < 0.001]])

plt.subplot(122)
plt.scatter(A[:, 0], A[:, 1], color='r', label='Class 1')
plt.scatter(B[:, 0], B[:, 1], color='b', label='Class 2')
plt.plot(plane1[:, 0], plane1[:, 1], color='r', label='Plane 1')
plt.plot(plane2[:, 0], plane2[:, 1], color='b', label='Plane 2')
plt.title('L1-TWSVM')
plt.legend()
plt.show()
```