SMO 算法

谢欣然

待求解的问题

含有松弛因子 ξ 的原始问题：

$$\min_{w,b,\xi} = \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{L}\xi_i$$

$$s.t. \qquad y_i(w\phi(x_i)+b) \geq 1-\xi_i, \qquad i=1,\dots,L \tag{1}$$

$$\xi_i \geq 0, i=1,\dots L$$

问题（1）的对偶问题的矩阵形式：

$$\begin{aligned} &\min_{\alpha} f(\alpha) = \frac{1}{2}\alpha^T Q\alpha - e^T\alpha \\ s.t. \qquad & 0 \leq \alpha_i \leq C \qquad i=1,\dots,L \\ & y^T\alpha = 0 \end{aligned} \tag{2}$$

其中，L是所有样本的数量，e是一个全为1的单位向量，C是惩罚函数，也是α的上界。Q是一个LxL的对称矩阵，$Q_{ij}=y_iy_jK(x_i,x_j)$,其中$K(x_i,x_j)$是核函数。

原始问题的最优解w^*可以根据对偶问题的最优解α^*求解得到，即：

$$w^* = \sum_{i=1}^{L} y_i\alpha_i^*\Phi(x_i) \tag{3}$$

原始问题中的最优解b^* 可以根据KKT条件求得,因为:

$$0 \leq \alpha_i \leq C \rightarrow y_i(w^*\Phi(x_i)+b^*) = 1$$

然后进一步求得：

$$
\begin{aligned}
b^* &= y_i - w^*\Phi(x_i) \\
&= y_i - \sum_{j=1}^{L} y_j a_j^* K(x_j, x_i)
\end{aligned} \tag{4}
$$

接下来我们需要利用SMO算法进行求解对偶问题（2）的最优解α^*

SMO的核心思想是，在一轮迭代中，不是一次更新α的所有分量，而是每次选 中α两个分量i和j，固定其他的分量，在子问题上进行优化。

SMO算法涉及到两个核心的问题:

1. 如何选择i和j
2. 如何更新α，即选择了i和j后，如何更新α_i和α_j使得在(2)中的目标函数$f(\alpha)$变小

如何选择i 和 j

- ## i的选择：

对偶问题（2）的拉格朗日函数：

$$
\begin{gathered}
L(\alpha, \lambda, \mu) = f(\alpha) - \sum_{i=1}^{L} \lambda_i \alpha_i + \sum_{i=1}^{L} \mu_i(\alpha_i - C) + \sum_{i=1}^{L} \eta y^T \alpha \\
\lambda_i > 0 \\
\mu_i > 0
\end{gathered} \tag{5}
$$

如果α为极值点的话，我们可以得到$\Delta(\alpha, \lambda, \mu) = 0$,即：

$$
\Delta f(\alpha) + \eta y = \lambda - \mu \tag{6}
$$

其中：

$$\Delta f(\alpha) = Q\alpha - e$$
$$\Delta f(\alpha)_i = \sum_{i=1}^{L} \alpha_i y_i y_j K(x_i, x_j) - 1 \tag{7}$$

根据KKT的对偶互补条件，可以得到：

$$\lambda_i \alpha_i = 0 \tag{8}$$

$$\mu_i (C - \alpha_i) = 0 \tag{9}$$

根据（6）（8）（9），可以得到：

$$\begin{aligned} \Delta f(\alpha)_i + \eta y_i \geq 0 \qquad if \alpha_i < C \\ \Delta f(\alpha)_i + \eta y_i \leq 0 \qquad if \alpha_i > 0 \end{aligned} \tag{10}$$

接下来我们定义两个集合，这两个集合是关于α_t下标t的集合:

$$\begin{aligned} I_{up}(\alpha) = \{t | \alpha_t < C, y_t = 1 \qquad or \qquad \alpha_t > 0, y_t = -1\} \\ I_{low}(\alpha) = \{t | \alpha_t < C, y_t = -1 \qquad or \qquad \alpha_t > 0, y_t = 1\} \end{aligned} \tag{11}$$

结合上式，可以改写（10）式（两边同乘y_i）：

$$\begin{aligned} -y_i \Delta f(\alpha)_i \leq \eta \qquad \forall_i \in I_{up}(\alpha) \\ -y_j \Delta f(\alpha)_j \geq \eta \qquad \forall_j \in I_{low}(\alpha) \end{aligned} \tag{11}$$

即有：

$$-y_i \Delta f(\alpha)_i \leq \eta \leq -y_j \Delta f(\alpha)_j, \qquad i \in I_{up} \qquad j \in I_{low} \tag{12}$$

我们令：

$$\begin{aligned} m(\alpha) = \max_{i \in I_{up}(\alpha)} -y_i \Delta f(\alpha)_i \\ M(\alpha) = \min_{j \in I_{low}(\alpha)} -y_i \Delta f(\alpha)_j \end{aligned} \tag{13}$$

一个合理的驻点α应该满足：

$$m(\alpha) \leq M(\alpha) \tag{14}$$

下面我们给出一个违反对的定义：

如果$i \in I_{up}$，$j \in I_{low}$ 并且$-y_i\Delta f(\alpha)_i > -y_j\Delta f(\alpha)_j$,那么$\{\alpha_i, \alpha_j\}$就是一个违反对。一个普遍的做法是每次迭代挑选“最大违反对”，即：

$$\begin{aligned} i \in arg \quad \max_t \{-y_t\Delta f(\alpha^k)_t | t \in I_{up}(\alpha^k)\} \\ j \in arg \quad \max_t \{-y_t\Delta f(\alpha^k)_t | t \in I_{low}(\alpha^k)\} \end{aligned} \tag{15}$$

因为最优的α需要满足$m(\alpha) \leq M(\alpha)$，如果违反了，它就是不是最优的，所以每次找出这个最大违反对，然后更新它们的值.

最后我们再设置一个停止条件，即$m(\alpha^k) - M(\alpha^k) < \epsilon$，这里$\epsilon$称为容忍值，如果在第k轮的时候，$m(\alpha^k)$没有比$M(\alpha^k)$大太多的话，就停止。

- ## j的选择：

LIBSVM里面对于违反对的选择对上述进行性了改进，其中α_i的选择和上面一样，但是α_j的选择除了要满足其是违反对以外，还需要它能够使目标函数减少的最多。

根据泰勒公式在α^k处二阶展开，可以得到：

$$\begin{aligned} f(\alpha^k + d) &= f(\alpha^k) + \Delta f(\alpha^k)^T d + \frac{1}{2} d^T \Delta^2 f(\alpha^k) d \\ &= f(\alpha^k) + \Delta f(\alpha^k)_B^T d_B + \frac{1}{2} d_B^T \Delta^2 f(\alpha^k)_{BB} d_B \end{aligned} \tag{16}$$

其中B = {i,j}，注意这里的变量只有α_i和α_j，所以对其他α_t,$t \neq i$ and $t \neq i$的导数为0，直接舍去，我们只取含有B的分量。此时，我们定义一个子问题，这个子问题只B有关，即和α_i和α_j有关：

$$\begin{aligned} Sub(B) &= f(\alpha^k + d) - f(\alpha^k) \\ &= \Delta f(\alpha^k)_B^T d_B + \frac{1}{2} d_B^T \Delta^2 f(\alpha^k)_{BB} d_B \end{aligned} \tag{17}$$

我们的目标就是让Sub(B)越小越好，即希望α_k加上偏移量d，$f(\alpha^k + d)$的值能够比更新之前$f(\alpha^k)$要小，也就是$f(\alpha^k + d) << f(\alpha^k)$，整个问题就变成:

$$\begin{gathered} \min_{d_B} \quad \Delta f(\alpha^k)_B^T d_B + \frac{1}{2} d_B^T \Delta^2 f(\alpha^k)_{BB} d_B \\ s.t. \quad y_B^T d_B = 0 \\ d_t \geq 0, if a_t^k = 0, t \in B \end{gathered}$$

$$d_t \leq 0, if a_t^k = C, t \in B \tag{18}$$

这里解释下以及在(18)的约束条件，因为:

$$\begin{cases} y_i\alpha_i + y_j\alpha_j = Constant \\ y_i(\alpha_i + d_i) + y_j(\alpha_j + d_j) = Constant \end{cases}$$

所以：
$y_i d_i + y_j d_j = 0 \Rightarrow y_B^T d_B = 0$

又因为：

$$\begin{cases} 0 \leq \alpha_i \leq C \\ 0 \leq \alpha_i + d_i \leq C \end{cases}$$

所以：

$$-\alpha_i \leq d_i \leq C - \alpha_i$$

定理:

如果B={i, j}是一个违反对且$K_{ii} + K_{jj} - 2K_{ij} > 0$，上述问题能够取得的最小值是:

$$-\frac{(-y_i \Delta f(\alpha^k)_i + y_i \Delta f(\alpha^k)_j)^2}{2(K_{ii} + K_{jj} - 2K_{ij})} \tag{19}$$

对应的j的选择就是：

$$j \in arg \quad \min_t \{-\frac{b_{it}^2}{a_{it}} | t \in I_{low}(\alpha^k), -y_t \Delta f(\alpha)_t < -y_i \Delta f(\alpha)_i\} \tag{20}$$

证明:

首先定义如下变量：

$$\begin{cases} \hat{d}_i = y_i d_i \\ \hat{d}_j = y_j d_j \end{cases}$$

因为：$y_B^T d_B = 0$
所以有：$\hat{d}_i = -\hat{d}_j$
再定义：

$$\begin{cases} \Delta f(\alpha)_i = \sum_{j=1}^{L} \alpha_j y_i y_j K(x_i, x_j) - 1 \\ Q_{ij} = \frac{\Delta f(\alpha)_i}{\Delta \alpha_i} = y_i y_j K(x_i x_j) \end{cases}$$

然后将目标函数进行改写：

$$\begin{aligned} sub(B) &= \Delta f(\alpha^k)_B^T d_B + \frac{1}{2} d_B^T \Delta^2 f(\alpha^k)_{BB} d_B \\ &= \frac{1}{2}[d_i \quad d_j] \begin{bmatrix} \frac{\Delta f(\alpha)_i}{\Delta \alpha_i} & \frac{\Delta f(\alpha)_i}{\Delta \alpha_j} \\ \frac{\Delta f(\alpha)_j}{\Delta \alpha_i} & \frac{\Delta f(\alpha)_j}{\Delta \alpha_j} \end{bmatrix} \begin{bmatrix} d_i \\ d_j \end{bmatrix} + [\Delta f(\alpha^k)_i \quad \Delta f(\alpha^k)_j] \begin{bmatrix} d_i \\ d_j \end{bmatrix} \\ &= \frac{1}{2}[d_i \quad d_j] \begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ji} & Q_{jj} \end{bmatrix} \begin{bmatrix} d_i \\ d_j \end{bmatrix} + [\Delta f(\alpha^k)_i \quad \Delta f(\alpha^k)_j] \begin{bmatrix} d_i \\ d_j \end{bmatrix} \\ &= \frac{1}{2}[(d_i y_i)^2 K_{ii} - 2(d_i y_j)^2 K_{ij} + (d_j y_j)^2 K_{jj}] + (\Delta f(\alpha)_i d_i y_i^2 + \Delta f(\alpha)_j d_j y_j^2) \\ &= \frac{1}{2}(K_{ii} + K_{jj} - 2K_{ij}) \hat{d}_j^2 + (-y_i \Delta f(\alpha^k)_i + y_j \Delta f(\alpha^k)_j) \hat{d}_j \end{aligned} \tag{21}$$

因为我们假设$K_{ii} + K_{jj} - 2K_{ij} > 0$，且B是违反对，即$-y_i \Delta f(\alpha^k)_i > -y_j \Delta f(\alpha^k)_j$
我们定义:

$$\begin{gathered} a_{ij} = K_{ii} + K_{jj} - 2K_{ij} > 0 \\ b_{ij} = -y_i \Delta f(\alpha^k)_i + y_j \Delta f(\alpha^k)_j > 0 \end{gathered} \tag{22}$$

将（22）带入（21） 可以得到：

$$sub(B) = \frac{1}{2} a_{ij} \hat{d}_j^2 + b_{ij} \hat{d}_j \tag{23}$$

标准的二次函数$f(x) = ax^2 + bx + c$，极值点在$x = -\frac{b}{2a}$取得，极值为$y = -\frac{4ac-b^2}{4a}$，在该问题中，极值点在$\hat{d}_j = -\frac{b_{ij}}{a_{ij}}$取得，极值为$-\frac{b_{ij}^2}{2a_{ij}}$，即$-\frac{(-y_i \Delta f(\alpha^k)_i + y_i \Delta f(\alpha^k)_j)^2}{2(K_{ii}+K_{jj}-2K_{ij})}$。

因此对应的j的选择就是：

$$j \in arg \quad \min_t \{-\frac{b_{it}^2}{a_{it}} | t \in I_{low}(\alpha^k), -y_t \Delta f(\alpha)_t < -y_i \Delta f(\alpha)_i\} \tag{24}$$

α的更新

上面我们阐述了如何选择违反对，但是我们最终的目的是要找到一个最优的$\alpha_T = [\alpha_1, \alpha_2 \ldots, \alpha_l]$，现在我们只是找到$\alpha_1, \alpha_2$ ，所以我们需要在每轮迭代中更 新这两个变量，使得目标函数变小。

我们将问题(18)重新改写成下面形式

$$
\begin{aligned}
\min_{d_i, d_j} \quad & \frac{1}{2} \begin{bmatrix} d_i & d_j \end{bmatrix} \begin{bmatrix} Q_{ii} & Q_{ij} \\ Q_{ji} & Q_{jj} \end{bmatrix} \begin{bmatrix} d_i \\ d_j \end{bmatrix} + \begin{bmatrix} \nabla f(\alpha^k)_i & \nabla f(\alpha^k)_j \end{bmatrix} \begin{bmatrix} d_i \\ d_j \end{bmatrix} \\
s.t. \quad & y_i d_i + y_j d_j = 0 \\
& -\alpha_i^k \le d_i \le C_i - \alpha_i^k \\
& -\alpha_j^k \le d_j \le C_j - \alpha_j^k
\end{aligned}
$$

以及根据上面的推导我们有：

$$
\begin{aligned}
a_i^{k+1} = a_i^k + d_i = a_i^k + y_i \frac{b_{ij}}{a_{ij}} \\
a_j^{k+1} = a_j^k + d_j = a_j^k - y_j \frac{b_{ij}}{a_{ij}}
\end{aligned}
$$

将问题分成两种情况进行讨论（此处省略$y_i = y_j$的情况），我们对于$y_i \ne y_j$的情况进行讨论：

因为：

$$
\begin{aligned}
\alpha_i^{k+1} &= \alpha_i^k + y_i \frac{b_{ij}}{a_{ij}} \\
&= \alpha_i^k + y_i \frac{(-y_i \nabla f(\alpha)_i + y_j \nabla f(\alpha)_j)}{a_{ij}} \\
&= \alpha_i^k + \frac{-\nabla f(\alpha)_i - \nabla f(\alpha)_j}{a_{ij}}
\end{aligned}
$$

$$\begin{aligned}\alpha_j^{k+1} &= \alpha_j^k - y_j\frac{b_{ij}}{a_{ij}} \\ &= \alpha_j^k - y_j\frac{(-y_i\nabla f(\alpha)_i + y_j\nabla f(\alpha)_j)}{a_{ij}} \\ &= \alpha_j^k + \frac{-\nabla f(\alpha)_i - \nabla f(\alpha)_j}{a_{ij}}\end{aligned}$$

又因为$\sum_{i=1}^{L}\alpha_i y_i = 0$的约束，所以：

$$y_i\alpha_i^k + y_j\alpha_j^k = y_i\alpha_i^{k+1} + y_j\alpha_j^{k+1} = constant$$

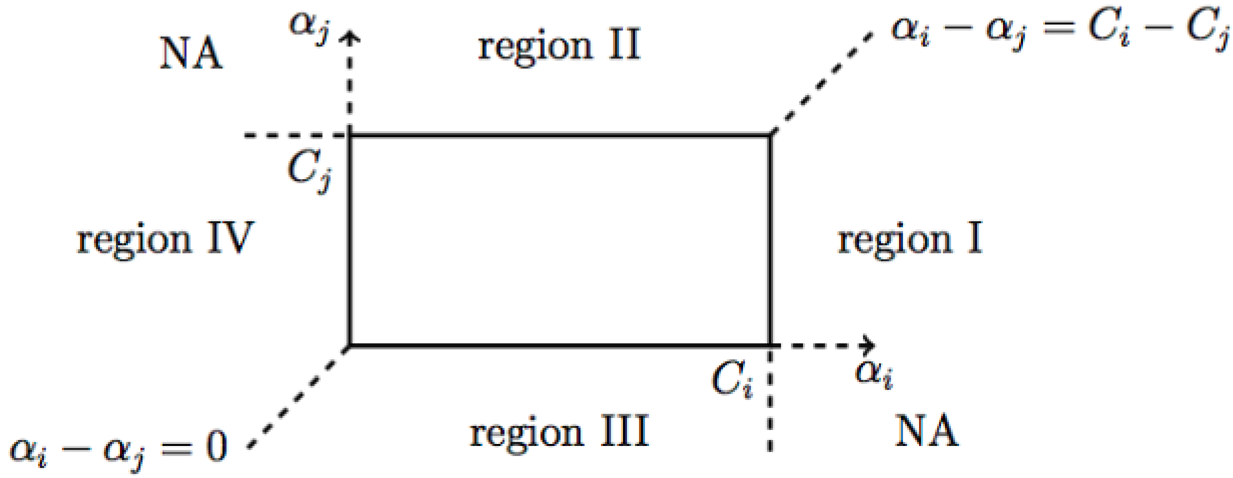
又因为$y_i \neq y_j$,我们将上式同乘以y_i:

$$\begin{aligned}&y_iy_i\alpha_i^k + y_iy_j\alpha_j^k = \alpha_i^k - \alpha_j^k = constant' \\ &y_iy_i\alpha_i^{k+1} + y_iy_j\alpha_j^{k+1} = \alpha_i^{k+1} - \alpha_j^{k+1} = constant'\end{aligned}$$

所以我们有：

$$\alpha_i^{k+1} - \alpha_i^k = \alpha_j^{k+1} - \alpha_j^k$$

如下图所示，$\alpha_i^{k+1}, \alpha_j^{k+1}$的取值也是有约束的，（$\alpha_i^{k+1}, \alpha_j^{k+1}$）必须在中间的矩形中，不合理的区域是在下图的4个区域中（region I，region II，region III，region IV）,不可能在你NA区域中，所以整个候选区域是在中间矩形+4个区域，合理的只有矩形，如果落在了均个区域中，我们要做一些clipping工作：



我们对这幅图进行解释，首先我们看看$\alpha_i - \alpha_j$的取值范围：

$$\begin{cases} 0 \le \alpha_i^k \le C_i \\ 0 \le \alpha_j^k \le C_j \end{cases} \Rightarrow \begin{cases} 0 \le \alpha_i^k \le C_i \\ -C_j \le -\alpha_j^k \le 0 \end{cases}$$

$$\Rightarrow -C_j \le \alpha_i^k - \alpha_j^k \le C_i$$

这说明$\alpha_i^k - \alpha_j^k$的取值是有范围的($-C_j \le \alpha_i - \alpha_j \le C_i$)，该结果与K无关，即不管K是在第几轮迭代$\alpha_i^k - \alpha_j^k$必须满足这个约束。

(1）左上角NA区域（不满足$-C_j \le \alpha_i - \alpha_j \le C_i$ 条件），右下角NA区域同理。

$$\alpha_i < 0 \ and \ \alpha_j > C_j \quad \Rightarrow \quad \alpha_i - \alpha_j < -C_j$$

（2）region I，在这个区域，我们有；

$$\begin{cases} \alpha_i - \alpha_j > C_i - C_j \\ a_i > C_i \end{cases}$$

但是由于$\alpha_i > C_i$,也就是α_i超出了矩形的区域，我们让a_i落回到矩形区域的边缘，即让$\alpha_i^{k+1} = C_i$,因为$\alpha_i^{k+1} - \alpha_i^k = \alpha_j^{k+1} - \alpha_j^k$,所以有：

$$\begin{cases} a_i^{k+1} = C_i \\ \alpha_j^{k+1} = C_i - \alpha_i^k + \alpha_j^k = C_i - (\alpha_i^k - \alpha_j^k) \end{cases}$$

(3) region II 区域:

$$\begin{cases} \alpha_i - \alpha_j < C_i - C_j \\ \alpha_j > C_j \end{cases}$$

我们令:

$$\begin{cases} \alpha_j^{k+1} = C_j \\ \alpha_i^{k+1} = C_j + \alpha_i^k - \alpha_j^k \end{cases}$$

(4) region III 区域:

$$\begin{cases} \alpha_i - \alpha_j > 0 \\ a_j < 0 \end{cases}$$

我们令:

$$\begin{cases} \alpha_j^{k+1} = 0 \\ \alpha_i^{k+1} = \alpha_i^k - \alpha_j^k \end{cases}$$

(5) region IV 区域:

$$\begin{cases} \alpha_i - \alpha_j < 0 \\ \alpha_i < 0 \end{cases}$$

我们令:

$$\begin{cases} \alpha_i^{k+1} = 0 \\ \alpha_j^{k+1} = -(\alpha_i^k - \alpha_j^k) \end{cases}$$

G($\Delta f(\alpha)$)的更新

G_{k+1}的更新相当于除了α_i, α_j，其他α都没有改变，然后将α_i, α_j的改变量更新到G_{k+1}.

$$
\begin{gathered}
\Delta \alpha_i = \alpha_i^{k+1} - \alpha_i^k \\
\Delta \alpha_j = \alpha_j^{k+1} - \alpha_j^k \\
G_{k+1} = G_k + Q_i \Delta \alpha_i + Q_j \Delta \alpha_j
\end{gathered}
$$

$\bar{G}$的更新

在训练过程中，当α_i到达了边界，即$\alpha_i = 0$或$\alpha_i = C$，α_i的值就不会改变了，它们的状态应该是inactive，而对于$0 < \alpha_i < C$的那些α_i，它们的状态是active的，这些$i \in A$，集合A的大小为active_size，在每一轮中都要检测active_size中的α_i 如果它们到达了边界，要把它们inactive.

我们更新G，其实只更新active的α_i梯度，即$\Delta f(\alpha)_i$，对于那些inactive的α_i，我 们为了加速计算，我们用$\bar{G}$来存储。

首先定义：

$$
\bar{G} = C \sum_{j:a_j=C} Q_{ij}, \quad i = 1, ..., L
$$

对于$i \notin A$，即那些inactive的α_i，我们需要计算他们的梯度：

$$
\begin{aligned}
\nabla f(\alpha)_i &= \sum_{j=1}^{L} Q_{ij}\alpha_j + e_i \\
&= \sum_{\alpha_j=0} Q_{ij} * 0 + \sum_{\alpha_j=C} Q_{ij} * C + \sum_{0<\alpha_j<C} Q_{ij}\alpha_j + e_i \\
&= \sum_{\alpha_j=C} Q_{ij} * C + \sum_{0<\alpha_j<C} Q_{ij}\alpha_j + e_i \\
&= \bar{G}_i + \sum_{0<\alpha_j<C} Q_{ij}\alpha_j + e_i
\end{aligned}
$$

提前计算好$\bar{G}$，这样就可以帮助我们加速计算梯度。