攀登传统机器学习的珠峰-SVM (下)

Technologies come and technologies go, but insight is forever.

---- Michael Nielsen

----------author：August助教（网易云课堂，机器学习微专业）----------

SMO (序列最小优化) 算法原理

友情提示：前方高能，公式推导特别多！！！

1. 回顾SVM优化目标

我们首先回顾下SVM的目标函数：

$$\underbrace{min}_{\alpha}\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{m}\alpha_i$$

$$s.t.\ \sum_{i=1}^{m}\alpha_i y_i=0$$

$$0\le\alpha_i\le C,\ \ i=1,2,\cdots,m$$

向量 α 是拉格朗日算子，每个样本 (x_i,y_i) 都对应一个向量分量 α_i 。

根据上节课的我们需要先求解 α，然后根据 α 再求解分类决策函数的参数 w 和 b。SMO算法的作用就是高效求解参数向量 α 的值 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_m^*)$ 。

KKT约束条件：

$$\alpha_i^* = 0 \Rightarrow y_i h(x_i) \geq 1$$
$$0 < \alpha_i^* < C \Rightarrow y_i h(x_i) = 1$$
$$\alpha_i^* = C \Rightarrow y_i h(x_i) \leq 1$$

其中 $h(x) = w^* \bullet \phi(x) + b = \sum_{j=1}^{m} \alpha_j^* y_j K(x, x_j) + b^*$ 为预测值。

2. SMO算法的基本思想

SMO 是微软研究院的 John C. Platt 在《Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines》一文中提出的，其基本思想是：把含有 m 个参数的原始问题分解成多个子问题分别求解，每个子问题只需要求解两个参数，这样SMO算法就将一个复杂的优化问题转化为一个比较简单的两变量优化问题。每次启发式选择两个变量进行优化，不断循环，直到达到函数最优值。

什么意思呢？下面我们具体讲解一下。

- 假设第一次迭代所选的两个变量是 α_{i1}, α_{j1} ，则$\alpha_1, \cdots, \alpha_{i1-1}, \alpha_{i1+1}, \cdots, \alpha_{j1-1}, \alpha_{j1+1}, \cdots, \alpha_m$视为常数。
- 通过极小目标函数求解 $\alpha_{i1} = \alpha_{i1}^{(1)}, \alpha_{j1} = \alpha_{j1}^{(1)}$ ，更新参数向量的 α 的值为 $\alpha_1, \cdots, \alpha_{i1}^{(1)}, \cdots, \alpha_{j1}^{(1)}, \cdots, \alpha_m$ 。
- 更新假设函数的参数 b 。
- 进入第二轮迭代，选择两个变量为 α_{i2}, α_{j2} ，则 $\alpha_1, \cdots, \alpha_{i2-1}, \alpha_{i2+1}, \cdots, \alpha_{j2-1}, \alpha_{j2+1}, \cdots, \alpha_m$ 视为常数。
- 通过极小目标函数求解 $\alpha_{i2} = \alpha_{i2}^{(2)}, \alpha_{j2} = \alpha_{j2}^{(2)}$ ，更新参数向量的 α 的值为 $\alpha_1, \cdots, \alpha_{i2}^{(2)}, \cdots, \alpha_{j2}^{(2)}, \cdots, \alpha_m$ 。
- 更新假设函数的参数 b 。
- 不断循环迭代，直到达到目标函数最优值。

Q1：之前有没有学过类似的迭代算法，他们的异同点是什么？

梯度下降算法，每次循环只更新一次参数；EM算法（坐标上升法），每个循环更新两次参数。而SMO算法属于坐标上升法，和EM算法更加亲密 (这句话好像也似曾相识，在哪儿见到过？回忆一下)。

后面我们会详细分析三者的区别与联系。

Q2：为什么要转化为两变量的优化问题？为什么不采用一个变量，三个变量，四个变量，...？

假如采用一个变量，比如说 α_1 ，即意味着其他 m-1 个参数视为常数，可以看成关于 α_1 的一元函数求解。但是由于约束条件 $\sum_{i=1}^{m} \alpha_i y_i = 0$, 当其他m-1 个参数为常数时，参数 $\alpha_1 = -y_1 \sum_{i=2}^{m} y_i \alpha_i$ 也被固定，意味着 m 个参数均视为常数，因此不能采用一个变量。

采用两个变量，比如 α_1 和 α_2 ，其他 m-2 个参数参数视为常数，那么根据 $\sum_{i=1}^{m} \alpha_i y_i = 0$ 可以用 α_1 来表示 α_2 ，即 $\alpha_2 = y_2 \sum_{i=3}^{m} \alpha_i y_i - \alpha_1 y_1 y_2$ ，将一个二变量 (α_1, α_2) 的优化转化为单变量 (α_1) 的优化问题，从而简化问题。

假如采用三个变量，四个变量，属于多变量优化问题，问题的复杂度还是很大。

以上我们了解了SMO算法的基本思想，SMO的第一步需要选择两个变量，那么怎样选择这两个变量呢？

3. SMO算法两个变量的选择

和梯度下降类似，SMO算法选择变量的原则是使得目标函数下降的最快。具体该怎样操作呢？

3.1 第一个变量的选择

第一个变量 α_1 的选择也称为外循环，这个变量需要选择距离真实值（最终收敛结果）最离谱的变量 α_1 ，也就是违反 KKT 约束条件最严重的样本点。

对于每个样本点，要满足的 KKT 条件我们上一节已经讲到了：

$$\alpha_i^* = 0 \Rightarrow y_i g(x_i) \geq 1$$
$$0 < \alpha_i^* < C \Rightarrow y_i g(x_i) = 1$$
$$\alpha_i^* = C \Rightarrow y_i g(x_i) \leq 1$$

Q3:对于样本种的任意一个点，都有可能违反上诉三个条件中的任意一个条件，我们应该首先以违反哪个条件的 α_i 为主呢？也就是违反三个条件中的哪个条件更离谱呢？

对 SVM 决策超平面结果影响最大的是支持向量，因此我们首先应该选择违反 $0 < \alpha_i^* < C \Rightarrow y_i g(x_i) = 1$ 这个条件的点。如果这些支持向量都满足KKT条件，再选择违反其他条件的点。

3.2 第二个变量的选择

第二个变量 α_2 的选择也称为内层循环，假设我们在外层循环已经找到了 α_1, α_2 的选择标准需要满足 α_2 变化最大，即最大 $|E_1 - E_2|$ 对应的 α_2 （其中 $E_i = \hat{y}_i - y_i$ ，即预测值和真实值之间的差值）。

Q4：为什么选择 $|E_1 - E_2|$ 最大化作为标准 ？怎样理解选择 $|E_1 - E_2|$ 足够大变化对应的变量 α_2

更新两个差别很大的变量，比起相似的变量，会带给目标函数更大的变化。下面会说明，α_2 的变化正好与 $|E_1 - E_2|$ 成正比。

由于 α_1 定了的时候，E_1 也确定了，所以要想 $|E_1 - E_2|$ 最大，只需要在 E_1 为正时，选择最小的 E_i 作为 E_2 ，在 E_1 为负时，选择最大的 E_i 作为 E_2 ，可以将所有的 E_i 保存下来加快迭代。

如果内存循环找到的点不能让目标函数有足够的下降， 可以采用以下步骤：

- 遍历支持向量点来做 α_2，直到目标函数有足够的下降；
- 如果所有的支持向量做 α_2 都不能让目标函数有足够的下降，可以在整个样本集上选择 α_2 ；
- 如果整个样本集依然不存在，则跳回外层循环重新选择 α_1。

4. SMO算法优化求解

假设我们通过上述方法得到了两个变量α_1 和 α_2那么

$$\sum_{i=1}^{m} \alpha_i y_i = 0 \Rightarrow \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{m} y_i \alpha_i = \varsigma$$

这样我们目标优化函数变为：

$$\underbrace{min}_{\alpha_1, \alpha_1} \frac{1}{2} K_{11} \alpha_1^2 + \frac{1}{2} K_{22} \alpha_2^2 + y_1 y_2 K_{12} \alpha_1 \alpha_2 - (\alpha_1 + \alpha_2) + y_1 \alpha_1 \sum_{i=3}^{m} y_i \alpha_i K_{i1} + y_2 \alpha_2 \sum_{i=3}^{m} y_i \alpha_i K_{i2} + \text{Const}$$

$$s.t. \;\; \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{m} y_i \alpha_i = \varsigma$$
$$0 \leq \alpha_i \leq C \;\; i = 1, 2$$

以上为有约束条件的的优化问题。传统的方法是采用拉格朗日乘子法进行求解转化为无约束的优化问题进行求解。这里我们采用一种更为自然的方法：

- 由于 α_1, α_2 满足 $\alpha_1 y_1 + \alpha_2 y_2 = \varsigma$，得到 $\alpha_2 = y_2 \varsigma - \alpha_1 y_1 y_2$ ，两变量的优化问题可以转化为单变量的优化问题。
- 对无约束条件下的单变量优化问题进行求导。求得该变量的值。
- 根据约束条件对得到的该变量的值进行修正。

假设我们上一轮迭代得到的解是 $\alpha_1^{old}, \alpha_2^{old}$，单变量求导 (未考虑约束条件) 得到的解是 $\alpha_1^{new,unc}, \alpha_2^{new,unc}$， 考虑约束条件之后的解为 $\alpha_1^{new}, \alpha_2^{new}$ 。下面我们详细讲解一下求解流程。

4.1 求解无约束的优化问题

求解无约束的优化问题非常简单，对目标函数求导即可。但在求导之前，最好能够对目标函数进行一下化简。

4.1.1 化简为单变量优化问题

令：

$$v_i = \sum_{j=3}^{m} y_j \alpha_j K(x_i, x_j) = h(x_i) - \sum_{j=1}^{2} y_j \alpha_j K(x_i, x_j) - b$$

其中 $h(x_i) = w \bullet \phi(x_i) + b = \sum_{j=1}^{m} \alpha_j y_j K(x_i, x_j) + b$ 。

这样我们的优化目标函数简化为：

$$W(\alpha_1, \alpha_2) = \frac{1}{2} K_{11} \alpha_1^2 + \frac{1}{2} K_{22} \alpha_2^2 + y_1 y_2 K_{12} \alpha_1 \alpha_2 - (\alpha_1 + \alpha_2) + y_1 \alpha_1 v_1 + y_2 \alpha_2 v_2$$

由于 $\alpha_1 y_1 + \alpha_2 y_2 = \varsigma$，并且 $y_i^2 = 1$，可以得到 α_1 用 α_2 表达的式子为：

$$\alpha_1 = y_1(\varsigma - \alpha_2 y_2)$$

将上式带入我们的目标优化函数，得到仅仅包含 α_2 的单变量函数：

$$W(\alpha_2) = \frac{1}{2} K_{11}(\varsigma - \alpha_2 y_2)^2 + \frac{1}{2} K_{22} \alpha_2^2 + y_2 K_{12}(\varsigma - \alpha_2 y_2)\alpha_2 - (\alpha_1 + \alpha_2) + (\varsigma - \alpha_2 y_2) v_1 + y_2 \alpha_2 v_2$$

4.1.2 求解单变量无约束问题

现在我们开始通过求偏导数来得到 $\alpha_2^{new,unc}$ 。

$$\frac{\partial W}{\partial \alpha_2} = K_{11}\alpha_2 + K_{22}\alpha_2 - 2K_{12}\alpha_2 - K_{11}\varsigma y_2 + K_{12}\varsigma y_2 + y_1 y_2 - 1 - v_1 y_2 + y_2 v_2 = 0$$

整理上式有：

$$(K_{11} + K_{22} - 2K_{12})\alpha_2 = y_2(y_2 - y_1 + \varsigma K_{11} - \varsigma K_{12} + v_1 - v_2)$$

$$= y_2(y_2 - y_1 + \varsigma K_{11} - \varsigma K_{12} + (h(x_1) - \sum_{j=1}^{2} y_j \alpha_j K_{1j} - b) - (h(x_2) - \sum_{j=1}^{2} y_j \alpha_j K_{2j} - b))$$

我们令:

$$E_i = h(x_i) - y_i = \sum_{j=1}^{m} \alpha_j^* y_j K_{ij} + b - y_i$$

其中 $h(x)=w\bullet\phi(x)+b=\sum_{j=1}^{m}\alpha_j y_j K(x,x_j)+b$ 为假设函数。

将 $\varsigma=\alpha_1^{old}y_1+\alpha_2^{old}y_2$ 带入上式，我们有：

$$\begin{aligned}(K_{11}+K_{22}-2K_{12})\alpha_2^{new,unc}&=y_2((K_{11}+K_{22}-2K_{12})\alpha_2^{old}y_2+y_2-y_1+h(x_1)-h(x_2))\\&=(K_{11}+K_{22}-2K_{12})\alpha_2^{old}+y_2(E_1-E_2)\end{aligned}$$

我们终于得到了 $\alpha_2^{new,unc}$ 的表达式：

$$\alpha_2^{new,unc}=\alpha_2^{old}+\frac{y_2(E_1-E_2)}{K_{11}+K_{22}-2K_{12}}$$

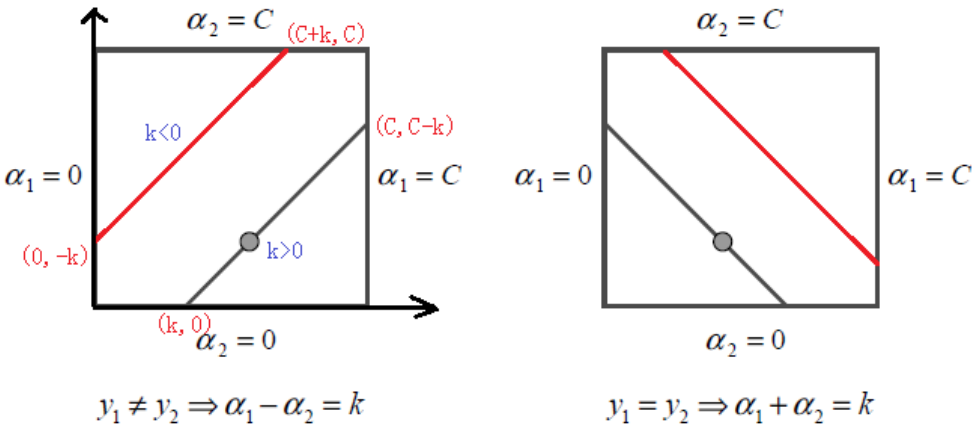
4.2 约束条件修正变量结果

下面我们考虑约束条件：

$\alpha_1y_1+\alpha_2y_2=\varsigma$

$0\le\alpha_i\le C\ \ i=1,2$

Q5：请在坐标轴中画出上述约束条件的几何表示



$0\le\alpha_i\le C$ 将两变量限制在 $[0,C]\times[0,C]$ 的矩形中，$\alpha_1y_1+\alpha_2y_2=\varsigma$ 将两个变量限制在矩形中平行于对角线的线段上。左图为 y_1,y_2 异号的情况，右图为 y_1,y_2 同号的情况。

由于 α_2^{new} 必须要在方框内且在直线上取得。假设 L 和 H 分别是上图中 α_2^{new} 所在的线段的边界。那么：

$$L\le\alpha_2^{new}\le H$$

对于上面左图中的情况 $y_1\neq y_2$，则

$$L=max(0,\alpha_2^{old}-\alpha_1^{old})\quad H=min(C,C+\alpha_2^{old}-\alpha_1^{old})$$

对于上面右图中的情况 $y_1=y_2$，有：

$$L=max(0,\alpha_2^{old}+\alpha_1^{old}-C)\quad H=min(C,\alpha_2^{old}+\alpha_1^{old})$$

假如我们通过求导得到的 $\alpha_2^{new,unc}$，则最终的 α_2^{new} 应该为：

$$\alpha_2^{new} = \begin{cases} H & \alpha_2^{new,unc} > H \\ \alpha_2^{new,unc} & L \le \alpha_2^{new,unc} \le H \\ L & \alpha_2^{new,unc} < L \end{cases}$$

利用上面讲到的 $\alpha_2^{new,unc}$ 和 α_2^{new} 的关系式，我们就可以得到我们新的 α_2^{new} 了。

由于其他 $m-2$ 个变量固定，因此：

$$\alpha_1^{old} y_1 + \alpha_2^{old} y_2 = \alpha_1^{new} y_1 + \alpha_2^{new} y_2$$

求得：

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 (\alpha_2^{old} - \alpha_2^{new})$$

4.3 计算 b

在每次完成两个变量的优化之后，需要重新计算阈值 b。当 $0 < \alpha_1^{new} < C$ 时，我们有

$$y_1 - \sum_{i=1}^{m} \alpha_i y_i K_{i1} - b_1 = 0$$

于是新的 b_1^{new} 为：

$$b_1^{new} = y_1 - \sum_{i=3}^{m} \alpha_i y_i K_{i1} - \alpha_1^{new} y_1 K_{11} - \alpha_2^{new} y_2 K_{21}$$

计算出 E_1 为：

$$E_1 = h(x_1) - y_1 = \sum_{i=3}^{m} \alpha_i y_i K_{i1} + \alpha_1^{old} y_1 K_{11} + \alpha_2^{old} y_2 K_{21} + b^{old} - y_1$$

将 b_1^{new} 用 E_1 表示为：

$$b_1^{new} = -E_1 - y_1 K_{11} (\alpha_1^{new} - \alpha_1^{old}) - y_2 K_{21} (\alpha_2^{new} - \alpha_2^{old}) + b^{old}$$

同样的，如果 $0 < \alpha_2^{new} < C$, 那么有：

$$b_2^{new} = -E_2 - y_1 K_{12} (\alpha_1^{new} - \alpha_1^{old}) - y_2 K_{22} (\alpha_2^{new} - \alpha_2^{old}) + b^{old}$$

最终的 b^{new} 为：

$$b^{new} = \frac{b_1^{new} + b_2^{new}}{2}$$

得到了 b^{new} 为我们需要更新 E_i:

$$E_i^{new} = \sum_{S} y_j \alpha_j K(x_i, x_j) + b^{new} - y_i$$

其中，S 是所有支持向量 x_j 的集合，利用 b^{new} 和 E_i^{new} 选择新的变量进行循环迭代。

5. SMO算法总结

输入是 m 个线性可分的样本 $(x_1,y_1),(x_2,y_2),\ldots,(x_m,y_m)$,

其中 x 为 n 维特征向量。y 为二元分类结果 1 或 -1, 精度 ϵ。

输出是近似解 α。

算法过程如下：

1) 取初始值 $\alpha^0=0,k=0$。

2) 选择 α_1^k,α_2^k，求出新的 $\alpha_2^{new,unc}$ ：

$$\alpha_2^{new,unc}=\alpha_2^k+\frac{y_2(E_1-E_2)}{K_{11}+K_{22}-2K_{12}}$$

3) 按照下式求出 α_2^{k+1} ：

$$\alpha_2^{k+1}=\begin{cases}H & L\le\alpha_2^{new,unc}>H\\ \alpha_2^{new,unc} & L\le\alpha_2^{new,unc}\le H\\ L & \alpha_2^{new,unc}<L\end{cases}$$

4) 利用 α_2^{k+1} 和 α_1^{k+1} 的关系求出 α_1^{k+1}。

5) 计算 b^{k+1} 和 E_i。

6) 在精度 ϵ 范围内是否满足如下的终止条件：

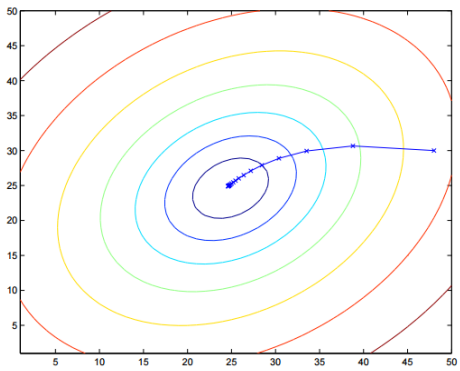
$$\sum_{i=1}^{m}\alpha_i y_i=0$$
$$0\le\alpha_i\le C,i=1,2...m$$
$$\alpha_i^*=0\Rightarrow y_i h(x_i)\ge 1$$
$$0<\alpha_i^*<C\Rightarrow y_i h(x_i)=1$$
$$\alpha_i^*=C\Rightarrow y_i h(x_i)\le 1$$

7) 如果满足则结束，返回 α^{k+1},否则转到步骤2。

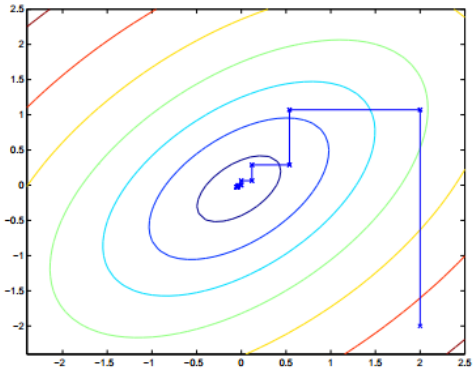
6. 迭代优化算法小结

目前我们已经学到两种基本的迭代优化算法，三种应用。下面我用更加直观的方式一一说明。

1. 梯度下降 (或上升)法：沿函数值增幅最大 (即梯度方向) 的反方向走，就能使函数值减小 (或增大) 幅度越大。主要应用在逻辑回归算法，深度学习等算法中。如下图所示：



2. 坐标上升 (或下降法) 法：每次只选择一个维度，将原始的问题变成一个一元函数，然后针对这个一元函数求极值，如此反复轮换不同的维度进行迭代。主要应用包括EM算法和SMO算法。
    - 对于EM算法，显然是一个二变量迭代的坐标上升问题，其特点是两个维度轮换迭代。难点是怎样连接迭代步骤中的E步和M步（ Jensen 不等式 ）。如下图所示：



    - 对于 SMO 算法则是一个 m-1 (m表示样本量) 维度的坐标下降问题，难点在于该选择哪个坐标作为下一个循环的迭代，同时还需要对求导得到的变量值进行约束条件限制。（暂时无法可视化高维图像，故此处没有示意图 ）

Q6：SMO算法中 α 的维度是 m，为什么坐标下降算法中坐标的维度变成了m-1？

由于约束条件 $\sum_{i=1}^{m}\alpha_i y_i = 0$ 的存在，我们可以判断，一直任意 m-1 个变量的值，那么另一个变量的值也就确定了，即 $\sum_{i=1}^{m}\alpha_i y_i = 0$

Q7：为什么EM算法和SMO算法不能直接用梯度下降 (或上升) 进行求解？

回忆一下EM算法中的似然函数：

$$\theta, z = arg\max_{\theta,z} L(\theta, z) = arg\max_{\theta,z} \sum_{i=1}^{m} log \sum_{z^{(i)}} P(x^{(i)}, z^{(i)}|\theta)$$

由于$logP(x^{(i)}|\theta)$ 是$P(x^{(i)}, z^{(i)}|\theta)$ 边缘概率，转化为 $logP(x^{(i)}|\theta)$ 求导后形式会非常复杂（可以想象下 $log(f_1(x) + f_2(x) + \ldots)$ 复合函数的求导，所以很难直接通过梯度下降法求解得到 z 和 θ。

对于SMO算法，其优化函数为：

$$\underbrace{min}_{\alpha}\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(x_i, x_j) - \sum_{i=1}^{m}\alpha_i$$

$$s.t. \sum_{i=1}^{m}\alpha_i y_i = 0$$

$$0 \le \alpha_i \le C, \quad i = 1, 2, \cdots, m$$

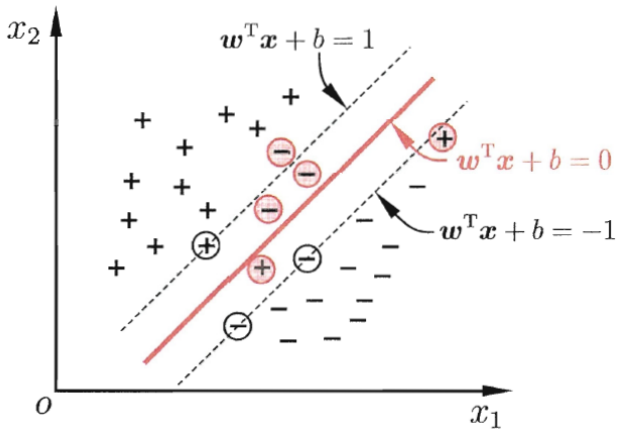
对于任意一个 α_i 求导得到 $\frac{1}{2}\sum_{j=1}^{m}\alpha_i\alpha_j y_i y_j K(x_i, x_j) - 1$，显然它的系数会包含其他所有的变量，每一个方程中还是包含 m 个变量，是无法用梯度下降法进行求解的。

总结一下：梯度下降 (或上升) 法是优化算法最简单的，也是最实用的。当问题复杂，梯度下降算法无法求出最优解时，我们可以考虑采用坐标下降 (或上升) 法。

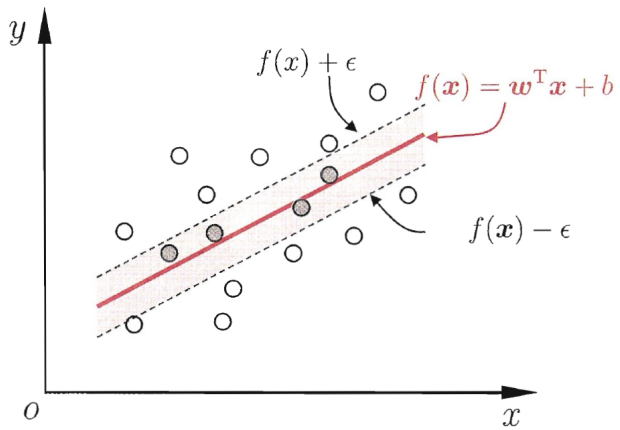
SVM 回归模型（又称为SVR）

1. SVR模型的损失函数度量

回顾下我们SVM分类模型 (SVC)，传统的分类模型通常直接基于模型输出 $h(x)$ 和真实类别 y 之间的差别来定义损失，每个样本点都对应一个损失，当 $h(x)$ 与 y 的符号相同，距离决策边界越近，损失越大，对于误分类的点，距离决策边界越远，损失越大。而 SVM 分类仅仅关心决策边界附近易混淆或误分类的样本点，即当且仅当样本点与决策边界的距离小于 $\epsilon = 1$ 时，才计算损失，相当于以 $f(x)$ 为中心，构建了一个 宽度为 2 的间隔带，间隔带外面正确分类的点都是没有损失的，误分的点和间隔带里面的点是有损失的。



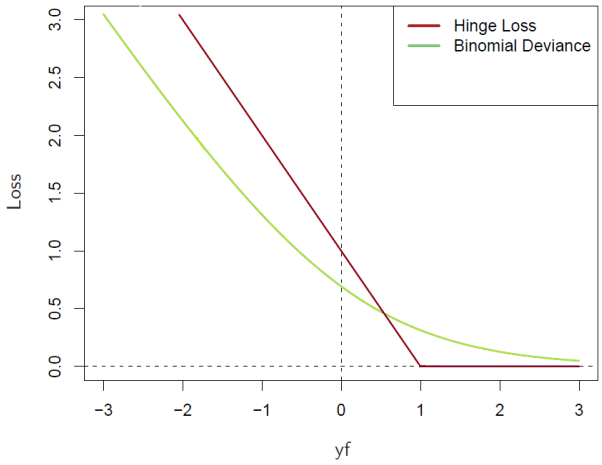
再来看SVR, 传统的回归模型通常直接基于模型输出 $h(x)$ 和真实输出 y 之间的差别来定义损失，当且仅当 $h(x)$ 与 y 完全相同时，损失才为 0。而 SVR 仅仅关系距离拟合模型距离较远的样本点，即当且仅当 $f(x)$ 与 y 直接的差别大于 ϵ (不敏感损失) 时，才计算损失，这就相当于以 $f(x)$ 为中心，构建了一个 宽度为 2ϵ 的间隔带，间隔带里面的点都是没有损失的，外面的点的是有损失的。



由上节课我们知道 SVC 的损失函数为：

$$min\ \frac{1}{m}\sum_{i=1}^{m}max\{0,1-y_i\hat{y}_i\}+\frac{\lambda}{2}||w||_2^2$$

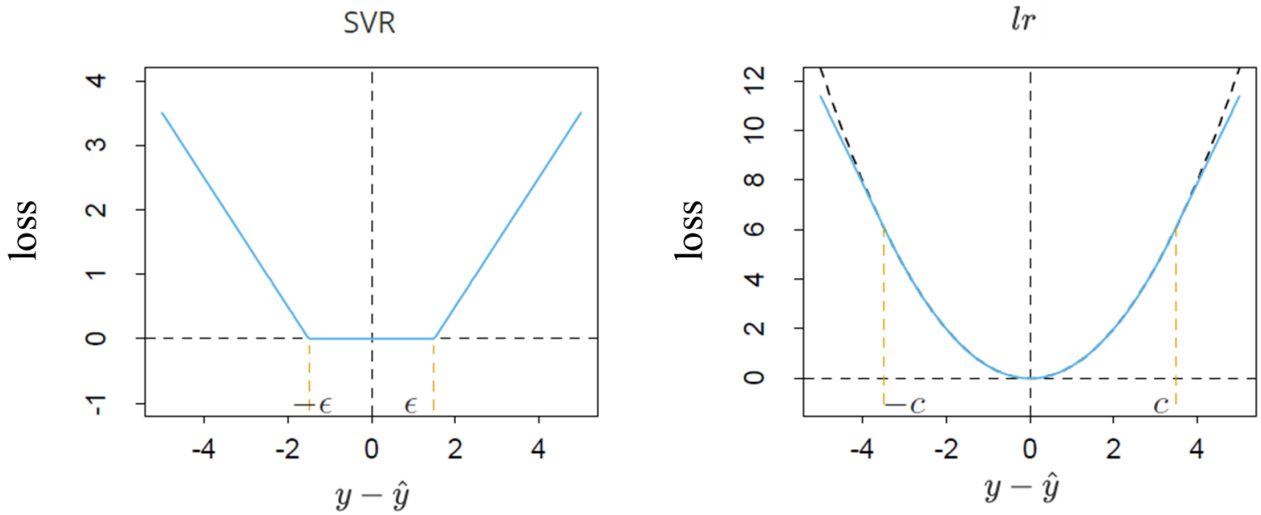
下图对比逻辑回归损失函数和SVC损失函数之间的区别：



我们可以得到 SVR 的损失函数为：

$$min \ \frac{1}{m}\sum_{i=1}^{m} max\{0, |y_i\hat{y}_i| - \epsilon\} + \frac{\lambda}{2}||w||_2^2$$

下图对比 SVR 损失函数和lr均方差损失函数之间的区别：



Q8：对比 SVC 和 SVR 的代价函数，有什么区别？为什么会有这种区别？

SVC 和 SVR 之间的区别主要体现在 $1-y_i\hat{y}_i$ 和 $|y_i\hat{y}_i|-\epsilon$，具体说来有三个区别：

- SVR 用的是 ϵ 而 SVC 用的是 1。即 SVC 采用相对距离（还记得怎么推导的吧？），而 SVR 采用的是绝对距离。这是因为对于最终结果来说，SVC 只需要考虑到决策边界的相对距离，距离的实际大小并没有意义。而 SVR 最后要得到实际值的大小，不能随便缩放。
- SVR 是 $|y_i\hat{y}_i|$，而 SVC 是 $y_i\hat{y}_i$，SVR 多了个绝对值，这是因为 SVR 只需要考虑样本点到拟合模型的绝对距离大小，而不需要考虑样本点在拟合模型的哪一侧。SVC 则需要考虑误分类点的情况，对于误分类的点 $y_i\hat{y}_i < 0$, 对于正确分类的点 $y_i\hat{y}_i > 0$，因此可以理解为 SVC 考虑的是有向距离。
- SVC 中常数为被除数，SVR 中常数是除数，类似于相反数的关系。这是由于 SVC 中间隔带内的样本点和误分类的点才有损失，即对于 $y_i\hat{y}_i < 1$ 的样本点损失为 $1-y_i\hat{y}_i = max\{0, 1-y_i\hat{y}_i\}$。而SVR中间隔带外的样本点才有损失，即对于$y_i\hat{y}_i > 1$ 的样本点损失为 $|y_i\hat{y}_i| - \epsilon = max\{0, |y_i\hat{y}_i| - \epsilon\}$。

2. SVR目标函数的原始形式

上一节我们已经得到了我们的损失函数的度量，现在可以可以定义我们的目标函数如下：

$$min \ \frac{1}{2}||w||_2^2 \ \ s.t \ \ |y_i - w \bullet \phi(x_i) - b| \le \epsilon \ \ (i = 1, 2, ...m)$$

SVR 中加入松弛变量 ξ_i 之后变为：

$$\begin{aligned} & min \ \frac{1}{2}||w||_2^2 + C\sum_{i=1}^{m}(\xi_i^\vee + \xi_i^\wedge) \\ s.t. \quad & -\epsilon - \xi_i^\vee \le y_i - w \bullet \phi(x_i) - b \le \epsilon + \xi_i^\wedge \\ & \xi_i^\vee \ge 0, \ \ \xi_i^\wedge \ge 0 \ (i = 1, 2, ..., m) \end{aligned}$$

用拉格朗日函数将目标优化函数变成无约束的形式：

$$L(w, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge) = \frac{1}{2}||w||_2^2 + C\sum_{i=1}^{m}(\xi_i^\vee + \xi_i^\wedge) + \sum_{i=1}^{m}\alpha^\vee(-\epsilon - \xi_i^\vee - y_i + w \bullet \phi(x_i) + b)$$

$$+ \sum_{i=1}^{m}\alpha^\wedge(y_i - w \bullet \phi(x_i) - b - \epsilon - \xi_i^\wedge) - \sum_{i=1}^{m}\mu^\vee\xi_i^\vee - \sum_{i=1}^{m}\mu^\wedge\xi_i^\wedge$$

其中 $\mu^\vee \ge 0, \mu^\wedge \ge 0, \alpha_i^\vee \ge 0, \alpha_i^\wedge \ge 0$，均为拉格朗日系数。

从上面的公式我们发现，由于绝对值的存在， SVR 中的变量比 SVC 中多了一倍。

3. SVR目标函数的对偶形式

上一节我们讲到了SVM回归模型的目标函数的原始形式,我们的目标是：

$$\underbrace{min}_{w,b,\xi_i^\vee,\xi_i^\wedge} \ \underbrace{max}_{\mu^\vee \ge 0, \mu^\wedge \ge 0, \alpha_i^\vee \ge 0, \alpha_i^\wedge \ge 0} \ L(w, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge)$$

和SVM分类模型一样，这个优化目标也满足KKT条件，也就是说，我们可以通过拉格朗日对偶将我们的优化问题转化为等价的对偶问题来求解如下：

$$\underbrace{max}_{\mu^\vee \ge 0, \mu^\wedge \ge 0, \alpha_i^\vee \ge 0, \alpha_i^\wedge \ge 0} \ \underbrace{min}_{w,b,\xi_i^\vee,\xi_i^\wedge} \ L(w, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge)$$

我们可以先求优化函数对于 $w, b, \xi_i^\vee, \xi_i^\wedge$ 的极小值, 接着再求拉格朗日乘子 $\alpha^\vee, \alpha^\wedge, \mu^\vee, \mu^\wedge$ 的极大值。

首先我们来求优化函数对于 $w, b, \xi_i^\vee, \xi_i^\wedge$ 的极小值，这个可以通过求偏导数求得：

$$\frac{\partial L}{\partial w} = 0 \ \Rightarrow w = \sum_{i=1}^{m}(\alpha_i^\wedge - \alpha_i^\vee)\phi(x_i)$$

$$\frac{\partial L}{\partial b} = 0 \ \Rightarrow \sum_{i=1}^{m}(\alpha_i^\wedge - \alpha_i^\vee) = 0$$

$$\frac{\partial L}{\partial \xi_i^\vee} = 0 \ \Rightarrow C - \alpha^\vee - \mu^\vee = 0$$

$$\frac{\partial L}{\partial \xi_i^\wedge} = 0 \ \Rightarrow C - \alpha^\wedge - \mu^\wedge = 0$$

好了，我们可以把上面4个式子带入 $L(w, b, \alpha^\vee, \alpha^\wedge, \xi_i^\vee, \xi_i^\wedge, \mu^\vee, \mu^\wedge)$ 去消去 $w, b, \xi_i^\vee, \xi_i^\wedge$ 了。

最终得到的对偶形式为：

$$\underbrace{min}_{\alpha^{\vee},\alpha^{\wedge}} \frac{1}{2}\sum_{i=1,j=1}^{m}(\alpha_i^{\wedge}-\alpha_i^{\vee})(\alpha_j^{\wedge}-\alpha_j^{\vee})K_{ij}+\sum_{i=1}^{m}(\epsilon-y_i)\alpha_i^{\wedge}+(\epsilon+y_i)\alpha_i^{\vee}$$

$$s.t.\ \sum_{i=1}^{m}(\alpha_i^{\wedge}-\alpha_i^{\vee})=0$$

$$0<\alpha_i^{\vee}<C\ (i=1,2,\dots m)$$

$$0<\alpha_i^{\wedge}<C\ (i=1,2,\dots m)$$

用SMO算法来求出对应的 $\alpha^{\vee},\alpha^{\wedge}$，进而求出我们的回归模型系数 w,b。

SVM 算法小结

SVM算法的主要优点有：

- 仅仅使用一部分支持向量来做超平面的决策，无需依赖全部数据，有较好的的鲁棒性。
- 有大量的核函数可以使用，从而可以很灵活的来解决各种非线性的分类回归问题。
- 分类准确率高，泛化能力强。

SVM算法的主要缺点有：

- 则SVM一般用于样本数大于特征维度的场景。
- 由于SVM计算量过大，不太适合在样本量非常大时使用。
- 非线性问题的核函数的选择没有通用标准，难以选择一个合适的核函数。

sklearn SVM库使用小结

skearn中 SVM 的算法库分为两类，一类是分类的算法库，包括 SVC， NuSVC，和 LinearSVC 。另一类是回归算法库，包括SVR， NuSVR，和 LinearSVR 。

对于分类模型，SVC 和 NuSVC 仅仅在于对损失的度量方式不同，而 LinearSVC 是线性分类，对线性不可分的数据不能使用。

同理，对于回归模型， SVR 和 NuSVR 仅仅在于对损失的度量方式不同。LinearSVR 是线性回归。

除非知道数据是线性的，我们使用 LinearSVC 或者 LinearSVR。一般使用 SVC 或 SVR。

如果我们对训练集训练的错误率或支持向量的百分比有要求的时候，可以选择 NuSVC 或 NuSVR 。它们有一个参数来控制这个百分比。

SVC 参数总结如下：

参数	LinearSVC	SVC	NuSVC
惩罚系数C	即惩罚系数C，默认为1，通过交叉验证来选择。如果噪音点较多时，C小一些。		NuSVC没有这个参数
nu	LinearSVC 和SVC没有这个参数。		nu代表训练集训练的错误率的上限，取值范围为(0,1],默认是0.5.它和C类似控制惩罚的力度。
核函数 kernel	LinearSVC没有这个参数， LinearSVC只能使用线性核函数	核函数有四种选择'linear', 'poly', 'rbf', 'sigmoid'。对应的核函数参数在后面有单独的参数需要调。默认是高斯核'rbf'。	
正则化参数penalty	仅对线性拟合有意义，可以选择'l1'即L1正则化 或者 'l2'即L2正则化。默认是L2。	SVC和NuSVC没有这个参数	
是否用对偶形式优化dual	布尔变量，控制是否使用对偶形式，默认是True。如果样本量比特征数多，设置为False，即采用原始形式	SVC和NuSVC没有这个参数	
核函数参数degree	LinearSVC没有这个参数，	如果kernel参数使用了 'poly'，那么我们就需要对这个参数进行调参一般需要通过交叉验证选择 d	
核函数参数gamma	LinearSVC没有这个参数	如果在kernel参数使用了 'poly'，'rbf'或'sigmoid'是，需要对这个参数进行调参。γ默认为'auto',即1/特征维度	
核函数参数 coef 0	LinearSVC没有这个参数	如果我们在kernel参数使用了 'poly'或'sigmoid'时， coef 0 即为公式中的 r, 默认为0	
样本权重 class_weight	对于非均衡样本指定样本各类别的的权重。可以自己指定各个样本的权重，或者用"balanced"，则算法会自己计算权重，样本量少的类别所对应的样本权重会高。样本均衡选择默认的"None"。		
分类决策 decision_function_shape	LinearSVC没有这个参数，使用multi_class参数替代。	可以选择'ovr'(one ve rest)或者'ovo'(one-vs-one ovr相对简单，但分类效果相对略差。而ovo分类相对精确，但是分类速度没有ovr快。建议使用OvO。	
分类决策 multi_class	可选 'ovr' 或 'crammer_singer' 'crammer_singer'是一种改良版的'ovr'，但是没有比'ovr'好，一般不建议使用。	SVC和nuSVC没有这个参数，使用decision_function_shape参数替代。	
缓存大小 cache_size	LinearSVC计算量不大，不需要这个参数	在大样本的时如果机器内存大，推荐用500MB甚至1000MB。默认是200，即200MB.	

SVR 参数总结如下：

参数	LinearSVR	SVR	nuSVR
惩罚系数C	即惩罚系数C，默认为1，一般需要通过交叉验证来选择一个合适的 C。一般来说，如果噪音点较多时，C需要小一些。大家可能注意到在分类模型里面，nuSVC 使用了nu这个等价的参数控制错误率，没有使用 C，在回归模型里面，回归错误率是惩罚系数C和距离误差 ϵ 共同作用的结果。		
nu	LinearSVR 和SVR没有这个参数，用ϵ控制错误率		nu代表训练集训练的错误率的上限，取值范围为(0,1],默认是0.5.通过选择不同的错误率可以得到不同的距离误差ϵ。
距离误差epsilon	ϵ定义了支持向量间隔		nuSVR没有这个参数，用nu控制错误率
是否用对偶形式优化dual	和SVC类似	SVR和NuSVR没有这个参数	
正则化参数penalty	和SVC类似	SVR和NuSVR没有这个参数	
核函数 kernel	LinearSVR没有这个参数，LinearSVR限制了只能使用线性核函数	和SVC, nuSVC类似	
核函数参数 degree, gamma 和 coef0	LinearSVR没有这些参数，LinearSVR限制了只能使用线性核函数	和SVC, nuSVC类似	
损失函数度量loss	可以选择为‘epsilon_insensitive’ 和‘squared_epsilon_insensitive’，‘epsilon_insensitive’ 是SVM回归的损失度量标准形式。一般用默认的‘epsilon_insensitive’就足够了。	SVR和NuSVR没有这个参数	
缓存大小\cache_size	LinearSVC计算量不大，不需要这个参数	在大样本的时候，如果机器内存大，和SVC，nuSVC一样，推荐用500MB甚至1000MB。默认是200，即200MB.	

这里对其他的调参要点做以下几点说明：

- 训练模型之前对数据进行归一化，。
- 在特征数非常多的情况下，或者样本数远小于特征数的时候，只需使用线性核即可。
- 在选择核函数时，如果线性拟合不好，一般推荐使用默认的高斯核 'rbf'。这时我们主要需要对惩罚系数C和核函数参数 γ 进行调参$。
- 理论上高斯核不会比线性核差，高斯核要花费更多的时间，所以能用线性核解决问题尽量使用线性核。

参考文献：

scikit-learn 支持向量机算法库使用小结 http://www.cnblogs.com/pinard/p/6117515.html

T. Hastie/ R. Tibshirani / J. H. Friedman 《The Elements of Statistical Learning》

李航 《统计学习方法》

周志华 《机器学习》