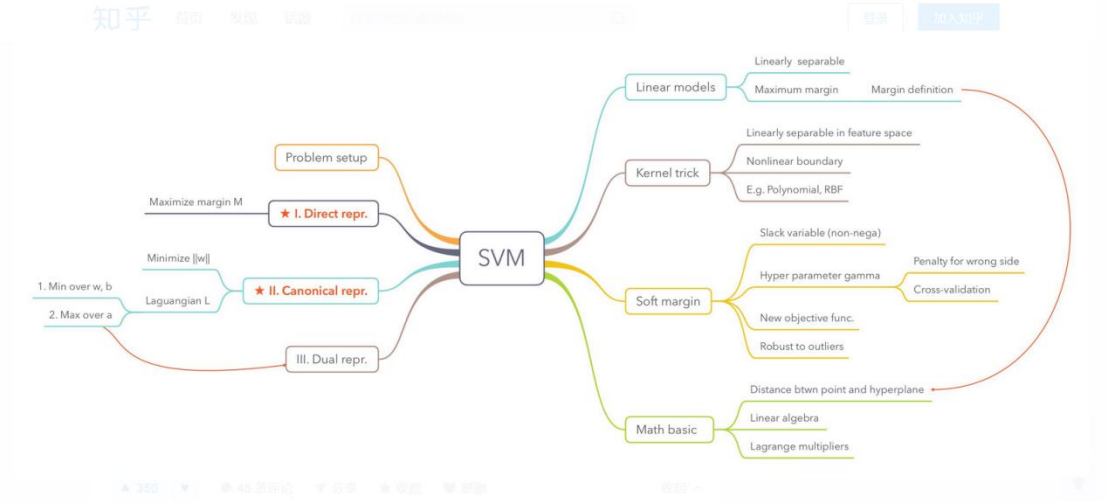
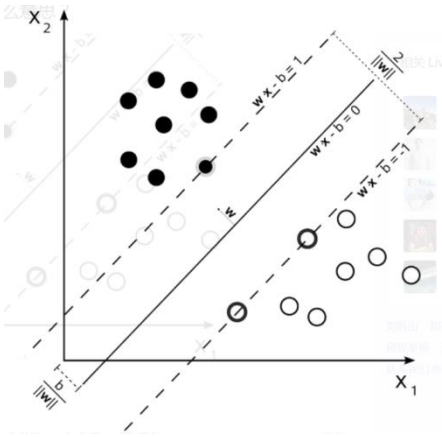
支持向量机

介绍内容 支持向量机、核函数、序列最小最优化算法（SMO、多核概念



什么是 SVM？

一个普通的 SVM 就是一条直线罢了，用来完美划分 linearly separable 的两类。但这又不是一条普通的直线，这是无数条可以分类的直线当中最完美的，因为它恰好在两个类的中间，距离两个类的点都一样远。



一、 线性可分支持向量机

给定一个特征空间的训练数据集 $T=\{(x_1,y_1),(x_2,y_2)...(x_N,y_N)\}$ 当 $y_i=+1$ 时称 x_i 正例，反之，同理。假设训练数据集是线性可分的。

学习目标是在特征空间找到一个分离超平面，能将实例分到不同的类。分离超平面方程 $w^*\bullet x+b=0$，相应的分类决策函数 $f(x)=sign(w^*\bullet x+b^*)$

1.2 函数间隔和几何间隔

定义超平面 (w,b) 关于样本点 (x_i,y_i) 的函数间隔为 $\hat{\gamma}_i=y_i(w\bullet x_i+b)$，所有样本间隔最小

值 $\hat{\gamma} = \min_{i=1...N} \hat{\gamma}_i$。函数间隔可以表示分类预测的正确性及确信度。但是选择分离超平面是，只有函数间隔还不够。因为 w 和 b 成比例的改变超平面没变，但是函数间隔变化，因此对超平面的法向量 w 坐一些约束规范化 $\|w\|=1$，使得间隔确定，称为几何间隔。

$\gamma_i = y_i(\frac{w}{\|w\|} \bullet x_i + \frac{b}{\|w\|})$，因此函数间隔和几何间隔关系 $\gamma = \frac{\hat{\gamma}}{\|w\|}$，$\|w\|=1$ 相等。

1.2 间隔最大化

几何间隔最大的超平面是唯一的（硬间隔最大化）：充分大的确信度对训练数据进行分类。

最大间隔分离超平面

$$\max_{w,b} \frac{\hat{\gamma}}{\|w\|}$$

$$s.t \quad y_i(\frac{w}{\|w\|} \bullet x_i + \frac{b}{\|w\|}) \geq \gamma_i, i = 1...N \qquad 1$$

函数间隔 $\hat{\gamma}$ 取值并不影响最优化问题的解。假设等比例放缩 λw 和 λb，函数间隔成为 $\lambda\hat{\gamma}$。就可以取值 $\hat{\gamma}$ =1，代入上面最优化问题最大化 $\frac{1}{\|w\|}$ 和最小化 $\frac{1}{2}\|w\|^2$ 是等价的

$$\min_{w,b} \frac{1}{2}\|w\|^2$$

$$s.t \quad y_i(\frac{w}{\|w\|} \bullet x_i + \frac{b}{\|w\|}) \geq \gamma_i, i = 1...N \qquad 2$$

这是一个凸二次规划问题(convex quadratic programming)

最大间隔分离超平面的存在 唯一性。在决定分离超平面时只有支持向量其作用，而其他实例点并不起作用，如果移动支持向量将改变所求的解；但是如果早间隔意外移动其他实例点甚至去掉这些点，则解是不会改变的。由于支持向量在确定分离超平面中其决定性作用，所以称为支持向量。支持向量个数很少。

1.3 学习的对偶算法

求解线性可分支持向量机最优化问题上述方程 2.应用拉格朗日对偶问题（dual problem）得到原始问题（primal）的最优解。优点 对偶问题容易求解，自然引入核函数，进而推广到非线性分类问题。

首先构建拉格朗日函数，引进拉格朗日乘子 $\alpha_i \geq 0, i = 1..N$

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (w \bullet x_i + b) + \sum_{i=1}^{N} \alpha_i \qquad 3$$

其中 $\alpha = (\alpha_1, ...\alpha_N)^T$ 拉格朗日乘子向量。根据拉格朗日对偶性，原始问题的对偶问题是极大极小问题：$\max_{\alpha} \min_{w,b} L(w,b,\alpha)$，为了得到对偶问题的解需要先求 $L(w,b,\alpha)$ 对 w，b 极小，再对 α 的极大

（1） 求 $\min_{w,b} L(w,b,\alpha)$

将拉格朗日函数 $L(w,b,\alpha)$ 分别对 w,b 求偏导数并令其=0.

$$\nabla_w L(w,b,\alpha) = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0$$

$$\nabla_b L(w,b,\alpha) = \sum_{i=1}^{N} \alpha_i y_i = 0$$

得

$$w = \sum_{i=1}^{N} \alpha_i y_i x_i$$

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

带入拉格朗日函数

$$\min_{w,b} L(w,b,\alpha) = -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \bullet x_j) + \sum_{i=1}^{N} \alpha_i$$

（2） 求 $\min_{w,b} L(w,b,\alpha)$ 对 α 的极大，即对偶问题

$$\max_{\alpha} -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \bullet x_j) + \sum_{i=1}^{N} \alpha_i$$

$$\text{s.t} \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, i = 1....N$$

将上述方程求极大转换为极小，得到与之等价的最优化问题

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N} \alpha_i \alpha_j y_i y_j (x_i \bullet x_j) - \sum_{i=1}^{N} \alpha_i$$

$$\text{s.t} \quad \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i \geq 0, i = 1....N$$

根据 KKT 条件成立

$$\nabla_w L(w,b,\alpha) = w - \sum_{i=1}^{N} \alpha_i y_i x_i = 0$$

$$\nabla_b L(w,b,\alpha) = \sum_{i=1}^{N} \alpha_i y_i = 0$$

$$\alpha_i^* (y_i (w^* \bullet x_i + b^*) - 1) = 0$$

$$y_i (w^* \bullet x_i + b^*) - 1 \geq 0$$

$$\alpha_i^* \geq 0, \quad i = 1,...N$$

由此得到

$$b^* = y_i - \sum_{i=1}^{N} \alpha_i^* y_i (x_i \bullet x_j)$$

$$w^* = \sum_{i=1}^{N} \alpha_i y_i x_i$$

线性可分支持向量机学习算法

1.4 软间隔最大化

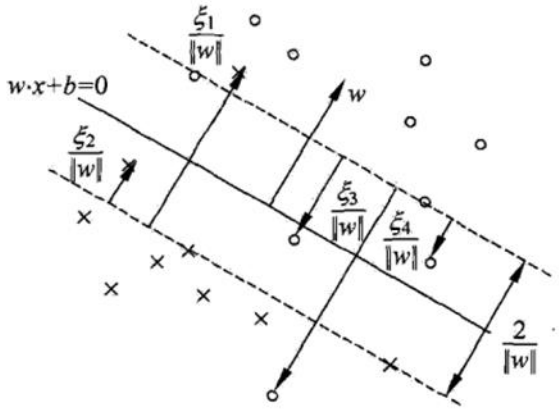


图 7.5　软间隔的支持向量

上述线性可分的支持向量机扩展到线性不可分，为了解决这个问题对每个样本(x_i, y_i)引入

松弛变量$\xi \geq 0$,约束条件变为

$$y_i(w^* \bullet x_i + b^*) \geq 1\text{-}\xi$$

目标函数变为

$$\frac{1}{2}\| w\|^2 + \mathrm{C}\sum_{\mathrm{i}=1}^{\mathrm{N}} \xi_{\mathrm{i}}$$

1.4.1 学习的对偶算法

原始对偶问题

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \bullet x_j)\text{-}\sum_{i=1}^{N}\alpha_i$$

$$\mathrm{s.t} \quad \sum_{i=1}^{N}\alpha_i y_i = 0$$

$$C \geq \alpha_i \geq 0, i = 1....N$$

原始最优化问题

$$L(w,b,\alpha,\xi,\mu) = \frac{1}{2}\| w\|^2 - \sum_{i=1}^{N}\alpha_i(y_i(w \bullet x_i + b) - 1 + \xi_i) + C\sum_{i=1}^{N}\xi_i - \sum_{i=1}^{N}\mu_i\xi_1$$

1.　）对α,ξ,μ求偏导后

$\mathrm{s.t}$

$$w = \sum_{i=1}^{N}\alpha_i y_i x_i$$

$$\sum_{i=1}^{N}\alpha_i y_i = 0$$

$$\mathrm{C}\text{-}\alpha_i - \mu_i = 0$$

$$\min_{w,b,\xi} L(w,b,\alpha,\xi,\mu) = \text{-}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \bullet x_j) + \sum_{i=1}^{N}\alpha_i$$

2）再对 $\min_{w,b,\xi} L(w,b,\alpha,\xi,\mu)$ 求对 α 的极大

$$\max_{\alpha} L(w,b,\alpha,\xi,\mu)=-\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \bullet x_j)+\sum_{i=1}^{N}\alpha_i$$

对偶问题

$$\text{s.t}\quad \sum_{i=1}^{N}\alpha_i y_i = 0$$

$$\text{C-}\alpha_i - \mu_i = 0$$

$$\alpha_i \geq 0$$

$$\mu_i \geq 0, i = 1....N$$

求解最优的 α **，最后计算**

$$w^* = \sum_{i=1}^{N}\alpha_i y_i x_i$$

$$b^* = y_i - \sum_{i=1}^{N}\alpha_i^* y_i (x_i \bullet x_j)$$

得到分离超平面

1.5 合页损失函数

$$\sum_{i=1}^{N}[1-y_i(w\bullet x_i+b)]_+ +\lambda \| w\|^2$$

最小化以下目标函数

第一项是经验损失或者经验风险 $L(y(w\bullet x+b))=[1-y(w\bullet x+b)]_+$ **称为合页损失函数。下标+表示以下取正值得函数**

$$[z]_+ =\begin{cases} z, & z>0 \\ 0, & z\leqslant 0 \end{cases}$$

这就是说，当样本点 (x_i,y_i) 被正确分类且函数间隔（确信度） $y_i(w\bullet x_i+b)$ 大于 1 时，损失是 0，否则损失是 $1-y_i(w\bullet x_i+b)$，注意到在图 7.5 中的实例点 x_4 被正确分类，但损失不是 0. 目标函数的第 2 项是系数为 λ 的 w 的 L_2 范数，是正则化项.

定理 7.4 线性支持向量机原始最优化问题：

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

$$\text{s.t.} \quad y_i(w\bullet x_i+b)\geqslant 1-\xi_i, \quad i=1,2,\cdots,N$$

$$\xi_i \geqslant 0, \quad i=1,2,\cdots,N$$

等价于最优化问题

$$\min_{w,b} \quad \sum_{i=1}^{N}[1-y_i(w\bullet x_i+b)]_+ +\lambda \| w\|^2$$

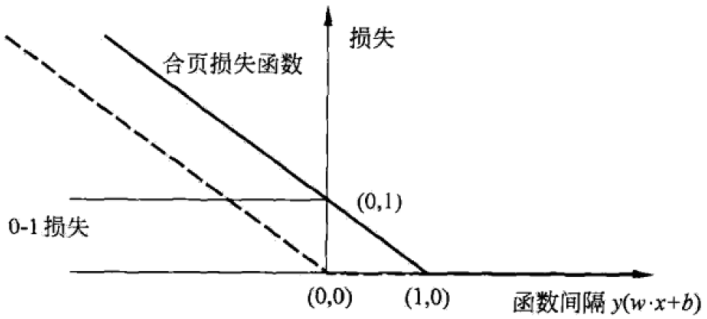
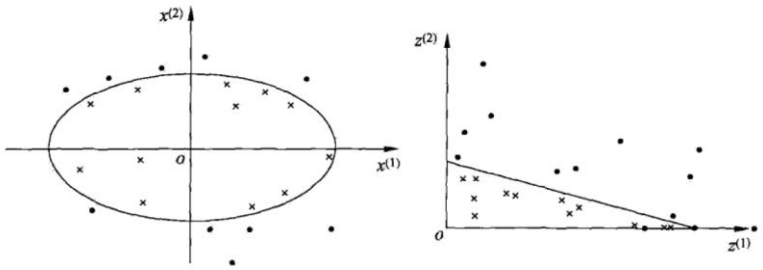


图 7.6　合页损失函数

图 7.6 中虚线显示的是感知机的损失函数$[y_i(w\cdot x_i+b)]_+$. 这时，当样本点(x_i,y_i)被正确分类时，损失是 0，否则损失是$-y_i(w\cdot x_i+b)$. 相比之下，合页损失函数不仅要分类正确，而且确信度足够高时损失才是 0. 也就是说，合页损失函数对学习有更高的要求.

二、　核函数

对解线性分类问题，线性分类分类支持向量机。但是有时分类问题时非线性的，使用非线性支持向量机，利用核技巧(kernel trick)



变换映射 $z=\phi(x)=((x^{(1)})^2,(x^{(2)})^2)^{\mathrm{T}}$

2.1 核函数定义

输入到特征空间映射

使得对所有 $x,z\in\mathcal{X}$，函数 $K(x,z)$ 满足条件

$$\phi(x):\mathcal{X}\to\mathcal{H} \qquad K(x,z)=\phi(x)\cdot\phi(z)$$

则称 $K(x,z)$ 为核函数，$\phi(x)$ 为映射函数，式中 $\phi(x)\cdot\phi(z)$ 为 $\phi(x)$ 和 $\phi(z)$ 的内积

2.2 正定核（封闭 内积空间）

定理 7.5（正定核的充要条件） 设 $K:\mathcal{X}\times\mathcal{X}\to\mathbf{R}$ 是对称函数，则 $K(x,z)$ 为正定核函数的充要条件是对任意 $x_i\in\mathcal{X}$，$i=1,2,\cdots,m$，$K(x,z)$ 对应的 Gram 矩阵：

$$K=\left[K(x_i,x_j)\right]_{m\times m}$$ 半正定矩阵

2.3 常用核函数

1. 多项式核函数 $K(x,z)=(x\cdot z+1)^p$ ，对应支持向量机 p 次多项式分类器，

   决策函数 $f(x)=\text{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i(x_i\cdot x+1)^p+b^*\right)$

2. 高斯核函数 $K(x,z)=\exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)$ ，对应高斯径向基函数分类器

$$f(x)=\text{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i\exp\left(-\frac{\|x-z\|^2}{2\sigma^2}\right)+b^*\right)$$

3. 字符串核函数

   在文本分类、信息检索、生物信息学

$$k_n(s,t)=\sum_{u\in\Sigma^n}[\phi_n(s)]_u[\phi_n(t)]_u=\sum_{u\in\Sigma^n}\sum_{(i,j):s(i)=t(j)=u}\lambda^{l(i)}\lambda^{l(j)}$$

三、 SMO 序列最小最优化算法

博客：http://blog.csdn.net/luoshixian099/article/details/51227754

支持向量机的实现，支持向量机可以转化为图二次规划问题，凸二次优化问题具有全局最优解。但是训练样本数据量很大时，这些算法很低效。

首先 SMO 需要解如下凸二次规划问题

$$\min_{\alpha}\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j(x_i\cdot x_j)-\sum_{i=1}^{N}\alpha_i$$

$$\text{s.t}\quad\sum_{i=1}^{N}\alpha_i y_i=0$$

$$\alpha_i\geq 0, i=1....N$$

变量时拉格朗日乘子 α_i ，变量总数是训练样本容量 N。

SMO 算法是一种启发式算法，其基本思路是：如果所有变量的解都满足此最优化问题的 KKT 条件（Karush-Kuhn-Tucker conditions），那么这个最优化问题的解就得到了．因为 KKT 条件是该最优化问题的充分必要条件．否则，选择两个变量，固定其他变量，针对这两个变量构建一个二次规划问题．这个二次规划问题关于这两个变量的解应该更接近原始二次规划问题的解，因为这会使得原始二次规划问题的目标函数值变得更小．重要的是，这时子问题可以通过解析方法求解，这样就可以大大提高整个算法的计算速度．子问题有两个变量，一个是违反 KKT 条件最严重的那一个，另一个由约束条件自动确定．如此，SMO 算法将原问题不断分解为子问题并对子问题求解，进而达到求解原问题的目的．

子问题的两个变量中只有一个是自由变量．假设 α_1，α_2 为两个变量 $\alpha_3,\alpha_4,\cdots,\alpha_N$ 固定

$$\alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N} \alpha_i y_i = C$$

$$\alpha_1 = (C - \alpha_2 y_2) y_1 = -y_1 \sum_{i=2}^{N} \alpha_i y_i$$

1. 两个变量二次规划的解析方法

SMO 的最优化子问题写成

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

$$-(\alpha_1 + \alpha_2) + y_1 \alpha_1 \sum_{i=3}^{N} y_i \alpha_i K_{i1} + y_2 \alpha_2 \sum_{i=3}^{N} y_i \alpha_i K_{i2}$$

$$\text{s.t.} \quad \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N} y_i \alpha_i = \varsigma$$

$$0 \leqslant \alpha_i \leqslant C, \quad i = 1, 2$$

最后解

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

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

2. 变量选择启发式方法

2.1 第一个变量选择，在训练样本中选取违反 KKT 条件最严重的样本点，并将其对应的变量作为第一个变量，检验样本点是否满足 KKT 条件。

$$\alpha_i = 0 \Leftrightarrow y_i g(x_i) \geqslant 1$$

$$0 < \alpha_i < C \Leftrightarrow y_i g(x_i) = 1$$

$$\alpha_i = C \Leftrightarrow y_i g(x_i) \leqslant 1$$

$$g(x_i) = \sum_{j=1}^{N} \alpha_j y_j K(x_i, x_j) + b$$

该检验是在 ε 范围内进行的．在检验过程中，外层循环首先遍历所有满足条件 $0 < \alpha_i < C$ 的样本点，即在间隔边界上的支持向量点，检验它们是否满足 KKT 条件．如果这些样本点都满足 KKT 条件，那么遍历整个训练集，检验它们是否满足 KKT 条件．

2.2 第二个变量选择

假设第一个变量 α_1 已经找到，第二个变量 α_2 选择标准时有足够大的变化。

在特殊情况下，如果内层循环通过以上方法选择的α_2不能使目标函数有足够的下降，那么采用以下启发式规则继续选择α_2. 遍历在间隔边界上的支持向量点，依次将其对应的变量作为α_2试用，直到目标函数有足够的下降. 若找不到合适的α_2，那么遍历训练数据集；若仍找不到合适的α_2，则放弃第 1 个α_1，再通过外层循环寻求另外的α_1.

2.3 计算阈值 b 和差值 E

每次完成两个变量优化后，都需要重新计算阈值 b，当$0 \le \alpha_1^{\text{new}} \le C$时，由 KKT 条件

$$\sum_{i=1}^{N} \alpha_i y_i K_{i1} + b = y_1$$

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

E 的计算

由 $$E_i = g(x_i) - y_i = \left(\sum_{j=1}^{N} \alpha_j y_j K(x_j, x_i) + b \right) - y_i, \quad i = 1,2$$

$$E_1 = \sum_{i=3}^{N} \alpha_i y_i K_{i1} + \alpha_1^{\text{old}} y_1 K_{11} + \alpha_2^{\text{old}} y_2 K_{21} + b^{\text{old}} - y_1$$

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

SMO 算法

输入：训练数据集$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，其中，$x_i \in \mathcal{X} = \mathbf{R}^n$，$y_i \in \mathcal{Y} = \{-1, +1\}$，$i = 1, 2, \cdots, N$，精度$\varepsilon$；

输出：近似解$\hat{\alpha}$.

（1）取初值$\alpha^{(0)} = 0$，令$k = 0$；

（2）选取优化变量$\alpha_1^{(k)}, \alpha_2^{(k)}$，解析求解两个变量的最优化问题 (7.101) ～ (7.103)，求得最优解$\alpha_1^{(k+1)}, \alpha_2^{(k+1)}$，更新$\alpha$为$\alpha^{(k+1)}$；

（3）若在精度ε范围内满足停机条件

$$\sum_{i=1}^{N} \alpha_i y_i = 0$$

$$0 \leqslant \alpha_i \leqslant C, \quad i = 1, 2, \cdots, N$$

$$y_i \cdot g(x_i) = \begin{cases} \geqslant 1, & \{x_i \mid \alpha_i = 0\} \\ = 1, & \{x_i \mid 0 < \alpha_i < C\} \\ \leqslant 1, & \{x_i \mid \alpha_i = C\} \end{cases}$$

其中，

$$g(x_i)=\sum_{j=1}^{N}\alpha_j y_j K(x_j,x_i)+b$$

则转 (4)；否则令 $k=k+1$，转 (2)；

（4）取 $\hat{\alpha}=\alpha^{(k+1)}$.

总结：

优点：

1.可用于线性/非线性分类，也可以用于回归，泛化错误率低，计算开销不大，结果容易解释；

2.可以解决小样本情况下的机器学习问题，可以解决高维问题 可以避免神经网络结构选择和局部极小点问题。

3.SVM 是最好的现成的分类器，现成是指不加修改可直接使用。并且能够得到较低的错误率，SVM 可以对训练集之外的数据点做很好的分类决策。

缺点：

对参数调节和和函数的选择敏感，原始分类器不加修改仅适用于处理二分类问题。

实现

```
>>> from sklearn import svm
>>> X = [[0, 0], [1, 1]]
>>> y = [0, 1]
>>> clf = svm.SVC()
>>> clf.fit(X, y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape=None, degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
>>> clf.predict([[2., 2.]])
array([1])
>>> clf.support_vectors_
array([[ 0.,  0.],
       [ 1.,  1.]])
array([[ 0.,  0.],
       [ 1.,  1.]])
>>> clf.predict([[-1., -1.]])
array([0])
>>> clf.predict([[-1., -1.],[-1., -1.1.],[1., 1.]])
SyntaxError: invalid syntax
>>> clf.predict([[-1., -1.],[-1., -1.1],[1., 1.]])
array([0, 0, 1])
>>>
```

多分类 one-organist-one　　/one-rest

```
>>> X = [[0], [1], [2], [3]]
>>> Y = [0, 1, 2, 3]
>>> from sklearn import svm
>>> clf = svm.SVC(decision_function_shape='ovo')
>>> clf.fit(X, Y)
SVC(C=1.0, cache_size=200, class_weight=None, coef0=0.0,
  decision_function_shape='ovo', degree=3, gamma='auto', kernel='rbf',
  max_iter=-1, probability=False, random_state=None, shrinking=True,
  tol=0.001, verbose=False)
>>> dec = clf.decision_function([[1]])
>>> dec.shape[1]
6L
>>> clf.predict([[2]])
array([2])
>>> clf.predict([[20]])
array([3])
>>>
```

四、多核 SVM

核函数：SVM 遇到线性不可分问题时，可以通过核函数将向量映射到高维空间，在高维空间线性可分

多核学习：在利用 SVM 进行训练时，会涉及核函数的选择问题，譬如线性核，rbf 核等等，多核即为融合几种不同的核来训练。

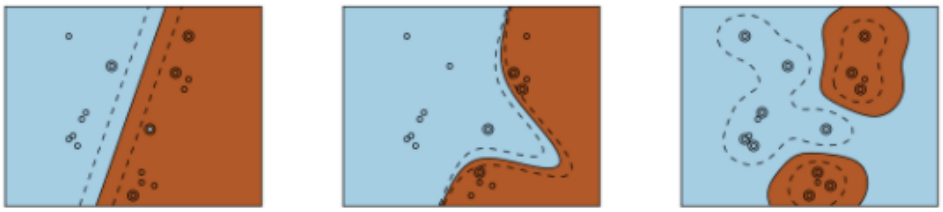
SVM 都是单核（single kernel）的，在使用的时候，需要我们根据经验或试验来选择用哪种核函数、怎样指定它的参数，这样很不方便。另一方面，实际应用当中，特征往往不是 single domain 的，而是异构的。拿图像分类来说，我们可能用到颜色相关的特征、纹理相关的特征、空间相关的特征，这几类特征对应的最佳的核函数未必相同，让他们共用同一个核函数，未必能得到最优的映射。对这些问题的思考，就引出了 MKL。

简单地说，我们给定一些 basekernels，比如 linear,Polynomial,RBF,Sigmoid，对于每一个，可以指定多组参数，也就是一共有 M 个 base kernels，我们想用它们的线性组合来作为最终的核函数。通过 training，得到这个线性组合中每个 kernel 的权重 d（weight）。由于融合了各种 kernel，可以照顾到异构的特征；由于自动学习权重，我们就不需要费脑子想究竟用哪一个核哪一种参数，把可能的核、参数都拿过来，组合着来用就可以了。

多核学习(MKL)可能是个不错的选择，该方法属于后期融合的一种，通过对不同的特征采取不同的核，对不同的参数组成多个核，然后训练每个核的权重，选出最佳核函数组合来进行分类。

与传统的单核 SVM 的不同，就是除了要学习 w、b 之外，还要学习上面提到的权重 d。这样的话，decision function, cost function 都会有些变化，棘手的是，cost function 的求解不再是一个 convex problem，传统的优化方法就不适用了。近年来 MKL 比较热，很多论文都是在优化方法上面下功夫，企图达到更快的收敛速度、更好的解。

'linear', 'poly', 'rbf



五 核方法：

基于核的算法中最著名的莫过于支持向量机（SVM）了。 基于核的算法把输入数据映射到一个高阶的向量空间， 在这些高阶向量空间里， 有些分类或者回归问题能够更容易的解决。 常见的基于核的算法包括：支持向量机（Support Vector Machine， SVM）， 径向基函数（Radial Basis Function ， RBF)， 以及线性判别分析（Linear Discriminate Analysis ， LDA)等。

就规模而言，其中一些最主要的问题已经使用支持向量机解决了（通过适当的修改），如，入广告显示，人类的剪接位点识别，基于图像的性别检测，大规模图像分类等等。

2.SMO原理分析

2.1视为一个二元函数

为了求解N个参数$(\alpha_1, \alpha_2, \alpha_3, \ldots, \alpha_N)$，首先想到的是坐标上升的思路，例如求解$\alpha_1$,可以固定其他N-1个参数，可以看成关于$\alpha_1$的一元函数求解，但是注意到上述问题的等式约束条件$\sum_{i=1}^{N} y_i \alpha_i = 0$,当固定其他参数时，参数$\alpha_1$也被固定，因此此种方法不可用。
SMO算法选择同时优化两个参数，固定其他N-2个参数，假设选择的变量为α_1, α_2,固定其他参数$\alpha_3, \alpha_4, \ldots, \alpha_N$,由于参数$\alpha_3, \alpha_4, \ldots, \alpha_N$的固定，可以简化目标函数为只关于$\alpha_1, \alpha_2$的二元函数，$Constant$表示常数项(不包含变量$\alpha_1, \alpha_2$的项)。

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

其中$v_i = \sum_{j=3}^{N} \alpha_j y_j K(x_i, x_j), i = 1, 2$

2.2视为一元函数

由等式约束得：$\alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N} \alpha_i y_i = \zeta$，可见$\zeta$为定值。
等式$\alpha_1 y_1 + \alpha_2 y_2 = \zeta$两边同时乘以$y_1$,且$y_1^2 = 1$，得

$$\alpha_1 = (\zeta - y_2\alpha_2) y_1 \qquad (2)$$

(2)式带回到(1)中得到只关于参数α_2的一元函数，由于常数项不影响目标函数的解，以下省略掉常数项$Constant$

$$min\ \Psi(\alpha_2) = \frac{1}{2} K_{11}(\zeta - \alpha_2 y_2)^2 + \frac{1}{2} K_{22}\alpha_2^2 + y_2 K_{12}(\zeta - \alpha_2 y_2)\alpha_2 - (\zeta - \alpha_2 y_2) y_1 - \alpha_2 + v_1(\zeta - \alpha_2 y_2) + y_2 v_2 \alpha_2 \qquad (3)$$

2.3对一元函数求极值点

上式中是关于变量α_2的函数，对上式求导并令其为0得：
$\frac{\partial \Psi(\alpha_2)}{\partial \alpha_2} = (K_{11} + K_{22} - 2K_{12})\alpha_2 - K_{11}\zeta y_2 + K_{12}\zeta y_2 + y_1 y_2 - 1 - v_1 y_2 + v_2 y_2 = 0$

1.由上式中假设求得了α_2的解，带回到(2)式中可求得α_1的解，分别记为$\alpha_1^{new}, \alpha_2^{new}$,优化前的解记为$\alpha_1^{old}, \alpha_2^{old}$;由于参数$\alpha_3, \alpha_4, \ldots, \alpha_N$固定，由等式约束$\sum_{i=1}^{N} y_i \alpha_i = 0$有$\alpha_1^{old} y_1 + \alpha_2^{old} y_2 = -\sum_{i=3}^{N} \alpha_i y_i = \alpha_1^{new} y_1 + \alpha_2^{new} y_2 = \zeta$

$$\zeta = \alpha_1^{old} y_1 + \alpha_2^{old} y_2 \qquad (4)$$

2.假设SVM超平面的模型为$f(x) = w^T x + b$,上一篇中已推导出w的表达式,将其带入得$f(x) = \sum_{i=1}^{N} \alpha_i y_i K(x_i, x) + b$; $f(x_i)$表示样本x_i的预测值,y_i表示样本x_i的真实值，定义E_i表示预测值与真实值之差为

$$E_i = f(x_i) - y_i \qquad (5)$$

3.由于$v_i = \sum_{j=3}^{N} \alpha_j y_j K(x_i, x_j), i = 1, 2$，因此

$$v_1 = f(x_1) - \sum_{j=1}^{2} y_j \alpha_j K_{1j} - b \qquad (6)$$

$$v_2 = f(x_2) - \sum_{j=1}^{2} y_j \alpha_j K_{2j} - b \qquad (7)$$

把(4)(6)(7)带入下式中：

$$(K_{11}+K_{22}-2K_{12})\alpha_2-K_{11}\zeta y_2+K_{12}\zeta y_2+y_1y_2-1-v_1y_2+v_2y_2=0$$

化简得: 此时求解出的α_2^{new}未考虑约束问题，先记为$\alpha_2^{new,unclipped}$：

$$(K_{11}+K_{22}-2K_{12})\alpha_2^{new,unclipped}=(K_{11}+K_{22}-2K_{12})\alpha_2^{old}+y_2\left[y_2-y_1+f(x_1)-f(x_2)\right]$$

带入(5)式，并记$\eta=K_{11}+K_{22}-2K_{12}$得：

$$\alpha_2^{new,unclipped}=\alpha_2^{old}+\frac{y_2(E_1-E_2)}{\eta}\qquad(8)$$

2.4对原始解修剪

上述求出的解未考虑到约束条件：

- $0\leq\alpha_{i=1,2}\leq C$
- $\alpha_1y_1+\alpha_2y_2=\zeta$

在二维平面上直观表达上述两个约束条件

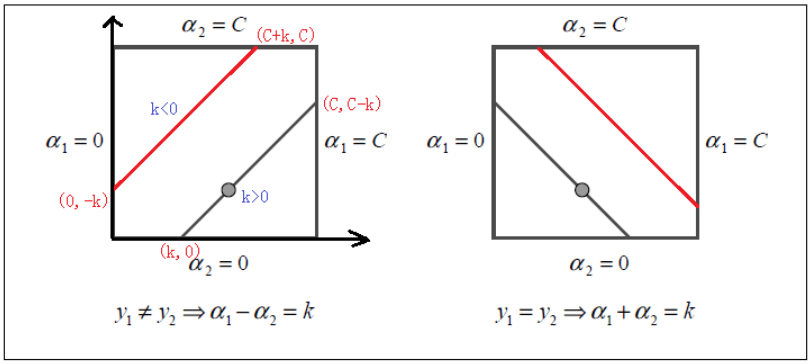


Figure 1. The two Lagrange multipliers must fulfill all of the constraints of the full problem. The inequality constraints cause the Lagrange multipliers to lie in the box. The linear equality constraint causes them to lie on a diagonal line. Therefore, one step of SMO must find an optimum of the objective function on a diagonal line segment.

最优解必须要在方框内且在直线上取得，因此$L\leq\alpha_2^{new}\leq H$;

当$y_1\neq y_2$时，$L=max(0,\alpha_2^{old}-\alpha_1^{old});H=min(C,C+\alpha_2^{old}-\alpha_1^{old})$

当$y_1=y_2$时，$L=max(0,\alpha_1^{old}+\alpha_2^{old}-C);H=min(C,\alpha_2^{old}+\alpha_1^{old})$

经过上述约束的修剪，最优解就可以记为α_2^{new}了。

$$\alpha_2^{new}=\begin{cases}\mathrm{H},\alpha_2^{new,unclipped}>H\\ \alpha_2^{new,unclipped},L\leqslant\alpha_2^{new,unclipped}\leqslant H\\ \mathrm{L},\alpha_2^{new,unclipped}<L\end{cases}$$

2.5求解α_1^{new}

由于其他N-2个变量固定，因此$\alpha_1^{old}y_1+\alpha_2^{old}y_2=\alpha_1^{new}y_1+\alpha_2^{new}y_2$所以可求得

$$\alpha_1^{new}=\alpha_1^{old}+y_1y_2(\alpha_2^{old}-\alpha_2^{new})\qquad(9)$$

2.6取临界情况

大部分情况下，有$\eta=K_{11}+K_{22}-2K_{12}>0$。但是在如下几种情况下，$\alpha_2^{new}$需要取临界值L或者H.

1. $\eta<0$,当核函数K不满足Mercer定理时，矩阵K非正定；
2. $\eta=0$,样本x_1与x_2输入特征相同；

也可以如下理解，对(3)式求二阶导数就是$\eta=K_{11}+K_{22}-2K_{12}$,

当$\eta<0$时，目标函数为凸函数，没有极小值，极值在定义域边界处取得。

当$\eta=0$时，目标函数为单调函数，同样在边界处取极值。

3.启发式选择变量

上述分析是在从N个变量中已经选出两个变量进行优化的方法，下面分析如何高效地选择两个变量进行优化，使得目标函数下降的最快。

第一个变量的选择

第一个变量的选择称为外循环，首先遍历整个样本集，选择违反KKT条件的α_i作为第一个变量，接着依据相关规则选择第二个变量(见下面分析)，对这两个变量采用上述方法进行优化。当遍历完整个样本集后，遍历非边界样本集($0 < \alpha_i < C$)中违反KKT的α_i作为第一个变量，同样依据相关规则选择第二个变量，对此两个变量进行优化。当遍历完非边界样本集后，再次回到遍历整个样本集中寻找，即在整个样本集与非边界样本集上来回切换，寻找违反KKT条件的α_i作为第一个变量。直到遍历整个样本集后，没有违反KKT条件α_i，然后退出。
边界上的样本对应的$\alpha_i = 0$或者$\alpha_i = C$，在优化过程中很难变化，然而非边界样本$0 < \alpha_i < C$会随着对其他变量的优化会有大的变化。

KKT条件

$$\begin{aligned} \alpha_i = 0 &\Rightarrow y^{(i)}(w^T x^{(i)} + b) \geq 1 \\ \alpha_i = C &\Rightarrow y^{(i)}(w^T x^{(i)} + b) \leq 1 \\ 0 < \alpha_i < C &\Rightarrow y^{(i)}(w^T x^{(i)} + b) = 1. \end{aligned}$$

第二个变量的选择

SMO称第二个变量的选择过程为内循环，假设在外循环中找个第一个变量记为α_1，第二个变量的选择希望能使α_2有较大的变化，由于α_2是依赖于$|E_1 - E_2|$,当E_1为正时，那么选择最小的E_i作为E_2,如果E_1为负，选择最大E_i作为E_2，通常为每个样本的E_i保存在一个列表中，选择最大的$|E_1 - E_2|$来近似最大化步长。

有时按照上述的启发式选择第二个变量，不能够使得函数值有足够的下降，这时按下述步骤:

> 首先在非边界集上选择能够使函数值足够下降的样本作为第二个变量，
> 如果非边界集上没有，则在整个样本集上选择第二个变量，
> 如果整个样本集依然不存在，则重新选择第一个变量。

4.阈值b的计算

每完成对两个变量的优化后，要对b的值进行更新，因为b的值关系到f(x)的计算，即关系到下次优化时E_i的计算。
1.如果$0 < \alpha_1^{new} < C$，由KKT条件$y_1(w^T x_1 + b) = 1$,得到$\sum_{i=1}^{N} \alpha_i y_i K_{i1} + b = y_1$,由此得：

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

由(5)式得，上式前两项可以替换为：

$$y_1 - \sum_{i=3}^{N} \alpha_i y_i K_{i1} = -E_1 + \alpha_1^{old} y_1 K_{11} + \alpha_2^{old} y_2 K_{11} + b^{old}$$

得出：

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

2.如果$0 < \alpha_2^{new} < C$，则

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

3.如果同时满足$0 < \alpha_i^{new} < C$,则$b_1^{new} = b_2^{new}$
4.如果同时不满足$0 < \alpha_i^{new} < C$，则b_1^{new}与b_2^{new}以及它们之间的数都满足KKT阈值条件，这时选择它们的中点。