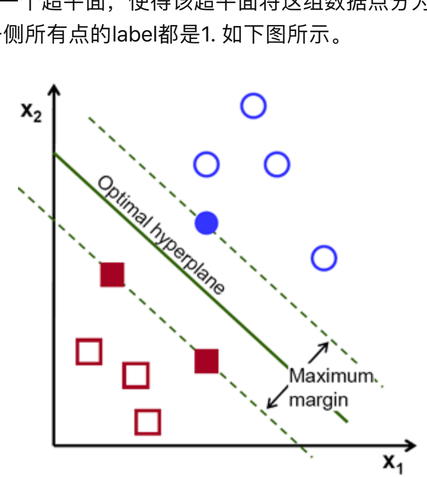
机器学习算法 II : support vector machine (SVM)

拉普拉斯算符
数学话题下的优秀答主

第一节：导论

Support vector machine 最早是用来处理线性可分问题的。假设我们有一组数据 $(\boldsymbol{x}_i, y_i), i = 1, 2, \dots, N$. 其中，矢量 $\boldsymbol{x}_i$ 是一组feature，标量 y_i 为该矢量所对应的label. 规定label的取值只能是 -1 或者 1. 对于线性可分问题，我们可以在由矢量 $\boldsymbol{x}_i$ 张成的空间中画出所有的数据点，并且可以找到一个超平面，使得该超平面将这组数据点分为两类。超平面一侧的所有点的label都是 -1，另外一侧所有点的label都是1. 如下图所示。



图片中是二维情形下的support vector machine的示意图。这里每一个数据点都有一个label。这些数据点可以用一条直线分离为两类。一类点label 都是1，另外一类点label都是 -1. 很明显，不止一条线可以将这组数据点分为两类。Support vector machine算法的目的是找出一条最优的分离线。最优分离线(或者对应高维情况下的最优分离面)的定义在下一节。

第二节：最优分离面的定义

把数据点记作 $(\boldsymbol{x}_i, y_i), i = 1, 2, \dots, N$. 假设分离平面为 $\beta^T \boldsymbol{x} + \beta_0 = 0$. 位于平面一侧的点满足关系 $\beta^T \boldsymbol{x} + \beta_0 > 0$ ，另一侧的点满足关系 $\beta^T \boldsymbol{x} + \beta_0 < 0$. 通过合适的scaling，我们总是可以令所有 label 为 1 的点满足关系 $\beta^T \boldsymbol{x} + \beta_0 \geq 1$ ，所有 label 为 -1 的点满足关系 $\beta^T \boldsymbol{x} + \beta_0 \leq -1$. 总是存在一组参数 β, β_0 ，使得平面 $P_1 : \beta^T \boldsymbol{x} + \beta_0 = 1$ 与 $P_2 : \beta^T \boldsymbol{x} + \beta_0 = -1$ 之间的距离最大。定义此时的分离平面 $\beta^T \boldsymbol{x} + \beta_0 = 0$ 为最优分离面。接下来我们要计算平面 P_1 与 P_2 之间的距离。

已知一点 $\boldsymbol{x}_0$ 到线性集合 $A\boldsymbol{x} = b$ 的距离的平方为 $d^2 = (b - A\boldsymbol{x}_0)^T (AA^T)^{-1} (b - A\boldsymbol{x}_0)$. 如果已知点 $\boldsymbol{x}_0$ 位于平面 P_2 上，也就是该点满足 $\beta^T \boldsymbol{x}_0 + \beta_0 = -1$. 那么该点距离平面 P_1 的距离为

$d = \frac{2}{||\beta||_2}$. 所以最优分离面的计算可以归结为下面的优化问题：

$$
\begin{aligned}
&\min_\beta \frac{1}{2}||\beta||^2 \\
\text{s.t.} \ &-y_i(\beta^T \boldsymbol{x}_i + \beta_0) \leq -1, i = 1, 2, \dots, N
\end{aligned}
$$

这个问题并不容易求解。为了求解这个问题，我们需要引入拉格朗日对偶关系。

第三节：拉格朗日对偶

上一节已经证明了，最优分离面的求解可以归结为计算一个受限约束条件下的极小值问题。这类问题可以通过拉格朗日乘子法来解决。定义拉格朗日函数为

$$L(\beta, \beta_0; \alpha) = \frac{1}{2}\beta^T\beta + \sum_{i=1}^N \alpha_i \Big(-y_i(\beta^T \boldsymbol{x}_i + \beta_0) + 1 \Big)$$

为了计算拉格朗日函数的极小值，根据 KKT 条件（关于KKT条件，这里有一篇文章，给出了非常好的推导: 浅谈最优化问题的KKT条件），求该函数对变量 β, β_0 的梯度，令梯度为零，得到

$$\nabla_\beta L(\beta, \beta_0; \alpha) = \beta - \sum_{i=1}^N \alpha_i y_i \boldsymbol{x}_i = 0$$

$$\frac{\partial L}{\partial \beta_0} = -\sum_{i=1}^N \alpha_i y_i = 0$$

此外，根据 KKT 条件，我们有

$$\alpha_i \geq 0, \forall i$$

$$\alpha_i \Big(-y_i(\beta^T \boldsymbol{x}_i + \beta_0) + 1 \Big) = 0, \forall i$$

由此，我们可以得到拉格朗日对偶函数为

$$g(\alpha) = \inf_{\beta, \beta_0} L(\beta, \beta_0; \alpha) = \sum_{i=1}^N \alpha_i - \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$$

利用拉格朗日对偶关系，原来的极小化问题可以转化为一个极大化问题：

$$
\begin{aligned}
&\max_\alpha g(\alpha) \\
\text{s.t.} \ &\alpha_i \geq 0, i = 1, 2, \dots, N \\
&\sum_{i=1}^N \alpha_i y_i = 0
\end{aligned}
$$

相比原问题，这个问题更容易求解。下一节将会给出求解该类问题的一个算法。

第四节：内点法求解拉格朗日对偶问题

上一节已经将计算最优分离面问题转化为它的拉格朗日对偶问题，该问题可以重新写作一种等价形式为：

$$
\begin{aligned}
&\min_\alpha -g(\alpha) \\
\text{s.t.} \ &-\alpha_i \leq 0, i = 1, 2, \dots, N \\
&\sum_{i=1}^N \alpha_i y_i = 0
\end{aligned}
$$

为了求解这个问题，我们可以用内点法(interior point method)。在这里我已经描述过内点法在线性规划问题中的应用：凸优化算法 I: 内点法(interior point method)求解线性规划问题

同样可以使用内点法求解现在的拉格朗日对偶问题。为此，定义函数：

$$h(\alpha, \lambda; t) = -\sum_{i=1}^N \alpha_i + \frac{1}{2} \sum_{i=1}^N \sum_{j=1}^N \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j + \sum_{i=1}^N -\frac{1}{t} \log \alpha_i + \lambda \sum_{i=1}^N \alpha_i y_i$$

定义对称矩阵 $A_{ij} = y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$. 这个矩阵可以进一步写作 $A = \xi^T \xi$, 其中 $\xi = (y_1 \boldsymbol{x}_1, y_2 \boldsymbol{x}_2, \dots, y_N \boldsymbol{x}_N)$. 因为通常情况下点的个数远大于矢量 $\boldsymbol{x}_i$ 的维数，所以矩阵 A 为不可逆矩阵。利用这个记号，函数 $h(\alpha, \lambda; t)$ 可以写作

$$h(\alpha, \lambda; t) = -\sum_{i=1}^N \alpha_i + \frac{1}{2} \alpha^T A \alpha + \sum_{i=1}^N -\frac{1}{t} \log \alpha_i + \lambda \alpha^T y$$

对于固定的参数 t ，我要计算使得该函数取得极值时变量 α, λ 的值. 一旦知道了 α 的值，我们就可以通过 $\beta = \sum_{i=1}^N \alpha_i y_i \boldsymbol{x}_i$ 求出最优分离面的法向量 β 的值，并且进一步求出截距 β_0 . 注意，α 为长度为 N 的矢量，λ 为一个标量。

对变量 α, λ 求微分，得到

$$\frac{\partial h}{\partial \alpha_k} = -1 + A_{kj}\alpha_j - \frac{1}{t}\frac{1}{\alpha_k} + \lambda y_k$$

$$\frac{\partial h}{\partial \lambda} = \alpha^T y$$

令微分等零，得到一个方程组

$$\begin{pmatrix} \frac{\partial h}{\partial \alpha} \\ \frac{\partial h}{\partial \lambda} \end{pmatrix} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$$

可以用牛顿迭代法求解该方程，牛顿迭代公式为

$$\begin{pmatrix} \alpha^{(n+1)} \\ \lambda^{(n+1)} \end{pmatrix} = \begin{pmatrix} \alpha^{(n)} \\ \lambda^{(n)} \end{pmatrix} - H^{-1} \begin{pmatrix} \frac{\partial h}{\partial \alpha} \\ \frac{\partial h}{\partial \lambda} \end{pmatrix}$$

这里，Hessian矩阵 H 为

$$H = \begin{pmatrix} A_{ij} + \frac{1}{t}\frac{\delta_{ij}}{\alpha_i^2} & y \\ y^T & 0 \end{pmatrix}$$

其中，矩阵 $A_{ij} = y_i y_j \boldsymbol{x}_i^T \boldsymbol{x}_j$.

通过牛顿迭代，可以求出对应固定参数 t 时方程的解 $(\alpha_t^\star, \lambda_t^\star)$.根据内点法的定义，我们有 $\lim_{t \to \infty} (\alpha_t^\star, \lambda_t^\star) = (\alpha^\star, \lambda^\star)$. 通过淬火算法，我们可以得到符合精度的解。

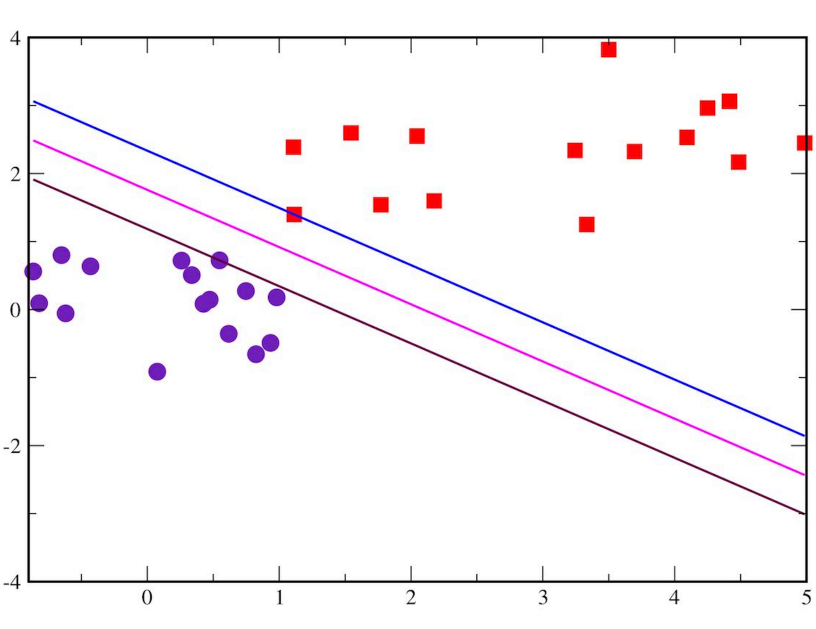
第五节：程序

我已经写了一个Python程序来实现这个算法。程序地址为：

https://github.com/PrimerLi/svm
github.com

第六节：结果

对于一组可以线性分离的点，通过上一节的程序可以解得最优分离面。如下图所示：



第七节：附录I

在这里，我将要推导如何计算空间中一点到一个线性集合的距离。这个公式在推导两个平面之间的距离时起了关键性作用。

定义一个线性集合为 $\Sigma : A\boldsymbol{x} = b$. 其中 A 为一个矩阵，b 为一个矢量。超平面可以理解为一个特殊的线性集合。空间中一点 $\boldsymbol{x}_0$ 到集合 Σ 的距离定义为:

$d = \min_{\boldsymbol{x}} ||\boldsymbol{x} - \boldsymbol{x}_0||, \boldsymbol{x} \in \Sigma$.

为了求解距离的表达式，我们可以借助凸优化算法。首先，将问题重新表述为

$$
\begin{aligned}
&\min_{\boldsymbol{x}} \frac{1}{2}||\boldsymbol{x} - \boldsymbol{x}_0||^2 \\
&\text{s.t.} \ A\boldsymbol{x} = b
\end{aligned}
$$

定义拉格朗日函数为

$$L(\boldsymbol{x}; \lambda) = \frac{1}{2}(\boldsymbol{x} - \boldsymbol{x}_0)^T (\boldsymbol{x} - \boldsymbol{x}_0) + \lambda^T (A\boldsymbol{x} - b)$$

求梯度:

$$\nabla_{\boldsymbol{x}} L = \boldsymbol{x} - \boldsymbol{x}_0 + A^T \lambda$$

令梯度为零，得到 $\boldsymbol{x} = \boldsymbol{x}_0 - A^T \lambda$. 拉格朗日对偶函数为

$$g(\lambda) = \inf_{\boldsymbol{x}} L(\boldsymbol{x}; \lambda) = \frac{1}{2}\lambda^T A A^T \lambda + \lambda^T (A\boldsymbol{x}_0 - AA^T\lambda - b) = -\frac{1}{2}\lambda^T AA^T \lambda + \lambda^T (A\boldsymbol{x}_0 - b)$$

$$\nabla_\lambda g(\lambda) = -AA^T \lambda + A\boldsymbol{x}_0 - b$$

令梯度为零，求得 $\lambda^\star = (AA^T)^{-1}(A\boldsymbol{x}_0 - b)$. 这里，因为线性集合的定义中，约束条件的个数小于变量的个数，所以矩阵 AA^T 可逆，并且该矩阵为正定矩阵。将 $\lambda^\star$ 代入 $g(\lambda)$ ，得到

$$g(\lambda^\star) = \max_\lambda g(\lambda) = \frac{1}{2}(A\boldsymbol{x}_0 - b)^T (AA^T)^{-1} (A\boldsymbol{x}_0 - b) = \frac{1}{2}d^2$$

所以，空间中一点到线性集合的距离为

$$d = \sqrt{(A\boldsymbol{x}_0 - b)^T (AA^T)^{-1} (A\boldsymbol{x}_0 - b)}$$

对于一张平面 $P : \beta^T \boldsymbol{x} + \beta_0 = 0$ ，空间中一点 $\boldsymbol{x}_0$ 到该平面的距离为

$$d = \frac{||\beta^T \boldsymbol{x}_0 + \beta_0||}{||\beta||}$$

很容易看出来，两张平面 $P_1 : \beta^T \boldsymbol{x} + \beta_0 = 1; P_2 : \beta^T \boldsymbol{x} + \beta_0 = -1$ 之间的距离为

$$d = \frac{2}{||\beta||}$$

第八节：附录II

上面的全都是hard margin SVM的算法。所谓hard margin，指的是该分类方法不允许出现任何不可分的点。实际情况下，两类点未必是严格可分的，而且就算两类点是严格可分的，我们有时也希望对margin做一些松动，以错分一些点为代价换取一个更宽的margin. 这时就需要用到soft margin SVM. 该算法与hard margin SVM 算法唯一的区别是拉格朗日对偶问题中限制条件变成了 $0 \leq \alpha_i \leq C, i = 1, 2, \dots, N$ ，也就是我们的目标函数变为

$$
\begin{aligned}
&\max_\alpha g(\alpha) \\
\text{s.t.} \ &0 \leq \alpha_i \leq C, i = 1, 2, \dots, N \\
&\sum_{i=1}^N \alpha_i y_i = 0
\end{aligned}
$$

当 $\alpha_i = 0$ 时，$y_i(\beta^T \boldsymbol{x}_i + \beta_0) > 1$ ，此时点落在上下两个边界之外；$\alpha_i = C$ 时，$y_i(\beta^T \boldsymbol{x}_i + \beta_0) < 1$，此时点落在上下两个边界之间；$0 < \alpha_i < C$ 时，$y_i(\beta^T \boldsymbol{x}_i + \beta_0) = 1$ ，此时点在落在上下两条边界上。

这个受限优化问题仍然可以使用内点法求解. 构造辅助函数:

$$f(\alpha, \lambda; t_1, t_2) = -\sum_i \alpha_i + \frac{1}{2}\alpha^T A \alpha + \lambda \sum_i \alpha_i y_i - \frac{1}{t_1} \sum_i \log \alpha_i - \frac{1}{t_2} \sum_i \log(C - \alpha_i)$$

为了简化，可以令 $t_1 = t_2$, 于是辅助函数简化为

$$f(\alpha, \lambda; t) = -\sum_i \alpha_i + \frac{1}{2}\alpha^T A \alpha + \lambda \sum_i \alpha_i y_i - \frac{1}{t} \sum_i \log \alpha_i (C - \alpha_i)$$

该函数的梯度为

$$\frac{\partial f}{\partial \alpha_k} = -1 + A_{kj}\alpha_j + \lambda y_k - \frac{1}{t}\Big(\frac{1}{\alpha_k} - \frac{1}{C - \alpha_k}\Big)$$

$$\frac{\partial f}{\partial \lambda} = \alpha^T y$$

Hessian矩阵元素为

$$\frac{\partial^2 f}{\partial \alpha_k \partial \alpha_l} = A_{kl} + \frac{1}{t}\delta_{kl}\Big(\frac{1}{\alpha_k^2} + \frac{1}{(C - \alpha_k)^2}\Big)$$

$$\frac{\partial^2 f}{\partial \alpha_k \partial \lambda} = \frac{\partial^2 f}{\partial \lambda \partial \alpha_k} = y_k$$

$$\frac{\partial^2 f}{\partial \lambda^2} = 0$$

矩阵形式为

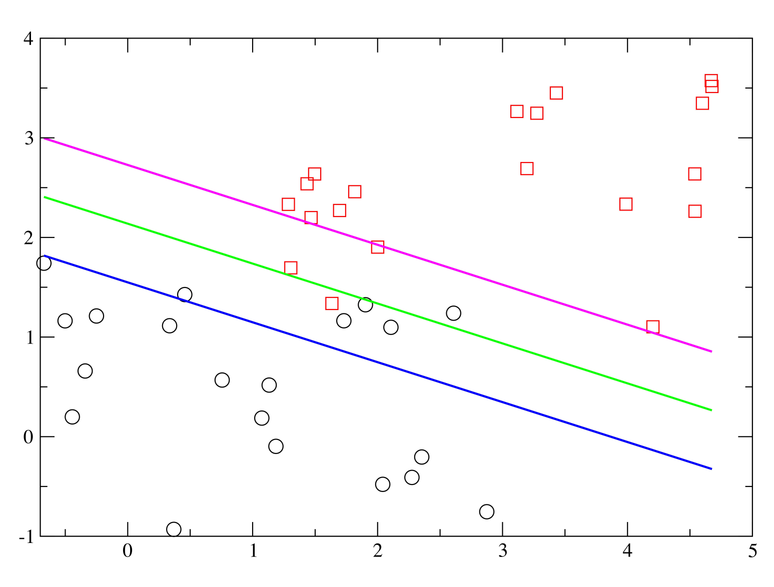
$$H = \begin{pmatrix} A_{kl} + \frac{1}{t}\delta_{kl}\Big(\frac{1}{\alpha_k^2} + \frac{1}{(C - \alpha_k)^2}\Big) & y \\ y^T & 0 \end{pmatrix}$$

当 $C \to \infty$ 时，Hessian矩阵退化为hard margin SVM的形式。

只需对原来的 SVM 程序稍作修改就可以实现soft margin SVM. 程序地址为:

https://github.com/PrimerLi/svm/blob/master/soft_margin_svm.py
github.com

结果如图所示：



如果点严格可分，那么我们仍然可以得到与SVM 相同的结果；如果有不可分的点，那么 soft margin SVM 仍然可以给出一个比较好的结果，而hard margin SVM在这时就得不到任何结果了。

编辑于 2018-05-14