SVM基本原理

笔记本：	机器学习		
创建时间：	2019/1/29 15:57	**更新时间：**	2019/1/31 16:41
标签：	SVM, 机器学习		

SVM基本原理

本文描述本文描述SVM的基本原理，主要关注基本概念和逻辑原理，不细抠数学推导细节。
参考了OpenCV的SVM手册，以及一些博客学习SVM，支持向量机通俗导论

SVM简介

SVM(Support Vector Machine)是一种高效的监督学习算法，在解决图像分类问题是有着高效的应用。

> support vector machines is the supervised learning algorithm that many people consider the most effective off-the-shelf supervised learning algorithm.That point of view is debatable, but there are many people that hold that point of view.

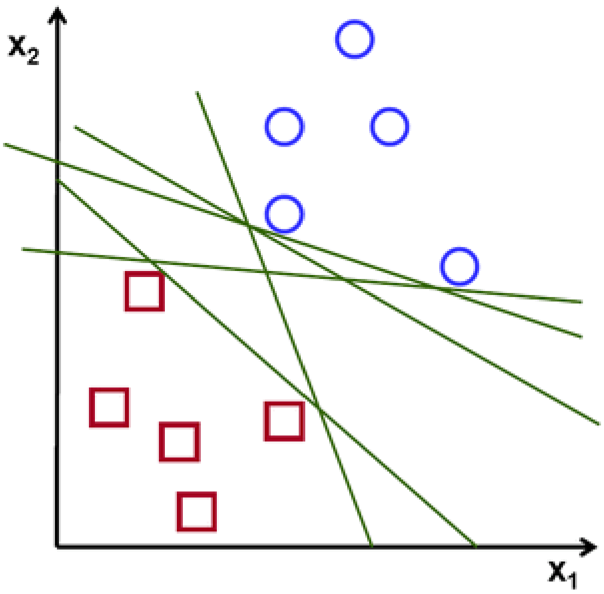
根据需要分类的数据的特征，可以将SVM分为三种情况：

1. 线性可分情况下的线性分类器，这是最原始的SVM，它最核心的思想就是最大分类间隔（margin maximization）
2. 线性不可分情况下的线性分类器，引入软间隔（soft margin）的概念
3. 线性不可分情况下的非线性分类器，是SVM与核函数（kernel function）的结合

下面依次来讨论这几种情况。

线性可分数据

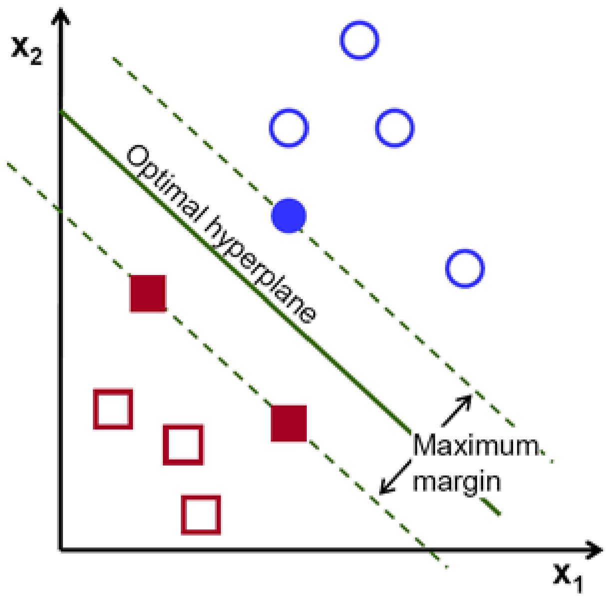
看下图中的两类数据，每个数据点p有两个自由度（x1，x2）。我们可以找到一条线 $f(x) = ax_1 + bx_2 + c$ 把两类数据分隔开来。



将任意数据点的坐标带入该直线的方程，要么 f(x)>0，要么 f(x)<0。显然，在直线上方的点，代入后则令 f(x)>0；在直线下方的点，带入后则令 f(x)<0。这条线就叫作决策边界（Decision Boundary），这种数据能被一条直线（或者一个超平面）分成两部分的数据就叫做线性可分数据。

由上图可以看出，有许多条线能将数据分为两部分，哪一条作为决策边界最好呢？
这时我们选取一条尽可能远离所有数据点的线，这样就能避免数据中噪声的影响，提高分类的精确性。

SVM就是要找到一条直线（或一个超平面），使数据点离这条直线（或者超平面）的距离最大。

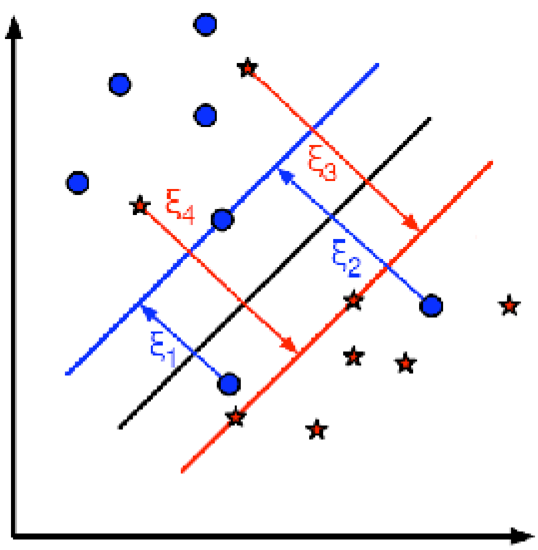


为了找到决策边界，我们需要训练数据，但并不是图中所有的数据都有用，我们只需要那些离对方最近的数据点。这些数据点称为**支持向量**，穿过支持向量的线或平面叫**支持平面。**只有支持向量对于决策边界的选择有贡献，其他数据点的贡献为0。这样就大大减少了计算量。
关于如何确定决策边界的方向和位置，如何确定支持向量，必须阅读数学推导过程。包括求极值、对偶问题以及拉格朗日乘子法等方法。

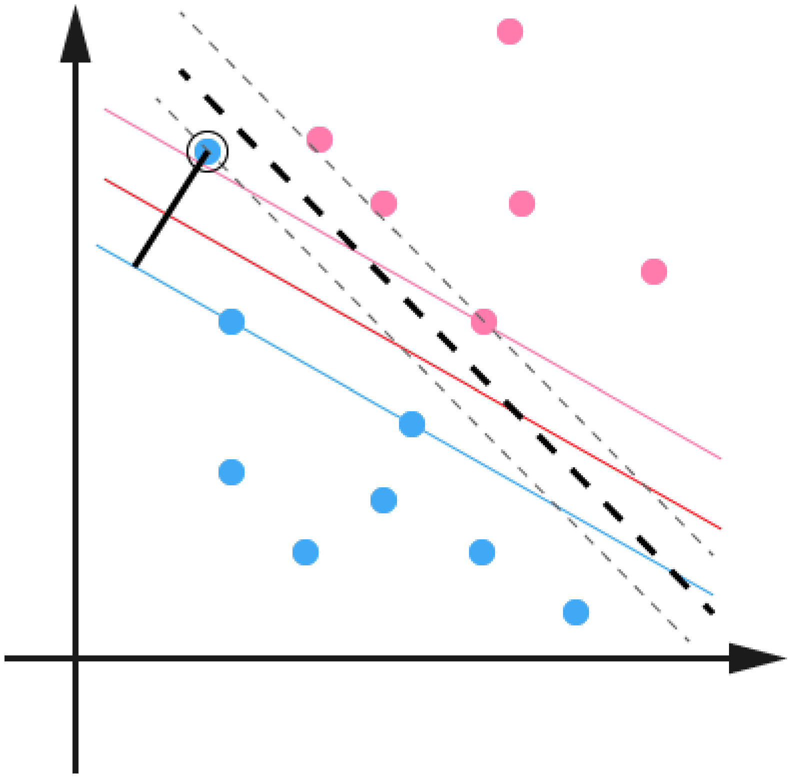
线性不可分情况

线性分类

有时候我们会遇到下图这样的数据，不能直接找到一条直线将数据点分隔在两侧。



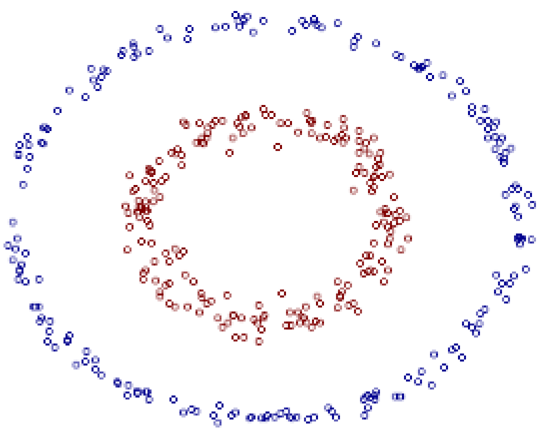
于是就引入**松弛变量**和**惩罚因子**的概念。
松弛变量就是数据点到支持平面的距离。
松弛变量使SVM找到一个决策边界，允许有数据点被误分类，但是让最大分类间隔比严格分类时要大。例如下图，如果严格分类的话，决策边界将会是黑色虚线所示的直线，但是由于松弛变量的存在，可以确定一个新的决策边界如粉红色实线，令最大分类间隔比黑色实线时的更大。



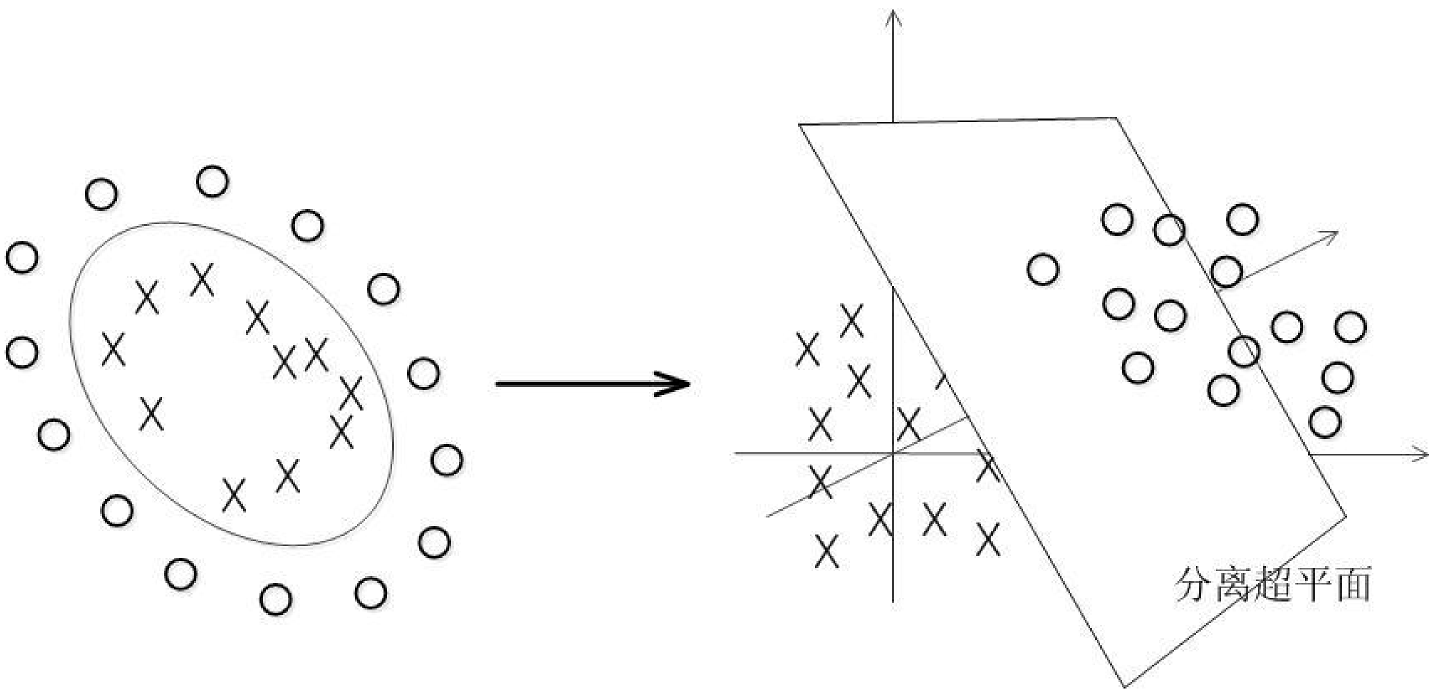
如果松弛变量选得足够大，那么任意的决策边界均能满足分类要求。就像没有底线的忍让误差一样。所以我们也要让松弛变量的总和最小。并且引入惩罚因子的概念。
先让我们回忆一下求极值时的罚函数，其中也包括一个惩罚因子。其目的是将所有偏离约束条件的微小偏差放大，趋于无穷大时我们就只能满足约束条件的限制。
这里的惩罚因子也是类似的概念，反映我们对分类误差的忍受程度。当惩罚因子选得足够大，趋于无穷时，就意味着我们完全不能忍受任何程度的分类误差，于是就又蜕变成严格分类了。
于是，我们通过惩罚因子和松弛变量两个手段，在“选取具有更大分类间隔的决策边界”和“准确率尽量高、误分类尽量低”之间取得一个平衡。当你选择更大的惩罚因子，就意味着更严格更准确的分类，但一旦出错代价也会更高；当你选择偏小的惩罚因子，分类间隔和误分类次数都会增大，换来的是更普适的分类效果。

非线性分类

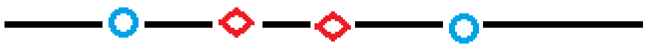
这次让我们看看下图这种特征的数据，显然不能找到一条直线将红蓝两色的数据点分隔开，最理想的分隔线应该是一个椭圆。但是SVM只能选择一条直线或者一个超平面，那我们怎么解决呢？



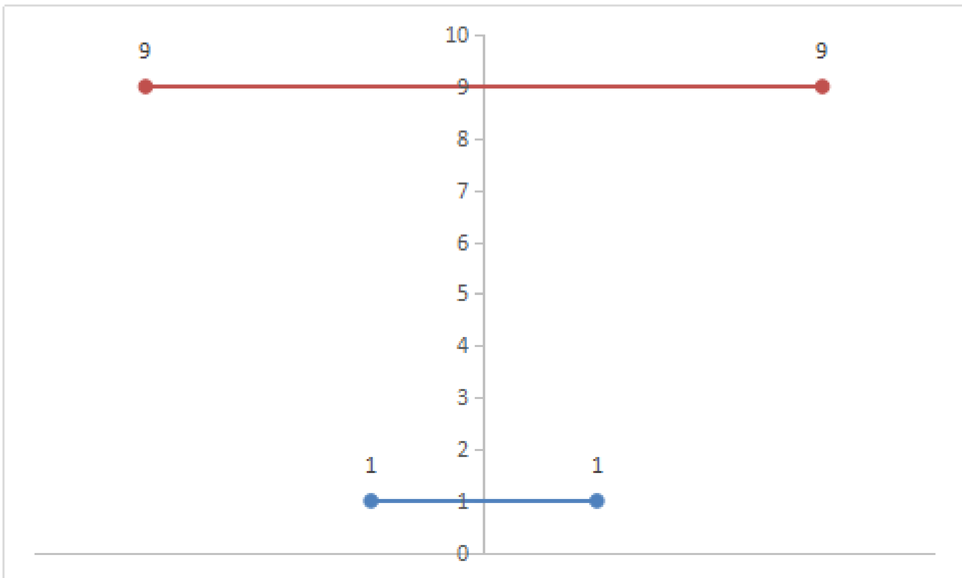
于是我们找到一个映射，将数据从低维空间映射到高维空间，再在高维空间中选择一个超平面座位决策边界。如下图：



再举个简单的例子，假设我们有两类数据，A类是 -1，1；B类是 -3，3。不可能用一个点将A，B分隔在两侧。



于是我们找到一个映射f(x)=(x，x^2)，A类数据映射为（-1,1），（1，1）；B类数据映射为（-3，9），（3，9）。在二维平面中就能很容易的找到决策边界了。



我们将映射后的高维空间称为**核空间**，如上所述，我们将在核空间中寻找决策边界，由于维度变高，所需的计算量也呈指数式增长（因为要计算向量点积），于是采取一种巧妙简单的方法在低维空间中计算高维空间的点积。接下来我们就介绍**核函数。**我们就是利用核函数，在低维空间进行本应在核空间中进行的运算。

例如，有两个二维数据点 $p = (p_1, p_2)$ 和 $q = (q_1, q_2)$，以及一个二维到三维的映射函数ϕ()。将p，q通过映射函数映射到三维空间：

$$\phi(p) = (p_1^2, p_2^2, \sqrt{2}p_1p_2)\phi(q) = (q_1^2, q_2^2, \sqrt{2}q_1q_2)$$

给定了这个映射后，我们可以定义一个核函数$K(p,q)$，令$K(p,q) = \phi(p).\phi(q)$，即输入低维空间的两个数据点，输出映射到高维空间后的两个数据点的点积。对于上面这个映射，定义后的K（p，q）为：

$$
\begin{aligned}
K(p,q) = \phi(p).\phi(q) &= \phi(p)^T, \phi(q) \\
&= (p_1^2, p_2^2, \sqrt{2}p_1p_2).(q_1^2, q_2^2, \sqrt{2}q_1q_2) \\
&= p_1^2q_1^2 + p_2^2q_2^2 + 2p_1q_1p_2q_2 \\
&= (p_1q_1 + p_2q_2)^2 \\
\phi(p).\phi(q) &= (p.q)^2
\end{aligned}
$$

于是通过核函数，我们就能在低维空间运算高维空间中的点积。前提是要给定映射函数，然后我们可以构造一个核函数，来简化运算量。无需写出映射之后的结果再运算点积，直接通过核函数给出结果即可。