SVM学习笔记20190525

第一想认真的看一个算法原理，就从SVM开始吧。 找了几篇博客，最终选了这篇，对博客的内容进行了精简，公司重新编辑了下https://blog.csdn.net/jack10086000/article/details/81358991。

重要的写在前面：

特点：非线性，是指SVM擅长应付样本数据线性不可分的情况，主要通过松弛变量（也有人叫惩罚变量）和核函数技术来实现，这一部分是SVM的精髓，后面会详细讨论。

优化目标是：最大化几何间隔

关于$g(x) = wx + b$这个表达式要注意三点：

一、式中的x不是二维坐标系中的横轴，而是样本的向量表示，例如一个样本点的坐标是(3,8)，则x^T=(3,8) ，而不是x=3；

二、这个形式并不局限于二维的情况，在n维空间中仍然可以使用这个表达式，只是式中的w成为了n维向量；

三、g(x)不是中间那条直线的表达式，中间那条直线的表达式是g(x)=0，即wx+b=0，我们也把这个函数叫做分类面。

实际上很容易看出来，中间那条分界线并不是唯一的，我们把它稍微旋转一下，只要不把两类数据分错，仍然可以达到上面说的效果，稍微平移一下，也可以。此时就牵涉到一个问题，对同一个问题存在多个分类函数的时候，哪一个函数更好呢？显然必须要先找一个指标来量化“好”的程度，通常使用的都是叫做“分类间隔”的指标。

几何间隔决定误分次数的上界

在超平面wx+b=0确定的情况下，$|wx + b|$*能够表示点x到距离超平面的远近，而通过观察*wx+b的符号与类标记y的符号是否一致可判断分类是否正确，所以，可以用$(y * (w * x + b))$的正负性来判定或表示分类的正确性。于此，我们便引出了函数间隔（functional margin）的概念。

当用归一化的w和b代替原值之后的间隔有一个专门的名称，叫做几何间隔，几何间隔所表示的正是点到超平面的欧氏距离。

之所以如此关心几何间隔这个东西，是因为几何间隔与样本的误分次数间存在关系：

误分次数 $\leq (\frac{2R}{\delta})^2$

其中的δ是样本集合到分类面的间隔，$R = max||x_i||(i = 1, 2, ...n)$，即R是所有样本中（$x_i$是以向量表示的第i个样本）向量长度最长的值（也就是说代表样本的分布有多么广）。先不必追究误分次数的具体定义和推导过程，只要记得这个误分次数一定程度上代表分类器的误差。而从上式可以看出，误分次数的上界由几何间隔决定！（当然，是样本已知的时候）

至此我们就明白为何要选择几何间隔来作为评价一个解优劣的指标了，原来几何间隔越大的解，它的误差上界越小。因此最大化几何间隔成了我们训练阶段的目标，而且，与二把刀作者所写的不同，最大化分类间隔并不是SVM的专利，而是早在线性分类时期就已有的思想。

目标函数

我们常用的方法并不是固定$||w||$的大小而寻求最大几何间隔，而是固定间隔（例如固定为1），寻找最小的$||w||$。那寻优的目标就是

$min||w||$

但实际上对于这个目标，我们常常使用另一个完全等价的目标函数来代替，那就是：

$$min\frac{1}{2}||w||^2 \qquad (1)$$

之所以采用这种形式，是因为后面的求解过程会对目标函数作一系列变换，而式（1）的形式会使变换后的形式更为简洁（正如聪明的读者所料，添加的系数二分之一和平方，皆是为求导数所需）。

那如果就只是当前的这个求最小值的问题，就很容易得到了，只要$||w|| = 0$的时候就得到了目标函数的最小值。但是你也会发现，无论你给什么样的数据，都是这个解！反映在图中，就是H_1与H_2两条直线间的距离无限大，这个时候，所有的样本点（无论正样本还是负样本）都跑到了H_1和H_2中间，而我们原本的意图是，H1右侧的被分为正类，H2 左侧的被分为负类，位于两类中间的样本则拒绝分类（拒绝分类的另一种理解是分给哪一类都有道理，因而分给哪一类也都没有道理）。这下可好，所有样本点都进入了无法分类的灰色地带。

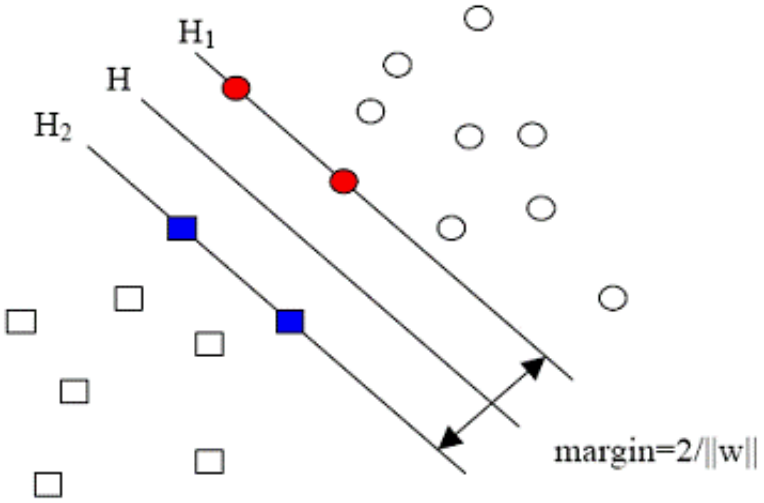


图2　线性可分情况下的最优分类线

造成这种结果的原因是在描述问题的时候只考虑了目标，而没有加入约束条件，约束条件就是在求解过程中必须满足的条件，体现在我们的问题中就是样本点必须在H1或H2的某一侧（或者至少在H1和H2上），而不能跑到两者中间。我们前文提到过把间隔固定为1，这是指把所有样本点中间隔最小的那一点的间隔定为1（这也是集合的间隔的定义，有点绕嘴），也就意味着集合中的其他点间隔都不会小于1，按照间隔的定义，满足这些条件就相当于让下面的式子总是成立：

$$y_i[(wx_i)+b] \geq 1(i=1,2,\ldots I)(I\text{是样本总数})$$

但是我们常常习惯让式子的值和0比较，因而经常用变换过的形式：

$$y_i[(w\cdot x_i)+b]-1 \geq 0(i=1,2,\ldots I)(I\text{是样本总数})$$

因此我们的两类分类问题也被我们转化成了它的数学形式，一个带约束的最小值的问题：

$$min \quad f(x)$$

$$subject\ to\ y_i[(wx_i)+b]-1 \geq 0(i=1,2,\ldots I)(I\text{是样本数})$$

到目前为止，我们的线性分类器问题只有不等式约束，因此形式上看似乎比一般意义上的规划问题要简单，但解起来却并非如此。因此我们需要把带不等式约束的问题转化成只带等式约束的问题，然后再转化成无约束问题求解。

问题转化

看出妙在哪了么？原来在二维空间中一个线性不可分的问题，映射到四维空间后，变成了线性可分的！因此这也形成了我们最初想解决线性不可分问题的基本思路——向高维空间转化，使其变得线性可分。

我们要求函数g(x)=wx+b，求这样的g(x)的过程就是求w（一个n维向量）和b（一个实数）两个参数的过程（但实际上只需要求w，求得以后找某些样本点代入就可以求得b）。因此在求g(x)的时候，w才是变量。

你肯定能看出来，一旦求出了w（也就求出了b），那么中间的直线H就知道了（因为它就是wx+b=0嘛，哈哈），那么H1和H2也就知道了（因为三者是平行的，而且相隔的距离还是||w||决定的）。那么w是谁决定的？显然是你给的样本决定的，一旦你在空间中给出了那些个样本点，三条直线的位置实际上就唯一确定了（因为我们求的是最优的那三条，当然是唯一的），我们解优化问题的过程也只不过是把这个确定了的东西算出来而已。

样本确定了w，用数学的语言描述，就是w可以表示为样本的某种组合：

$$w = \alpha_1 x_1 + \alpha_2 x_2 + \ldots + \alpha_n x_n$$

同时，w不仅跟样本点的位置有关，还跟样本的类别有关，所以还应该加上类别信息：

$$w = \sum_{i=1}^{n} (\alpha_i y_i x_i)$$

由上式可以得到，g(x)可以改写成:

$$g(x) =< w, x > +b =< \sum_{i=1}^{n} (\alpha_i y_i x_i), x > +b$$

注意式子中x才是变量，也就是你要分类哪篇文档，就把该文档的向量表示代入到 x的位置，而所有的x_i统统都是已知的样本。还注意到式子中只有x_i和x是向量，因此一部分可以从内积符号中拿出来，得到$g(x)$的式子为：

$$g(x) = \sum_{i=1}^{n} \alpha_i y_i < x_i, x > +b \qquad (2)$$

至此，w就没了，(2)式也就是我们说的分类器的公式，以这样的形式描述问题以后，我们的优化问题少了很大一部分不等式约束。但是接下来先跳过线性分类器求解的部分，来看看 SVM在线性分类器上所做的重大改进——核函数。

核函数

那假设w'和x'是高维向量，之前的分类面的公式就可以写成：

$$f(x') =< w', x' > +b$$

它能够将原问题变得可分。式中的 w'和x'都是2000维的向量，只不过w'是定值，而x'是变量（好吧,严格说来这个函数是2001维的,哈哈），现在我们的输入呢，是一个1000维的向量x，分类的过程是先把x变换为2000维的向量x'，然后求这个变换后的向量x'与向量w'的内积，再把这个内积的值和b相加，就得到了结果，看结果大于阈值还是小于阈值就得到了分类结果。

我们其实只关心那个高维空间里内积的值，那个值算出来了，分类结果就算出来了。这让我们幻想，是否能有这样一种函数K(w,x),他接受低维空间的输入值，却能算出高维空间的内积值<w',x'>?

如果有这样的函数，那么当给了一个低维空间的输入x以后，

g(x)=K(w,x)+b

f(x')=<w',x'>+b

这两个函数的计算结果就完全一样，我们也就用不着费力找那个映射关系，直接拿低维的输入往g(x)里面代就可以了。

那这种函数就是我们所说的核函数。核函数的基本作用就是接受两个低维空间里的向量，能够计算出经过某个变换后在高维空间里的向量内积值。几个比较常用的核函数：

(1) 线性核函数:$K(x, x_I) = x \cdot x_i$

线性核，主要用于线性可分的情况，我们可以看到特征空间到输入空间的维度是一样的，其参数少速度快，对于线性可分数据，其分类效果很理想，因此我们通常首先尝试用线性核函数来做分类，看看效果如何，如果不行再换别的

(2) 多项式核函数：$K(x, x_i) = (x \cdot x_i) + 1)^d$

多项式核函数可以实现将低维的输入空间映射到高纬的特征空间，但是多项式核函数的参数多，当多项式的阶数比较高的时候，核矩阵的元素值将趋于无穷大或者无穷小，计算复杂度会大到无法计算。

(3) 高斯(RBF)核函数:$K(x, x_i) = exp(-\frac{||x-x_i||^2}{\delta^2})$

高斯径向基函数是一种局部性强的核函数，其可以将一个样本映射到一个更高维的空间内，该核函数是应用最广的一个，无论大样本还是小样本都有比较好的性能，而且其相对于多项式核函数参数要少，因此大多数情况下在不知道用什么核函数的时候，优先使用高斯核函数。

(4) sigmoid核函数:$K(x, x_i) = tanh(\eta < x, x_i > +\theta)$

采用sigmoid核函数，支持向量机实现的就是一种多层神经网络。 因此，在选用核函数的时候，如果我们对我们的数据有一定的先验知识，就利用先验来选择符合数据分布的核函数；如果不知道的话，通常使用交叉验证的方法，来试用不同的核函数，误差最下的即为效果最好的核函数，或者也可以将多个核函数结合起来，形成混合核函数。在吴恩达的课上，也曾经给出过一系列的选择核函数的方法：

如果特征的数量大到和样本数量差不多，则选用LR或者线性核的SVM；

如果特征的数量小，样本的数量正常，则选用SVM+高斯核函数；

如果特征的数量小，而样本的·`数量很大，则需要手工添加一些特征从而变成第一种情况。

回到刚才的线性分类器，它的公式是：

$$f(x') = \sum_{i=1}^{n} \alpha_i y_i < x_i', x' > +b$$

现在这个就是高维空间里的线性函数（为了区别低维和高维空间里的函数和向量，我改了函数的名字，并且给w和x都加上了'），我们就可以用一个低维空间里的函数（再一次的，这个低维空间里的函数就不再是线性的啦）来代替，

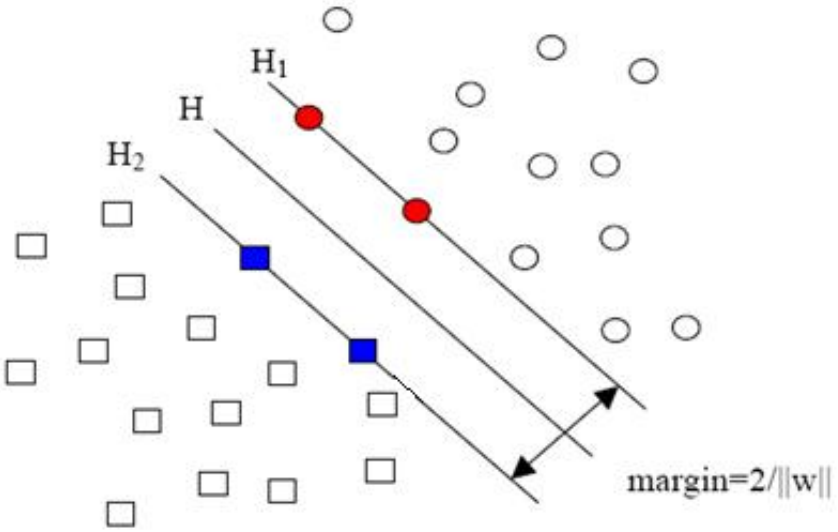
$$g(x) = \sum_{i=1}^{n} \alpha_i y_i K(x_i, x) + b$$

又发现什么了？f(x') 和g(x)里的α，y，b全都是一样一样的！这就是说，尽管给的问题是线性不可分的，但是我们就硬当它是线性问题来求解，只不过求解过程中，凡是要求内积的时候就用你选定的核函数来算。这样求出来的α再和你选定的核函数一组合，就得到分类器啦！

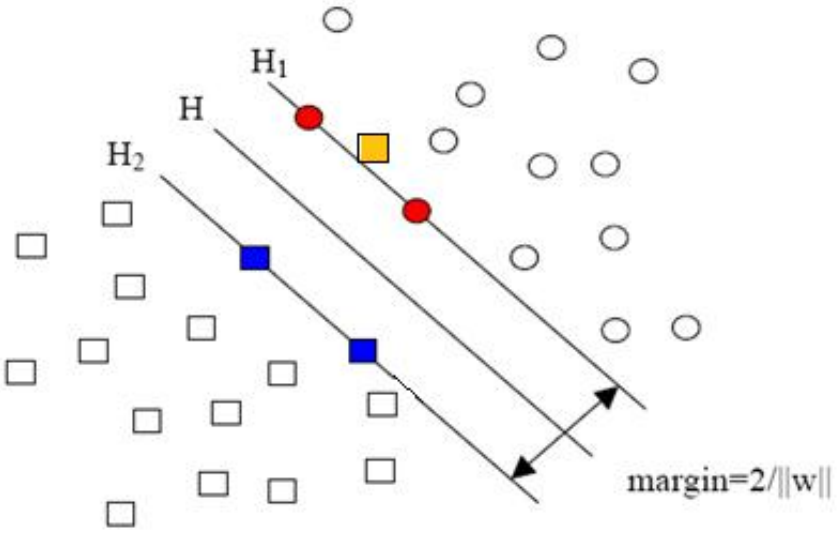
那如果已经用了核函数，还是线性不可分怎么办呢？那就有了下一节。

松弛变量

现在我们已经把一个本来线性不可分的文本分类问题，通过映射到高维空间而变成了线性可分的。就像下图这样：



圆形和方形的点各有成千上万个（毕竟，这就是我们训练集中文档的数量嘛，当然很大了）。现在想象我们有另一个训练集，只比原先这个训练集多了一篇文章，映射到高维空间以后（当然，也使用了相同的核函数），也就多了一个样本点，但是这个样本的位置是这样的：



就是图中黄色那个点，它是方形的，因而它是负类的一个样本，这单独的一个样本，使得原本线性可分的问题变成了线性不可分的。这样类似的问题（仅有少数点线性不可分）叫做“近似线性可分”的问题。

以我们人类的常识来判断，说有一万个点都符合某种规律（因而线性可分），有一个点不符合，那这一个点是否就代表了分类规则中我们没有考虑到的方面呢（因而规则应该为它而做出修改）？

其实我们会觉得，更有可能的是，这个样本点压根就是错误，是噪声，是提供训练集的同学人工分类时一打瞌睡错放进去的。所以我们会简单的忽略这个样本点，仍然使用原来的分类器，其效果丝毫不受影响。

因此如果要免去这些特殊点的作用，我们就要用间隔（而不是几何间隔）来衡量有利于我们表达形式的简洁。我们原先对样本点的要求是：

$$y_i[(wx_i)+b] \geq 1(i=1,2,\ldots I)(I\text{是样本数})$$

意思是说离分类面最近的样本点函数间隔也要比1大。如果要引入容错性，就给1这个硬性的阈值加一个松弛变量，即允许

$$y_i[(wx_i)+b] \geq 1-\xi_i(i=1,2,\ldots I)(I\text{是样本数})$$

$$\xi_i \geq 0$$

因为松弛变量是非负的，因此最终的结果是要求间隔可以比1小。但是当某些点出现这种间隔比1小的情况时（这些点也叫离群点），意味着我们放弃了对这些点的精确分类，而这对我们的分类器来说是种损失。但是放弃这些点也带来了好处，那就是使分类面不必向这些点的方向移动，因而可以得到更大的几何间隔（在低维空间看来，分类边界也更平滑）。显然我们必须权衡这种损失和好处。好处很明显，我们得到的分类间隔越大，好处就越多。回顾我们原始的硬间隔分类对应的优化问题：

$min \quad f(x)$

$subject\ to\ y_i[(wx_i)+b]-1 \geq 0(i=1,2,\ldots I)(I\text{是样本数})$

那我们既然加入了松弛变量，允许一部分值不符合约束条件，就会带来目标函数的值变大，衡量损失的方式有两种$\sum_{i=1}^{l}\xi^2$和$\sum_{i=1}^{l}\xi$。把损失加入到目标函数里的时候，就需要一个惩罚因子（cost，也就是libSVM的诸多参数中的C），原来的优化问题就变成了下面这样：

$min \quad f(x)+C\sum_{i=1}^{l}\xi_i$

$subject\ to\ y_i[(wx_i)+b] \geq 1-\xi_i(i=1,2,\ldots I)(I\text{是样本数})$

这个式子有这么几点要注意：

一是并非所有的样本点都有一个松弛变量与其对应。实际上只有“离群点”才有，或者也可以这么看，所有没离群的点松弛变量都等于0.

二是松弛变量的值实际上标示出了对应的点到底离群有多远，值越大，点就越远。

三是惩罚因子C决定了你有多重视离群点带来的损失，显然当所有离群点的松弛变量的和一定时，你定的C越大，对目标函数的损失也越大，此时就暗示着你非常不愿意放弃这些离群点，最极端的情况是你把C定为无限大，这样只要稍有一个点离群，目标函数的值马上变成无限大，马上让问题变成无解，这就退化成了硬间隔问题。

四是惩罚因子C不是一个变量，整个优化问题在解的时候，C是一个你必须事先指定的值，指定这个值以后，解一下，得到一个分类器，然后用测试数据看看结果怎么样，如果不够好，换一个C的值，再解一次优化问题，得到另一个分类器，再看看效果，如此就是一个参数寻优的过程，但这和优化问题本身决不是一回事，优化问题在解的过程中，**C一直是定值**，要记住。

五是尽管加了松弛变量这么一说，但这个优化问题仍然是一个优化问题（汗，这不废话么），解它的过程比起原始的硬间隔问题来说，没有任何更加特殊的地方。

其实两者还有微妙的不同。一般的过程应该是这样，还以文本分类为例。在原始的低维空间中，样本相当的不可分，无论你怎么找分类平面，总会有大量的离群点，此时用核函数向高维空间映射一下，虽然结果仍然是不可分的，但比原始空间里的要更加接近线性可分的状态（就是达到了近似线性可分的状态），此时再用松弛变量处理那些少数“冥顽不化”的离群点，就简单有效得多啦。

结语

SVM最精彩的地方在开头已经说过了，在于它能解决线性不可分的数据分类，然后本文也详细地介绍了SVM的优化目标、求解方向、核函数和松弛变量。