**6.支持向量机**

**6.1定义**

支持向量机的基本模型是定义在特征空间上的间隔最大的线性分类器，是一种二分类模型。

支持向量机从繁至简可以分为三种模型：1）**线性可分支持向量机**（linear support vector machine in linearly separable case，又称为硬间隔支持向量机），当数据线性可分时通过硬间隔最大化（hard margin maximization）学习一个线性分类器；2）**线性支持向量机**（又称为软间隔支持向量机），当数据近似线性可分时通过软间隔最大化（soft margin maximization）学习一个线性分类器；3）**非线性支持向量机**，当训练数据线性不可分时通过使用核技巧（kernel trick）及软间隔最大化学习非线性分类器（其实就是升维到高维空间使数据线性可分再隐式地在高维空间学习线性分类器）。

支持向量机假设**输入空间和特征空间是两个不同的空间**，**学习是在特征空间进行的**。输入空间为欧氏空间或离散空间，特征空间为欧氏空间或希伯特空间，线性可分支持向量机、线性支持向量机假设这两个空间的元素一一对应，并将输入空间中的输入映射为特征空间中的特征向量；非线性支持向量机则是通过一个输入空间到特征空间的非线性映射将输入映射为特征向量。

这个非线性映射就是核技巧，核函数（kernel function）表示将输入从输入空间映射到特征空间得到特征向量之间的内积。核方法（kernel method）是比支持向量机更为一般的机器学习方法。

支持向量机学习的基本思路是：在特征空间中找到一个分离超平面，能够将训练数据集正确分类并且几何间隔最大。几何间隔最大意味着以充分大的确信度对训练数据进行分类，不仅将正负实例点分开，而且对最难分的实例点（离超平面最近的点也就是支持向量）也有足够大的确信度将它们分开。同时间隔最大化求得的分离超平面解是唯一的。这不同于感知机，感知机的学习用误分类最小的策略求得的超平面解是有无穷多个的。

支持向量机的学习策略间隔最大化，可形式化为一个求解凸二次规划（convex quadratic programming）的问题，也等价于正则化的合页损失函数的最小化问题。支持向量机的学习算法是求解凸二次规划的最优化算法。

优点：泛化错误率低，计算开销不大，结果易解释

缺点：对参数调节和核函数选择敏感，原始分类器不加修改仅适用于处理二分类问题。

**6.2线性可分支持向量机**

给定线性可分训练数据集，通过间隔最大化或等价地求解相应的凸二次规划问题学习得到的分离超平面w\*·x + b\* = 0以及相应的决策分类函数f(x) = sign(w\*·x + b\*)称为线性可分支持向量机。

**6.2.1函数间隔和几何间隔**

**函数间隔**：给定的训练数据集T和超平面(w，b)，定义超平面(w，b)关于样本点(xi，yi)的函数间隔为

定义超平面(w，b)关于训练数据集T的函数间隔为超平面(w，b)关于T中所有样本点(xi，yi)的函数间隔的最小值

函数间隔可以表示分类预测的正确性及确信度。函数间隔对超平面(w，b)的取值敏感，将w和b相同倍数的缩放，函数间隔大小也相同倍数的缩放，但是超平面却并没有改变。所以我们要对法向量w加些约束，将函数间隔变成几何间隔。

**几何间隔**：给定的训练数据集T和超平面(w，b)，定义超平面(w，b)关于样本点(xi，yi)的几何间隔为

定义超平面(w，b)关于训练数据集T的几何间隔为超平面(w，b)关于T中所有样本点(xi，yi)的几何间隔的最小值

几何间隔一般是实例点到超平面带符号的距离（signed distance），当样本点被正确分类时几何间隔就是实例点到超平面的距离。由函数间隔和几何间隔的定义易知，几何间隔和函数间隔的关系为

**6.2.2间隔最大化学习**

支持向量机的学习问题就是凸优化问题即约束最优化问题

其中目标函数f(w)和约束函数gi(w)都是Rn上的连续可微凸函数，约束函数hi(w)是Rn上的仿射函数。目标函数是二次函数且约束函数gi(w)是仿射函数时问题就是凸二次规划问题。

和感知机一样，支持向量机的学习也可以分为原始问题和对偶问题。支持向量机学习时希望最大化超平面(w，b)关于训练数据集的几何间隔γ，即建立约束条件表示超平面(w，b)关于每个训练实例点几个间隔至少是γ：

根据几何间隔和函数间隔的关系，可将条件改写为

函数间隔的取值并不影响最优化问题的解。令带入上面的最优化问题，并将分子中的||w||去除，就将线性可分支持向量机学习问题变成

**线性可分支持向量机学习算法——最大间隔法：**

输入：线性可分训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn，yi∈{+1，-1}，i = 1，2，…，N）；

输出：最大间隔分离超平面和分类决策函数。

（1）构造并求解约束最优化问题

求得最优解w\*，b\*；

（2）由此得到分离超平面：

w\*·x + b\* = 0

分类决策函数

f (x) = sign(w\*·x + b\*)

在线性可分条件下，**训练数据集的样本点中与分离超平面距离最近的样本点称为支持向量（support vector）**，即支持向量是使约束条件yi(w·xi + b) – 1 = 0成立的点。支持向量的个数一般很少，但是在决定分离超平面时只有支持向量起作用，其他实例点不起作用，移动支持向量将改变所求的解，移动其他实例点甚至去掉这些点解都不会改变，所以这种分类模型才称为支持向量机。

支持向量都落在间隔边界上，对yi = +1的正例点支持向量在间隔边界H1：w·x + b = 1上，对对yi = -1的负例点支持向量在间隔边界H2：w·x + b = -1上。H1和H2平行，并且没有实例点落在它们终极，分离超平面与它们平行且位于它们的中央。H1和H2之间的距离称为间隔（margin），间隔依赖于分离超平面的法向量w，等于。

与感知机一样，应用拉格朗日对偶性通过求解对偶问题的都原始问题的最优解。对偶算法有两个优点：1）对偶问题往往更容易求解；2）可以自然地引入核函数，进而推广到非线性分类问题。

对原始问题的约束条件引入拉格朗日乘子（Lagrange multiplier）αi ≥ 0（i = 1，2，…，N）定义拉格朗日函数：

其中α = (α1，α2，…，αN)T为拉格朗日乘子向量。于是原始问题的求解就变成了对偶问题的极大极小问题：

先求L(w，b，α)对w，b的极小，再求对α的极大。

a）求

分别求L(w，b，α)对w，b的的偏导数并令其等于0：

可得

将以上两式带入拉格朗日函数

b）然后求对α的极大，对L(w，b，α)取负由极大转为极小

求得以上问题的解α\*可以求得原始问题的解w\*，b\*。根据前面拉格朗日函数L(w，b，α)求偏导这一步可知从α\*如何求得w\*的解

至于b\*的解（根据拉格朗日对偶性定理C.3），α\*中至少有一个αj\* > 0满足

并且yj2 = 1，可知

**线性支持向量机对偶算法：**

输入：线性可分训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn，yi∈{+1，-1}，i = 1，2，…，N）；

输出：分离超平面和分类决策函数。

（1）构造并求解约束最优化问题

求得最优解α\* = (α1\*，α2\*，…，αN\*)T。

（2）根据α\*求解

并选择α\*的一个正分量αj\* > 0，求解

（3）由此得到分离超平面：

w\*·x + b\* = 0

分类决策函数

f (x) = sign(w\*·x + b\*)

由w\*，b\*的求解公式可知，w\*和b\*只依赖于训练数据集中对应于αi\* > 0的样本点(xi, yi)，而与其他样本点无关，这些对应于αi\* > 0的样本点就是支持向量。

**6.3线性支持向量机**

**6.3.1学习算法**

线性可分的学习方法不适用于线性不可分训练数据，线性可分方法中的不等式约束在线性不可分训练数据中并不能都成立，因为在线性不可分训练数据中，有一些特异点（outlier），将这些特异点去除后，剩下的大部分样本点组成的集合才是线性可分的。

为了解决线性不可分数据中某些样本点(xi, yi)不能满足函数间隔大于等于1的约束条件，可以对每个样本点(xi, yi)引进一个松弛变量ξi ≥ 0，使函数间隔加上松弛变量等于等于1，于是约束条件就变为

同时对每个松弛变量ξi支付一个代价ξi，目标函数就变成

这个目标函数包含两层含义：a）使尽量小即间隔尽量大；b）使误分类点的个数尽量少。其中C > 0是调和二者的系数，称为惩罚参数，决定了惩罚的大小，一般由应用问题决定，C值增大时对误分类的惩罚增大，C值变小时对误分类的惩罚减小。

由此硬间隔最大化转化为软间隔最大化。线性不可分的线性支持向量机学习问题（原始问题）就变成了如下凸二次规划问题

可以证明w的解是唯一的，但是b的解存在于一个区间，并不是唯一的。通过求解以上的凸二次规划问题（软间隔最大化问题）得到的分离超平面w\*·x + b\* = 0以及相应的决策分类函数f(x) = sign(w\*·x + b\*)称为线性支持向量机。

线性支持向量机的原始最优化问题的拉格朗日函数是

其中，αi ≥ 0，μi ≥ 0。对偶问题就是拉格朗日函数的极大极小问题，求L(w，b，ξ，α，μ)对w，b，ξ的极小

可得

将上式代入拉格朗日函数可得

因此原始问题可以转化为对偶问题

可以看到，线性不可分的对偶算法约束条件相比线性可分的对偶算法约束条件只是多了αi ≤ C这样一个限制。惩罚参数C是一个需要慎重选择的超参数！

**线性支持向量机对偶学习算法：**

输入：训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn，yi∈{+1，-1}，i = 1，2，…，N）；

输出：分离超平面和分类决策函数。

（1）选择惩罚参数C > 0，构造并求解凸二次规划问题

求得最优解α\* = (α1\*，α2\*，…，αN\*)T。

（2）计算

选择α\*的一个分量αj\*适合条件0 < αj\* < C，计算

（3）求得分离超平面：

w\*·x + b\* = 0

分类决策函数

f (x) = sign(w\*·x + b\*)

软间隔的支持向量比线性可分时要复杂一些，若α\* < C，则ξi = 0，支持向量xi恰好落在间隔边界上；若α\* = C，则0 < ξi < 1，则分类正确，xi在间隔边界与分离超平面之间；若α\* = C，ξi = 1，则xi在分离超平面上；α\* = C，ξi > 1，则xi在分离超平面误分一侧。

**6.3.2合页损失函数**

线性支持向量机的原始最优化问题还等价于最小化以下目标函数

目标函数的第一项是经验损失（经验风险），函数

称为合页损失函数（hinge loss function）。下标“+”表示以下取正值的函数：

从上列函数可知，当样本点(xi, yi)被正确分类且函数间隔（确信度）yi(w·xi + b)大于1时，损失是0，否则损失是1- yi(w·xi + b)。目标函数的第二项是系数为λ的w的L2范数，是正则化项。

其实，合页损失函数的损失可以认为是原始问题引入的松弛变量的大小，因为1-yi (w·xi + b) > 0时，有1-yi (w·xi + b) = 1-ξi；当1-yi (w·xi + b) ≤ 0时，ξi = 0，有yi (w·xi + b) ≥ 1-ξi都成立。所以最优化问题可以写成

若令λ=，则原式变成

与原始问题等价。所以原始最优化问题就等价于最优化合页损失函数。

合页损失函数可以认为是0-1损失函数的上界，所以可以认为线性支持向量机是优化由0-1损失函数的上界（合页损失函数）构成的目标函数，这时的上界损失函数又称为代理损失函数（surrogate loss function）。合页损失函数被认为不仅要正确分类，而且要确信度足够高时损失才是0，对学习有更高的要求。

**6.4非线性支持向量机**

给定训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn属于输入空间，对应两类类标记yi∈{+1，-1}，i = 1，2，…，N），如果能用Rn中的一个超曲面将正负例正确分开，则称这个问题为非线性可分问题。

非线性问题往往转换为线性问题求解。一般分为两步：a）使用一个变换将原空间的数据映射到新空间；b）在新空间里用线性分类学习方法从训练数据中学习分类模型。核技巧就是这样的方法。

**6.4.1核函数**

核技巧就是学习是隐式地在特征空间进行的，不需要显式地定义特征空间和映射函数。

假设χ是输入空间（欧氏空间Rn的子集或者离散集合），又设Н为特征空间（希尔伯特空间），如果存在一个从χ到Н的映射

使得对所有x，z∈χ，函数K(x，z)满足条件

则称K(x，z)为核函数，(x)为映射函数（(x)·(x)为(x)和(z)的内积）。特征空间Н一般是高维的甚至无穷维的，且特征空间Н和映射函数取法并不唯一，可以取不同的特征空间，在同一特征空间也可以取不同的映射。

核函数就是在输入空间中将两个实例xi，xj的内积变换为特征空间中对应点的内积(xi)·(xj)，直接计算核函数往往比较容易，这样就不需要显式地定义映射函数进行转换。当映射函数是非线性时，学习到含有核函数的支持向量机也是非线性分类模型。在支持向量机的对偶问题中，无论目标函数还是决策函数（分离超平面）都只涉及输入实例和实例间的内积，用核函数K(xi，xj) = (xi)·(xj)替代内积xi·xj就得到非线性目标函数

此时分类决策函数中的内积也用核函数替代

**常用核函数：**

（1）多项式核函数（polynomial kernel function）

K(x，z) = (x·z + 1)p

对应的支持向量机是一个p次多项式分类器。此时分类决策函数为

（2）高斯核函数（Gaussian kernel function）

对应的支持向量机是高斯径向基函数（radial basis function）分类器，此时分类决策函数为

（3）字符串核函数（string kernel function）

字符串核函数是定义在离散数据集合的核函数，在文本分类、信息检索、生物信息学等方面都有应用。

假设S是长度大于等于n的字符串集合，s是S的元素，字符串集合S到特征空间Hn=的映射(s)，表示定义在∑n（所有长度为n的字符串集合记作∑n）上的实数空间。其每一维对应一个字符串u∈∑n，映射(s)将字符串s对应于空间的一个向量，其在u维上的取值为

其中0 < λ ≤ 1是一个衰减参数，l(i)表示字符串i的长度，求和在s中所有与u相同的字串上进行。

两个字符串s和t上的字符串核函数是基于映射的特征空间中的内积：

字符串核函数kn(s，t)给出了字符串s和t中长度等于n的所有子串组成的特征向量的余弦相似度。直观上两个字符串相同的子串越多它们就越相似，字符串核函数的值就越大。字符串核函数可以用动态规划快速计算。

通常所说的核函数一般是正定核，正定核K(x，z)对应的Gram矩阵K=[K(xi，xj)]m\*m是半正定矩阵。

**6.4.2非线性学习算法**

输入：训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn，yi∈{+1，-1}，i = 1，2，…，N）；

输出：分类决策函数。

（1）选取适当的核函数K(x，z)和适当的参数C，构造并求解最优化问题

求得最优解α\* = (α1\*，α2\*，…，αN\*)T。

（2）选择α\*的一个分量αj\*适合条件0 < αj\* < C，计算

（3）构造决策函数：

当K(x，z)是正定核函数时，凸二次规划问题的解是存在的。

**6.5序列最小最优化算法**

支持向量机的学习问题可以形式化为求解凸二次规划问题，这样的凸二次规划问题具有全局最优解。当训练样本容量很大时，序列最小优化（sequential minimal optimization，SMO）就是其中一种高效实现支持向量机学习问题的算法。

SMO算法是不断地将原二次规划问题分解为只有两个变量的二次规划子问题，并对子问题进行解析求解，直到所有变量满足KKT条件为止。子问题的两个变量，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。整个算法包括两个部分：1）求解两个变量二次规划的解析方法；2）选择变量的启发式方法。

**6.5.1求解两个变量二次规划的解析方法**

假设选择的两个变量α1，α2，其他变量αi（i = 3，4，…，N）固定。于是SMO的最优化问题的子问题可以写成

其中Kij= K(x，z)，i，j=1，2，…，N，ζ是常数，目标函数中省略了不含α1，α2的常数项。

因为两个α1，α2满足某些约束条件，我们可以将两个变量的约束问题变成单变量的最优化问题，比如先求α2的解。将以下函数记作

再引进记号

于是目标函数可以写成

因为且，所以可将α1表示为

将上式代入目标函数

然后对α2求导

令偏导数为0，得到

将函数g(x)对输入xi的预测值与真实输出yi之差记作

同时将约束代入目标函数求偏导的式子可得

令，代入上式可得未剪辑的新解

很容易证明，其中，(x)是输入空间到特征空间的映射。

而这只是未经剪辑的解，因为这个解要使其满足不等式，必须将其限制在某个区间[L，H]内。两个变量的约束可以由以下二维空间图形表示：

α2 = C

α1 = 0 α1 = C

α2 = 0

y1≠y2→α1-α2 = k y1=y2→α1+α2 = k

因为不等式约束0 ≤ α1，α2 ≤ C，使得（α1，α2）都在正方形[0，C]\*[0，C]内，同时又满足等式α1y1+α2y2 = ζ，因为y1，y2∈{-1，+1}，所以（α1，α2）在平行于正方形的对角线的直线上，目标函数在一条平行于对角线的线段上的最优解。

如果y1≠y2，则

如果y1=y2，则

所以经过剪辑后α2的解是

因为和满足下列等式

由上述等式可求得是

**6.5.2变量的选择方法**

SMO算法在每个子问题中选择两个变量优化，其中至少一个变量是违反KKT条件的。

（1）SMO选择第一个变量的过程为外层循环，外层循环在训练样本中选取违反KKT条件最严重的样本点。KKT条件即

其中。

该检验是在ε范围内进行的，可以分为两步：

a）首先遍历所有在间隔边界上的支持向量点（即满足条件），检验它们是否满足KKT条件；

b）如果这些支持向量点都满足KKT条件，则遍历整个训练集，检验它们是否满足KKT条件。

（2）SMO选择第二个变量的过程称为内层循环，内层循环的标准是能使α2有足够大的变化。α2的选择也可以分为两步：

a）简单做法，选择的α2使|E1-E2|最大。因为此时α1已定，E1也确定了。如果E1是正的，那么选择最小的Ei作为E2；如果E1是负的，则选择最大的的Ei作为E2。可以将所有的Ei保存在一个列表中以节省计算时间。

b）特殊情况下，上面的简单方法选择的α2不能使目标函数有足够的下降。那么就采用启发式规则继续选择α2：a）遍历间隔边界上的支持向量点作为候选α2，直到目标函数有足够的下降；b）若依然找不到合适的α2，则遍历训练数据集。c）若还是找不到合适的α2，则放弃现在的α1，再进入外层循环选择新的α1。

（3）每次完成两个变量的优化后，都要重新计算阈值b和差值Ei。由KKT条件可知，当时

于是有

由Ei的定义式可得E1的等式

所以的前两项可以写成

代入可得

同理，如果，那么

的取值可以分为两种情况：

a）如果、同时满足条件（i=1，2），那么=。

b）如果、是0或C，那么和以及它们之间的数都是符合KKT条件的阈值，这时选择它们中点作为bnew。

每次完成两个变量优化之后，还必须更新对应的Ei值并保存在列表中：

其中i = 1，2，S是所有支持向量xj的集合。

**6.5.3 SMO算法**

输入：训练数据集T = {(x1, y1), (x2, y2), ......, (xN, yN)}（xi∈Rn，yi∈{+1，-1}，i = 1，2，…，N）；精度ε；

输出：近似解。

（1）取初值α(0) = 0，令k = 0；

（2）选取优化变量α1(k)，α2(k)，解析求解两个变量的优化问题，求得最优解α1(k+1)，α2(k+1)，更新α为α(k+1)；

（3）若在精度ε范围内满足停机条件

其中，则转（4）否则转（2）；

（4）取= α(k+1)。