机器学习

第四章 支持向量机

鲍鹏
北京交通大学

目录

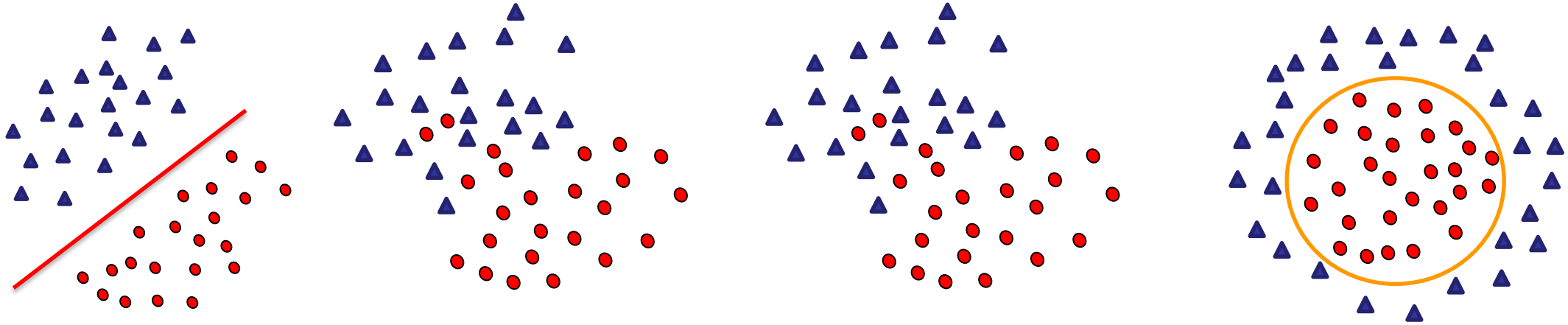
目录

支持向量机相关概念

- 支持向量机（support vector machines，SVM）是一种二类分类模型。
- 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，间隔最大使它有别于感知机；支持向量机还包括核技巧，这使它成为实质上的非线性分类器。
- 支持向量机的学习策略就是间隔最大化，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。

1.支持向量机概述

硬间隔、软间隔和非线性 SVM



线性可分　　硬间隔　　软间隔　　线性不可分

假如数据是完全的线性可分的，那么学习到的模型可以称为硬间隔支持向量机。换个说法，硬间隔指的就是完全分类准确，不能存在分类错误的情况。软间隔，就是允许一定量的样本分类错误。

支持向量机分类

- **线性可分支持向量机**(linear support vector machine in linearly separable case).
  - 当训练数据线性可分时，通过硬间隔最大化(hard margin maximization)，学习一个线性的分类器，即线性可分支持向量机，又称为硬间隔支持向量机；

- **线性支持向量机**(linear supportvector machine)
  - 当训练数据近似线性可分时，通过软间隔最大化(soft margin maximization)，学习一个线性的分类器，即线性支持向量机，又称为软间隔支持向量机；

- **非线性支持向量机**(non-linear support vector machine)
  - 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，学习非线性支持向量机。

支持向量机的优点

口优点：

- 有严格的数学理论支持，可解释性强，不依靠统计方法，从而简化了通常的分类和回归问题;能找出对任务至关重要的关键样本（支持向量);
- 采用映射到高维的解决方法之后，可以处理非线性分类/回归任务;
- 最终决策函数只由少数的支持向量所确定，计算的复杂性取决于支持向量的数目，而不是样本空间的维数，这在某种意义上避免了“维数灾难”。

支持向量机的缺点

口缺点：

- 训练时间长。当采用SMO算法时，由于每次都需要挑选一对参数，因此时间复杂度为O(n^2)其中n为训练样本的数量;
- 当采用核技巧时，如果需要存储核矩阵，则空间复杂度为O(n^2);
- 模型预测时，预测时间与支持向量的个数成正比。当支持向量的数量较大时，预测计算复杂度较高。因此支持向量机只适合小批量样本的任务，无法适应百万甚至上亿样本的任务。

核函数与核技巧

- 当输入空间为欧式空间或离散集合、特征空间为希尔伯特空间时，核函数（kernel function）表示将输入从输入空间映射到特征空间得到的特征向量之间的内积。

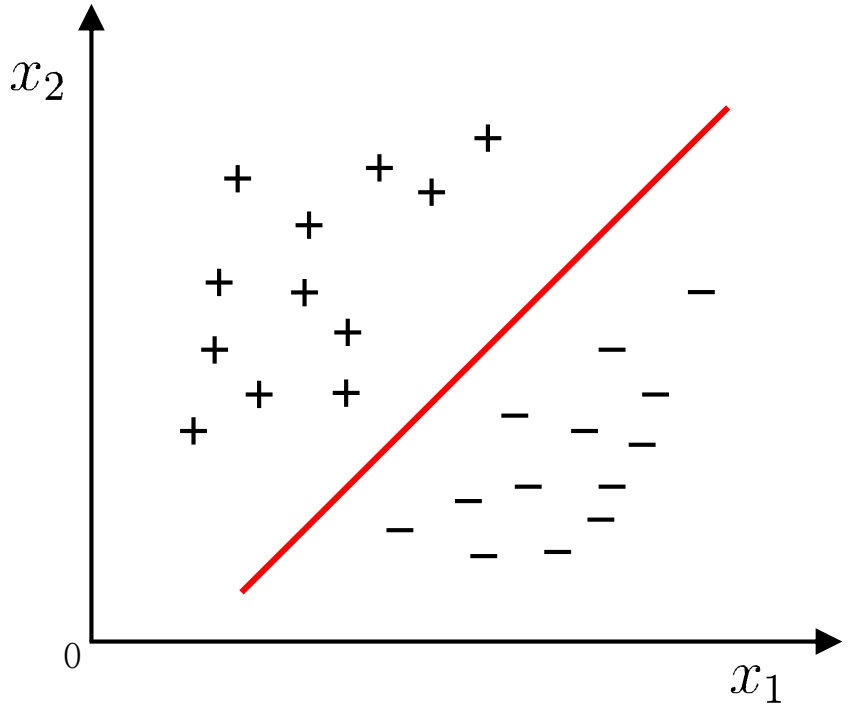
- 通过使用核函数可以学习非线性支持向量机，等价于隐式地在高维的特征空间中学习线性支持向量机。这样的方法称为核技巧。核方法（kernel method）是比支持向量机更为一般的机器学习方法。

- Cortes与Vapnik提出线性支持向量机，Boser、Guyon与Vapnik又引入核技巧，提出非线性支持向量机。

目录

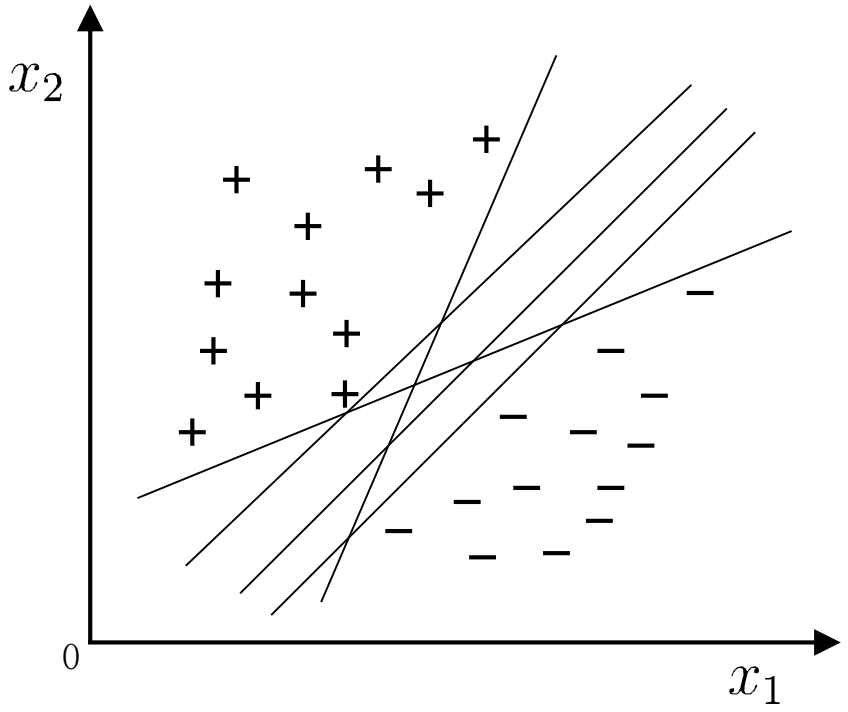
引例

线性模型：在样本空间中寻找一个超平面,将不同类别的样本分开。其中+代表正例，-代表负例。



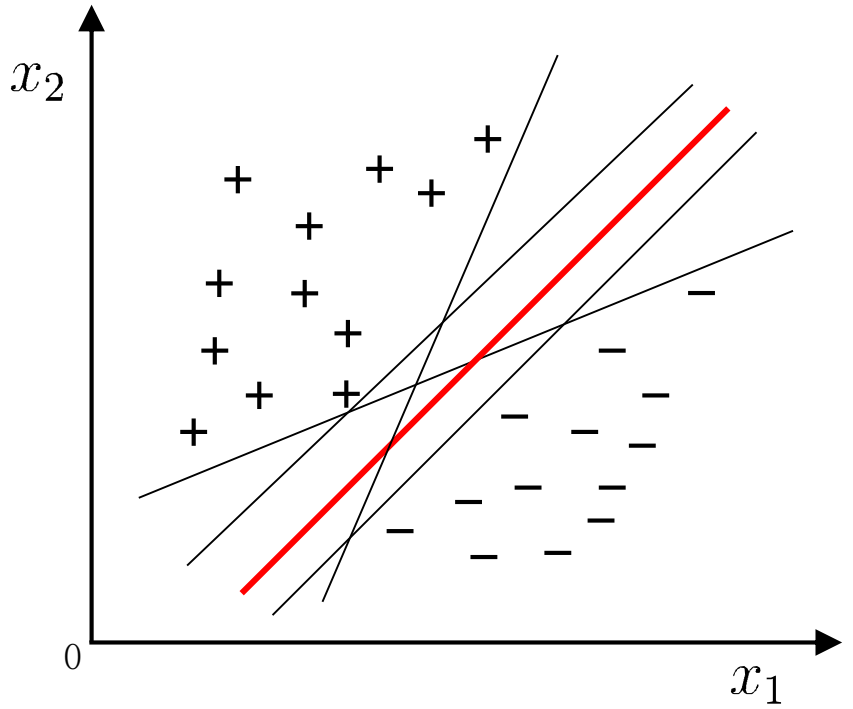
引例

-Q:将训练样本分开的超平面可能有很多, 哪一个好呢?



引例

-Q:将训练样本分开的超平面可能有很多，哪一个好呢?



-A:应选择”正中间”的超平面，容忍性最好，鲁棒性最高，泛化能力最强。

训练数据集线性可分，这时有许多直线能将两类数据正确划分。
线性可分支持向量机对应着将两类数据正确划分并且间隔最大的直线。

线性可分支持向量机

- 考虑一个二类分类问题。假设输入空间与特征空间为两个不同的空间。输入空间为欧氏空间或离散集合，特征空间为欧氏空间或希尔伯特空间。
- 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。
- 非线性支持向量机利用一个从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。
- 所以输入都由输入空间转化到特征空间，支持向量机的学习是在特征空间进行的。

线性可分支持向量机

假设给定一个特征空间上的训练数据集

$$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$$

其中，$x_i \in \mathcal{X}=\mathrm{R}^{\mathrm{n}}$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$\mathrm{i} = 1,2,\cdots,\mathrm{N}$。$x_i$为第$i$个特征向量,也成为实例, y_i为x_i的类标记,当$y_i = +1$时，称x_i为正例；当$y_i = -1$时，称x_i为负例，(x_i, y_i)称为样本点。再假设训练数据集是线性可分的。

- 学习的目标是在特征空间中找到一个分离超平面，能将实例分到不同的类。分离超平面对应于方程$w \cdot x + b = 0$，它由法向量w和截距b决定，可用(w, b)来表示。分离超平面将特征空间划分为两部分，一部分是正类，一部分是负类。法向量指向的一侧为正类，另一侧为负类。

线性可分支持向量机

- 当训练数据集线性可分时，存在无穷个分离超平面可将两类数据正确分开。
- 感知机利用误分类最小的策略，求得分离超平面，不过这时的解有无穷多个。
- 线性可分支持向量机利用间隔最大化求最优分离超平面，这时解是唯一的。

线性可分支持向量机定义

定义4.1（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为

$$w^* \cdot x + b^* = 0 \quad (4.1)$$

以及相应的分类决策函数

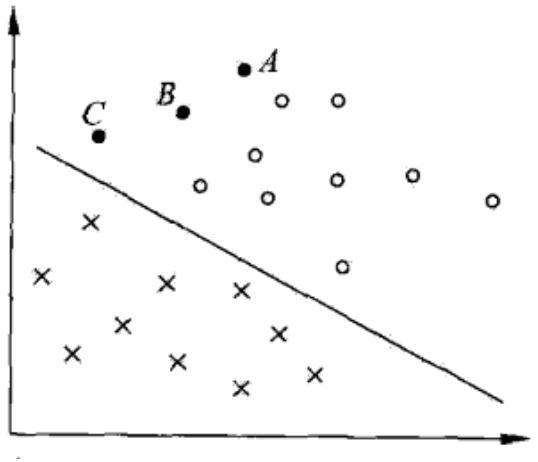
$$f(x) = sign(w^* \cdot x + b^*) \quad (4.2)$$

称为线性可分支持向量机。

函数间隔

- 如图，有A、B、C三个点，表示3个实例，均在分离超平面的正类一侧，预测它们的类。点A距分离超平面较远，若预测该点为正类，就比较确信预测是正确的；点C距分离超平面较近，若预测该点为正类就不那么确信；点B介于A与C之间，预测其为正类的确信度也在A与C之间。

- 点到分离超平面的远近：$|w \cdot x + b|$ ⇒ 表示分类预测的确信程度
- $w \cdot x + b$的符号与类标记y的符号是否一致 ⇒ 表示分类是否正确
- 所以可用$y(w \cdot x + b)$表示分类的正确性和确信度
- 这就是函数间隔的概念。



函数间隔

定义4.2（函数间隔）对于给定的训练数据集T和超平面（w,b），定义超平面（w,b）关于样本点（x_i,y_i）的函数间隔为

$$\hat{\gamma}_i = y_i(w \cdot x_i + b) \quad (4.3)$$

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（x_i,y_i）的函数间隔之最小值，即

$$\hat{\gamma} = \min_{i=i,\ldots,N} \hat{\gamma}_i \quad (4.4)$$

函数间隔

- 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时，只有函数间隔还不够。因为只要成比例地改变w和b，例如将它们改为2w和2b，超平面并没有改变。但函数间隔却成为原来的2倍。
- 这一事实启示我们，可以对分离超平面的法向量w加某些约束，如规范化，$\|w\| = 1$，使得，间隔是确定的。这时函数间隔成为几何间隔。

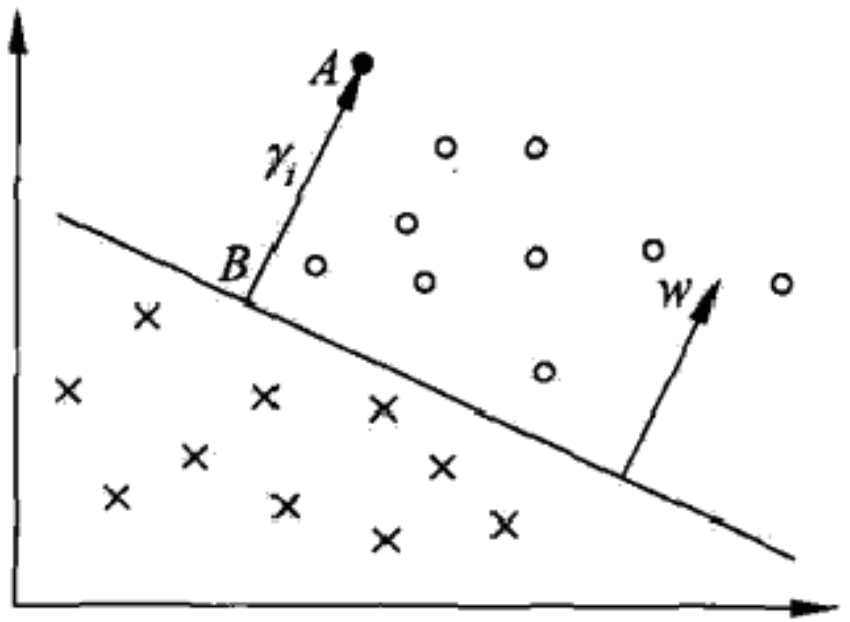
几何间隔

如图给出了超平面（w,b）及其法向量w。点A表示某一实例x_i,其类标记为$y_i=+1$。点A与超平面（w,b）的距离由线段AB给出，记作γ_i。

$$\gamma_i = (\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|})$$

其中，$\|w\|$为w的L_2范数。这时点A在超平面正的一侧的情形。如果点A在超平面负的一侧，即$y_i=-1$，那么点与超平面的距离为

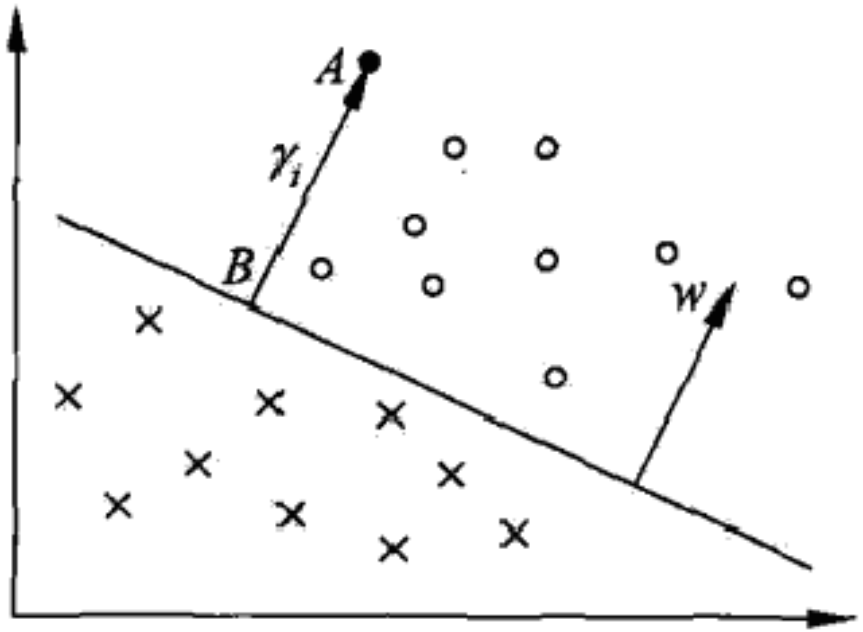
$$\gamma_i = -(\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|})$$



几何间隔

一般地，当样本点（x_i,y_i）被超平面（w,b）正确分类时，点x_i与超平面（w,b）的距离是

$$\gamma_i = y_i(\frac{w}{\|w\|} \cdot x_i + \frac{b}{\|w\|})$$



几何间隔

定义4.3（几何间隔）对于给定的训练数据集T和超平面（w,b），定义超平面（w,b）关于样本点（x_i,y_i）的函数间隔为

$$\gamma_i = y_i\left(\frac{w}{\|w\|}\cdot x_i + \frac{b}{\|w\|}\right) \quad (4.5)$$

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（x_i,y_i）的函数间隔之最小值，即

$$\gamma = \min_{i=1,\ldots,N} \gamma_i \quad (4.6)$$

超平面（w,b）关于样本点（x_i,y_i）的几何间隔一般是实例点到超平面的带符号的距离，当样本点被超平面正确分类时就是实例点到超平面的距离。

函数间隔与几何间隔

从函数间隔和几何间隔的定义（式(4.3)~式(4.6)）可知，函数间隔和几何间隔有下面的关系：

$$\gamma_i = \frac{\hat{\gamma}_i}{\|w\|} \quad (4.7)$$

$$\gamma = \frac{\hat{\gamma}_i}{\|w\|} \quad (4.8)$$

如果$\|w\| = 1$，那么函数间隔和几何间隔相等。如果超平面参数w和b成比例地改变（超平面没有改变），函数间隔也按此比例改变，而几何间隔不变。

间隔最大化

- 支持向量机学习的基本想法是求解能够正确划分训练数据集并且几何间隔最大的分离超平面。
- 对线性可分的训练数据集而言，线性可分分离超平面有无穷多个（等价于感知机），但是几何间隔最大的分离超平面是唯一的。这里的间隔最大化又称为硬间隔最大化。
- 间隔最大化的直观解释是：对训练数据集找到几何间隔最大的超平面意味着以充分大的确信度对训练数据集进行分类。也就是说，不仅将正负实例点分开，而且对最难分的实例点（离超平面最近的店）也有足够大的确信度将它们分开。这样的超平面应该对未知的新实例有很好的分类预测能力。

最大间隔分离超平面

- 求得最大间隔分类超平面可以表示为以下的约束最优化问题

$$\max_{w,b} \quad \gamma \qquad (4.9)$$

$$s.t \quad y_i\left(\frac{w}{\|w\|}\cdot x_i+\frac{b}{\|w\|}\right)\geq\gamma, \quad i=1,2,\dots,N \qquad (4.10)$$

- 根据几何间隔和函数间隔的关系式(4.8)，可以将上述问题改写为

$$\max_{w,b} \quad \frac{\hat{\gamma}}{\|w\|} \qquad (4.11)$$

$$s.t \quad y_i(w\cdot x_i+b)\geq\hat{\gamma}, \quad i=1,2,\dots,N \qquad (4.12)$$

最大间隔分离超平面

- 由于函数间隔$\hat{\gamma}$的改变对于上述问题没有影响，因此可以考虑取$\hat{\gamma}=1$

  则最大化$\frac{1}{\|w\|}$和最小化$\frac{1}{2}\|w\|^2$是等价的

- 得到下面的线性可分支持向量机学习的最优化问题（凸二次规划问题）：

$$\min_{w,b} \quad \frac{1}{2}\|w\|^2 \qquad (4.13)$$

$$s.t. \quad y_i(w\cdot x_i+b)-1\geq 0, i=1,2,\dots,N \qquad (4.14)$$

- 如果求出了约束最优化问题(4.13)~(4.14)最优解w^*,b^*，那么就可以得到最大间隔分离超平面$w^*\cdot x+b^*=0$及分类决策函数$f(x)=sign(w^*\cdot x+b^*)$，即线性可分支持向量机模型。

间隔最大化

- 凸优化问题是指约束最优化问题

$$\min_{w} \quad f(w) \quad (4.15)$$

$$s.t.\, g_i(w) \leq 0, i = 1,2,\ldots,k \quad (4.16)$$

$$h_i(w) = 0, i = 1,2,\ldots,l \quad (4.17)$$

当目标函数f(w)是二次函数且约束函数$g_i(w)$是仿射函数时，上述凸最优化问题成为凸二次规划问题。

最大间隔分离超平面

算法 1（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$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，其中，$x_i \in \mathcal{X} = \mathrm{R}^{\mathrm{n}}$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$\mathrm{i} = 1,2,\cdots,\mathrm{N}$；

输出：最大间隔分离超平面和分类决策函数。

最大间隔分离超平面

（1）构造并求解约束最优化问题：

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

$$s.t. \quad y_i(w \cdot x_i + b) - 1 \geq 0, \qquad i = 1,2,\cdots,N$$

求得最优解w^*，b^*。

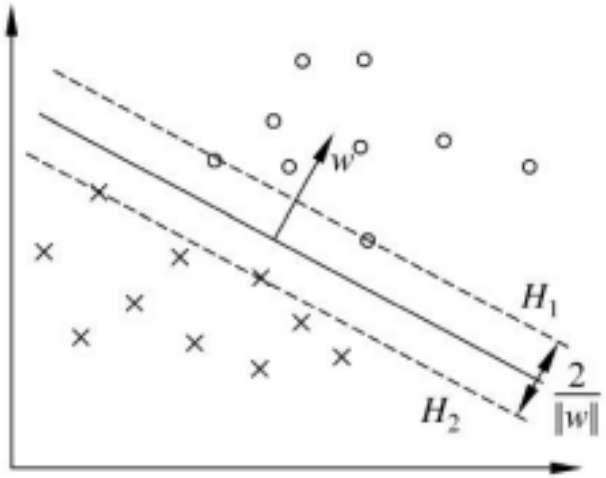
（2）由此得到分离超平面：$w^* \cdot x + b^* = 0$

分类决策函数 $f(x) = sign(w^* \cdot x + b^*)$

线性可分训练数据集的最大间隔分离超平面是存在且唯一的。

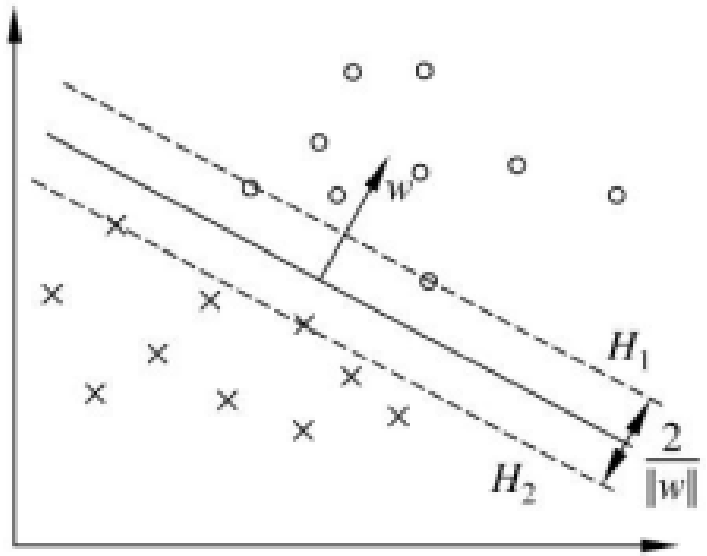
支持向量和间隔边界

- 在线性可分情况下，训练数据集的样本点中与分离超平面最近的样本点的实例称为支持向量（support vector）。
- 支持向量是使得$y_i(w \cdot w_i + b) - 1 = 0$成立的点，对$y_i = +1$的正例点，支持向量在超平面$H_1: w \cdot x + b = 1$上，对$y_i = -1$的负例点，支持向量在超平面$H_2: w \cdot x + b = -1$上。
- 如图， H_1和H_2上的点就是支持向量。



支持向量和间隔边界

- 在H_1和H_2之间形成一条长带，分离超平面与它们平行且位于它们中央。
- 长带的宽度，即H_1和H_2之间的距离称为间隔。间隔依赖于分离超平面的法向量w，等于$\frac{2}{\|w\|}$。
- H_1和H_2称为间隔边界。



支持向量和间隔边界

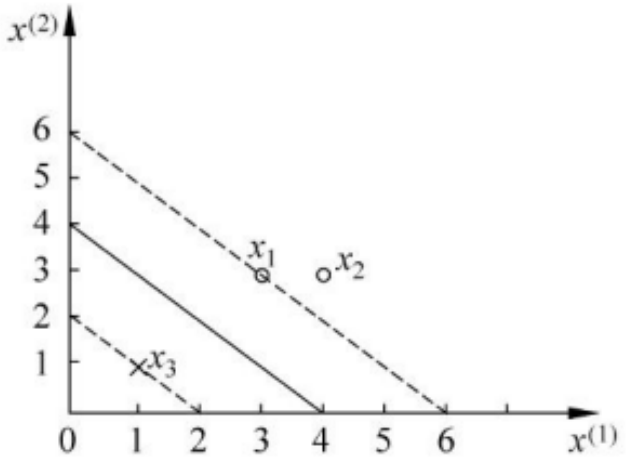
- 在决定分离超平面时只有支持向量起作用，而其他实例点并不起作用。如果移动支持向量将改变所求的解；但是如果在间隔边界以外移动其他实例点，甚至去掉这些点，则解是不会改变的。
- 由于支持向量在确定分离超平面中起着决定性作用，所以将这种分类模型称为支持向量机。支持向量的个数一般很少，所以支持向量机由很少的“重要的”训练样本确定。

例题

- 例4.1 已知如图所示的训练数据集，其正例点是$x_1 = (3,3)^{\mathrm{T}}, x_2 = (4,3)^{\mathrm{T}}$，负例点是$x_3 = (1,1)^{\mathrm{T}}$，试求最大间隔分离超平面。

解：按照算法4.1，根据训练数据集构造约束最优化问题：

$$\min_{w,b} \ \frac{1}{2}\left(w_1^2 + w_2^2\right)$$
$$s.t. \quad 3w_1 + 3w_2 + b \geq 1$$
$$4w_1 + 3w_2 + b \geq 1$$
$$-w_1 - w_2 - b \geq 1$$



求得此最优问题的解$w_1 = w_2 = \frac{1}{2}, b = -2$。于是最大间隔分离超平面为

$$\frac{1}{2}x^{(1)} + \frac{1}{2}x^{(2)} - 2 = 0$$

其中，$x_1 = (3,3)^{\mathrm{T}}$与$x_3 = (1,1)^{\mathrm{T}}$为支持向量。

学习的对偶算法

- 为了求解线性可分支持向量机的最优化问题，将它作为原始最优化问题，应用拉格朗日对偶性，通过求解对偶问题(dual problem)得到原始问题(primal problem)的最优解，这就是线性可分支持向量机的对偶算法(dual algorithm)。

- 这样做的优点:
  - ✓对偶问题往往更容易求解;
  - ✓自然引入核函数，进而推广到非线性分类问题。

拉格朗日对偶性

1.原始问题

假设$f(x)$，$c_i(x)$，$h_j(x)$是定义在R^n上的连续可微函数。考虑约束最优化问题

$$\min_{x \in R^n} f(x) \qquad (C.1)$$

$$s.t.\ \ c_i(x) \leq 0.\ \ i = 1,2,\cdots,k \qquad (C.2)$$

$$g_j(x) = 0.\ \ i = 1,2,\cdots,l \qquad (C.3)$$

称此约束最优化问题为原始最优化问题或原始问题。

学习的对偶算法

首先构建拉格朗日函数。引入拉格朗日乘子 $\alpha_i \geq 0$，定义拉格朗日函数

$$L(w,b,\alpha) = \frac{1}{2}\|w\|^2 - \sum_{i=1}^{N} \alpha_i y_i (w \cdot x_i + b) + \sum_{i=1}^{N} \alpha_i \quad (4.18)$$

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,α）对w,b的极小，再求对α的极大。

学习的对偶算法

（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求偏导数另其等于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， $\mathrm{w} = \sum_{i=1}^{N} \alpha_i y_i x_i$ (4.19)　　$\sum_{i=1}^{N} \alpha_i y_i = 0$ (4.20)

再回代入拉格朗日函数，得到

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

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

即

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

学习的对偶算法

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

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\left(x_i \cdot x_j\right)+\sum_{i=1}^{N}\alpha_i \qquad (4.21)$$

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

$$\alpha_i \geq 0, i = 1,2,\ldots,N$$

学习的对偶算法

将式(4.21)的目标函数由极大转换成求极小，就得到与之等价的对偶最优化问题:

$$\min_{\alpha} \quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j\left(x_i \cdot x_j\right) - \sum_{i=1}^{N}\alpha_i \qquad (4.22)$$

$$s.t. \quad \sum_{i=1}^{N}\alpha_i y_i = 0 \qquad (4.23)$$

$$\alpha_i \geq 0, i = 1,2,\ldots,N \qquad (4.24)$$

考虑原始最优化问题(4.13)~(4.14)和对偶最优化问题(4.22)~(4.24)，存在w^*、α^*、β^*，使w^*是原始问题的解，α^*、β^*是对偶问题的解。这就意味着求解原始问题(4.13)~(4.14)可以转换为求解对偶问题(4.22)~(4.24)。

学习的对偶算法

定理4.1 设$\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*)^T$是对偶最优化问题的解，则存在下标j，使得$\alpha_j^* > 0$，并可按下式求得原始最优化问题的解$w^*$、$b^*$。

$$w^* = \sum_{i=1}^{N} \alpha_i^* y_i x_i \quad (4.25)$$

$$b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i (x_i \cdot x_j) \quad (4.26)$$

对于给定的线性可分训练数据集，可以首先求对偶问题(4.22)~(4.24)的解 α^*;再利用式(4.25)和式(4.26)求得原始问题的解w^*、b^*；从而得到分离超平面及分类决策函数。这种算法称为线性可分支持向量机的对偶学习算法，是线性可分支持向量机学习的基本算法。

学习的对偶算法

算法 7.2（线性可分支持向量机学习算法）

输入：线性可分训练数据集

$T = \{(x_1, y_1), (x_2, y_2), \cdots, (x_N, y_N)\}$，其中，$x_i \in \mathcal{X} = \mathrm{R}^{\mathrm{n}}$，$y_i \in \mathcal{Y} = \{+1, -1\}$，$\mathrm{i} = 1,2,\cdots,\mathrm{N}$；

输出：分离超平面和分类决策函数。

学习的对偶算法

（1）构造并求解约束最优化问题

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

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

$$\alpha_i \geq 0, \qquad i = 1,2,\cdots,N$$

求得最优解$\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_N^*)^T$。

学习的对偶算法

（2）计算

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

并选择α^*的一个正分量$\alpha_j^* > 0$，计算

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

学习的对偶算法

（3）求得分离超平面

$$w^* \cdot x + b^* = 0$$

分类决策函数：

$$f(x) = sign(w^* \cdot x + b^*)$$

在线性可分支持向量机中，由式（7.25）、式（7.26）可知，w^*和b^*只依赖于训练数据中对应于$\alpha_i^* > 0$的样本点(x_i, y_i)，而其他样本样本点对w^*和b^*没有影响。我们将训练数据中对应于$\alpha_i^* > 0$的实例点$x_i \in R^n$称为支持向量。

线性可分支持向量机学习算法

定义（支持向量） 考虑原始最优化问题及对偶最优化问题，将训练数据集中对应于$\alpha_i^* > 0$的样本点(x_i, y_i)的实例$x_i \epsilon R^n$称为支持向量。

根据这一定义，支持向量一定在间隔边界上。由KKT互补条件可知，

$$\alpha_i^*(y_i(w^* \cdot x_i + b^*) - 1) = 0, i = 1,2, \dots, N$$

对于$\alpha_i^* > 0$的实例x_i，有

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

或

$$w^* \cdot x_i + b^* = \pm 1$$

即x_i一定在间隔边界上。

这里的支持向量的定义与前面给出的支持向量的定义是一致的。

学习的对偶算法

例2 训练数据如图所示，正例点是$x_1=(3,3)^T$，$x_2=(4,3)^T$，负实例点是$x_3=(1,1)^T$，试用算法7.2（线性可分支持向量机学习算法）求线性可分支持向量机。

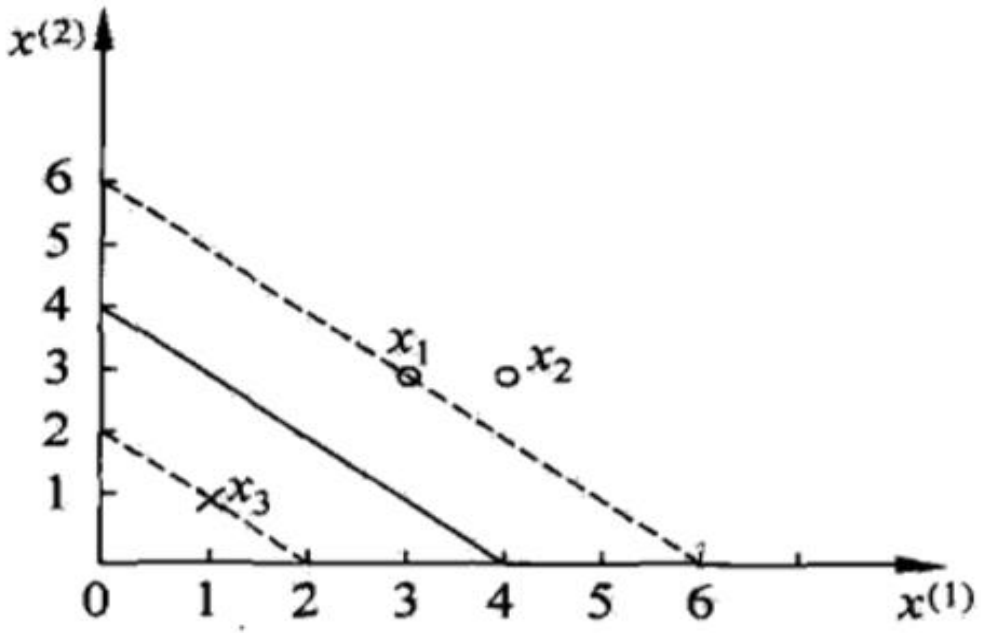


图 7.4 间隔最大分离超平面示例

学习的对偶算法

解 根据所给数据，对偶问题

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

$$= \frac{1}{2}\left(18\alpha_1^2 + 25\alpha_2^2 + 2\alpha_3^2 + 42\alpha_1\alpha_2 - 12\alpha_1\alpha_3 - 14\alpha_2\alpha_3\right) - \alpha_1 - \alpha_2 - \alpha_3$$

$$s.t. \quad \alpha_1 + \alpha_2 - \alpha_3 = 0$$

$$\alpha_i \geq 0, \qquad i = 1,2,3$$

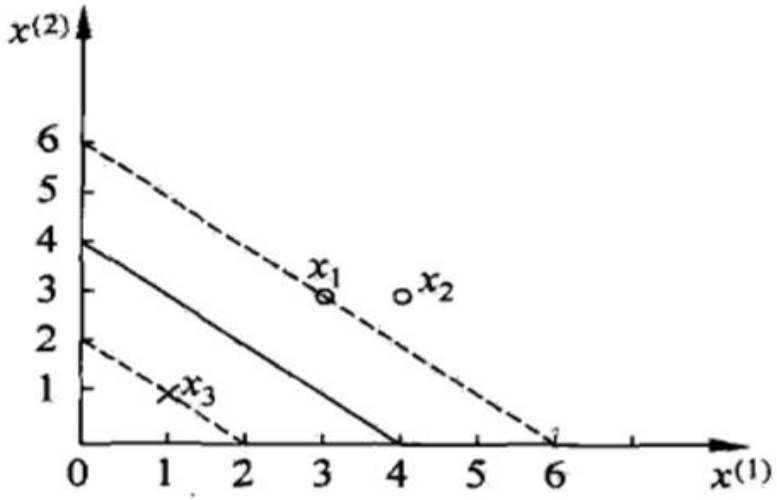


图 7.4 间隔最大分离超平面示例

学习的对偶算法

解这一最优化问题。

将$\alpha_3 = \alpha_1 + \alpha_2$代入目标函数并记为

$$s(\alpha_1, \alpha_2) = 4\alpha_1^2 + \frac{13}{2}\alpha_2^2 + 10\alpha_1\alpha_2 - 2\alpha_1 - 2\alpha_2$$

对α_1，α_2求偏导数并令其为0，易知$s(\alpha_1, \alpha_2)$在点$(\frac{3}{2}, -1)^T$取极值，但该点不满足约束条件$\alpha_2 \geq 0$，所以最小值在边界上达到。

当$\alpha_1 = 0$时，最小值$s\left(0, \frac{2}{13}\right) = -\frac{2}{13}$；当$\alpha_2 = 0$时，最小值$s\left(\frac{1}{4}, 0\right) = -\frac{1}{4}$。于是$s(\alpha_1, \alpha_2)$在$\alpha_1 = \frac{1}{4}$，$\alpha_2 = 0$达到最小，此时$\alpha_3 = \alpha_1 + \alpha_2 = \frac{1}{4}$。

学习的对偶算法

这样，$\alpha_1^* = \alpha_3^* = \frac{1}{4}$对应的实例点$x_1$，$x_3$是支持向量。根据式（7.25）和式（7.26）计算得

$$w_1^* = w_2^* = \frac{1}{2}$$

$$b_1^* = -2$$

分离超平面为

$$\frac{1}{2}x^{(1)} + \frac{1}{2}x^{(2)} - 2 = 0$$

分离决策函数为

$$f(x) = sign\left(\frac{1}{2}x^{(1)} + \frac{1}{2}x^{(2)} - 2\right)$$

对偶问题

- 拉格朗日乘子法
  - 第一步：引入拉格朗日乘子$\alpha_i \geq 0$得到拉格朗日函数

$$L(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \frac{1}{2}\|\boldsymbol{w}\|^2 - \sum_{i=1}^{m} \alpha_i \left(y_i(\boldsymbol{w}^\top \boldsymbol{x}_i + b) - 1\right)$$

  - 第二步：令$L(\boldsymbol{w}, b, \boldsymbol{\alpha})$对$\boldsymbol{w}$和$b$的偏导为零可得

$$\boldsymbol{w} = \sum_{i=1}^{m} \alpha_i y_i \boldsymbol{x}_i, \quad \sum_{i=1}^{m} \alpha_i y_i = 0.$$

  - 第三步：回代

$$\min_{\boldsymbol{\alpha}} \ \frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m} \alpha_i \alpha_j y_i y_j \boldsymbol{x}_i^\top \boldsymbol{x}_j - \sum_{i=1}^{m} \alpha_i$$

$$\text{s.t.} \ \sum_{i=1}^{m} \alpha_i y_i = 0, \ \alpha_i \geq 0, \ i = 1, 2, \ldots, m.$$

目录

线性支持向量机

线性可分问题的支持向量机学习方法，对线性不可分训练数据是不适用的，因为这时上述方法中的不等式约束并不能都成立。怎么才能将它扩展到线性不可分问题呢?

这就需要修改硬间隔最大化，使其成为软间隔最大化

线性支持向量机

- 假设给定一个特征空间上的训练数据集$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$,其中，$x_i \in R^n, y_i \in Y = \{+1, -1\}, i = 1,2, \dots, N$, x_i为第i个特征向量， y_i为x_i的类标记。

- 再假设训练数据集不是线性可分的。通常情况是，训练数据中有一些特异点(outlier)，将这些特异点除去后，剩下大部分的样本点组成的集合是线性可分的。

- 线性不可分意味着某些样本点(x_i, y_i)不能满足函数间隔大于等于1的约束条件。为了解决这个问题，可以对每个样本点(x_i, y_i)引进一个松弛变量$\xi_i \geq 0$，使函数间隔加上松弛变量大于等于1。这样，约束条件变为

$$y_i(w \cdot x_i + b) \geq 1 - \xi_i$$

线性支持向量机

- 对每个松弛变量ξ_i ，支付一个代价ξ_i 。目标函数变成:

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

- 这里, C>0称为惩罚参数，一般由应用问题决定，C值大时对误分类的惩罚增大，C值小时对误分类的惩罚减小。

- 最小化目标函数包含两层含义:使$\frac{1}{2}\|w\|^2$尽量小即间隔尽量大，同时使误分类点的个数尽量小，C是调和二者的系数。

线性支持向量机

- 有了上面的思路，可以和训练数据集线性可分时一样来考虑训练数据集线性不可分时的线性支持向量机学习问题。相应于硬间隔最大化，它称为软间隔最大化。
- 线性不可分的线性支持向量机的学习问题变成如下凸二次规划问题(原始问题):

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \qquad (4.28)$$

$$s.t. \qquad y_i(w \cdot x_i + b) \geq 1 - \xi_i, \qquad i = 1,2,\dots,N \qquad (4.29)$$

$$\xi_i \geq 0, \qquad i = 1,2,\dots,N \qquad (4.30)$$

- 原始问题(4.28)~(4.30)是一个凸二次规划问题，因而关于(w,b,ξ)的解是存在的。

线性支持向量机

- 设上述问题的解是w*，b*，于是可以得到分离超平面w*· x ＋b*= 0及分类决策函数f(x) = sign(w*· x ＋b*)。称这样的模型为训练样本线性不可分时的线性支持向量机，简称为线性支持向量机。

- 显然，线性支持向量机包含线性可分支持向量机。

- 由于现实中训练数据集往往是线性不可分的，线性支持向量机具有更广的适用性。

线性支持向量机

定义（线性支持向量机） 对于给定的线性不可分的训练数据集，通过求解凸二次规划问题，即软间隔最大化问题， 得到的分离超平面为

$$w^* \cdot x + b^* = 0$$

以及相应的分类决策函数

$$f(x) = sign(w^* \cdot x + b^*)$$

称为线性支持向量机。

学习的对偶算法

原始问题

$$\min_{\alpha} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i$$

$$s.t. \quad y_i(w \cdot x_i + b) \geq 1 - \xi_i, i = 1,2,\dots,N$$

$$\xi_i \geq 0, \quad i = 1,2,\dots,N$$

对偶问题

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j y_i y_j\left(x_i \cdot x_j\right) - \sum_{i=1}^{N}\alpha_i \qquad (4.33)$$

$$s.t. \quad \sum_{i=1}^{N}\alpha_i y_i = 0, \qquad (4.34)$$

$$0 \leq \alpha_i \leq C, i = 1,2,\dots,N \qquad (4.35)$$

学习的对偶算法

- 原始问题的拉格朗日函数

$$L(w,b,\xi,\alpha,\mu)=\frac{1}{2}\|w\|^2+C\sum_{i=1}^{N}\xi_i-\sum_{i=1}^{N}\alpha_i(y_i(w\cdot x_i+b)-1+\xi_i)-\sum_{i=1}^{N}\mu_i\xi_i \quad (4.36$$

其中，$\alpha_i\geq 0,\mu_i\geq 0$。

- 对偶问题是拉格朗日函数的极大极小问题。

首先求$L(w,b,\xi,\alpha,\mu)$对w,b,ξ的极小，由

- $\nabla_w L(w,b,\xi,\alpha,\mu)=w-\sum_{i=1}^{N}\alpha_i y_i x_i=0$
- $\nabla_b L(w,b,\xi,\alpha,\mu)=-\sum_{i=1}^{N}\alpha_i y_i$
- $\nabla_\xi L(w,b,\xi,\alpha,\mu)=C-\alpha_i-\mu_i=0$

得 $w=\sum_{i=1}^{N}\alpha_i y_i x_i$ (4.37) $\sum_{i=1}^{N}\alpha_i y_i$ (4.38) $C-\alpha_i-\mu_i=0$ (4.39)

学习的对偶算法

将上述(4.37)~(4.39)式子代入(4.40)，得，

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

再对$\min_{w,b,\xi} L(w,b,\xi,\alpha,\mu)$求$\alpha$的极大，即得对偶问题

$$\max_{\alpha} \quad -\frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\,\alpha_j y_i y_j\left(x_i \cdot x_j\right) + \sum_{i=1}^{N}\alpha_i \qquad (4.40)$$

$$s.t.\sum_{i=1}^{N}\alpha_i y_i = 0 \quad (4.41)$$

$$C - \alpha_i - \mu_i = 0 \quad (4.42)$$

$$\alpha_i \geq 0 \quad (4.43)$$

$$\mu_i \geq 0, i = 1,2,\ldots,N \quad (4.44)$$

学习的对偶算法

- 将对偶最优化问题(4.40)~(4.44)进行变换，利用等式约束(4.42)消去μ_i,从而只留下变量α_i,并将约束(4.42)~(4.44)写成
  - $0 \leq \alpha_i \leq C \quad (4.45)$
- 再将对目标函数求极大转换为求极小，于是得到对偶问题(4.33)~(4.35)。

- 可以通过求解对偶问题而得到原始问题的解，进而确定分离超平面和决策函数。为此，就可以定理的形式叙述原始问题的最优解和对偶问题的最优解的关系。

学习的对偶算法

定理2 设$\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*)^T$是对偶最优化问题的一个解，则存在$\alpha^*$的一个分量$\alpha_j^*$，$0 < \alpha_j^* < C$，则原始问题的解$w^*$、$b^*$可按下式求得：

$$w^* = \sum_{i=1}^{N} \alpha_i^* y_i x_i \quad (4.46)$$

$$b^* = y_i - \sum_{i=1}^{N} y_i \alpha_i^* (x_i \cdot x_j) \quad (4.47)$$

学习的对偶算法

由此定理可知，分离超平面可以写成

$$\sum_{i=1}^{N} \alpha_i^* y_i (x \cdot x_i) + b^* = 0 \quad (4.48)$$

分类决策函数可以写成

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i^* y_i (x \cdot x_i) + b^*) \quad (4.49)$$

该式(4.49)为线性支持向量机的对偶形式。

线性支持向量机学习算法

算法4.3（线性支持向量机学习算法）

输入：线性可分训练集$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$,其中，$x_i \in R^n, y_i \in Y = \{+1, -1\}, i = 1,2, \dots, N$,

输出：分离超平面和分类决策函数

线性支持向量机学习算法

（1）选择惩罚参数C>0，构造并求解凸二次规划问题

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

$$s.t.\sum_{i=1}^{N}\alpha_i\,y_i=0,\,0\le\alpha_i\le C,i=1,2,\dots N$$

求得最优解

（2）计算$w^*=\sum_{i=1}^{N}\alpha_i^* y_i x_i$

选择α^*的一个分量$0<\alpha_{\mathrm{j}}^*<C$,计算

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

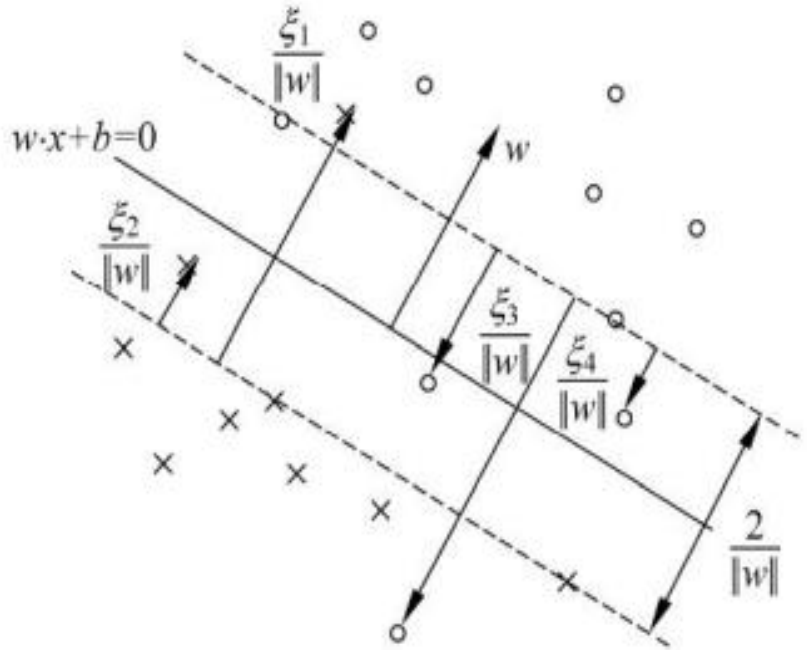
（3）求得分离超平面$w^*\cdot x+b^*=0$

分类决策函数：$f(x)=sign(w^*\cdot x+b^*)$

步骤（2）中，对任一适合条件$0<\alpha_j^*<C$的α_j^*，按式子(4.47)都可求出b*，从理论上，原始问题(4.28)~(4.30)对b的解可能不唯一，然而在实际应用中，往往只会出现算法叙述的情况。

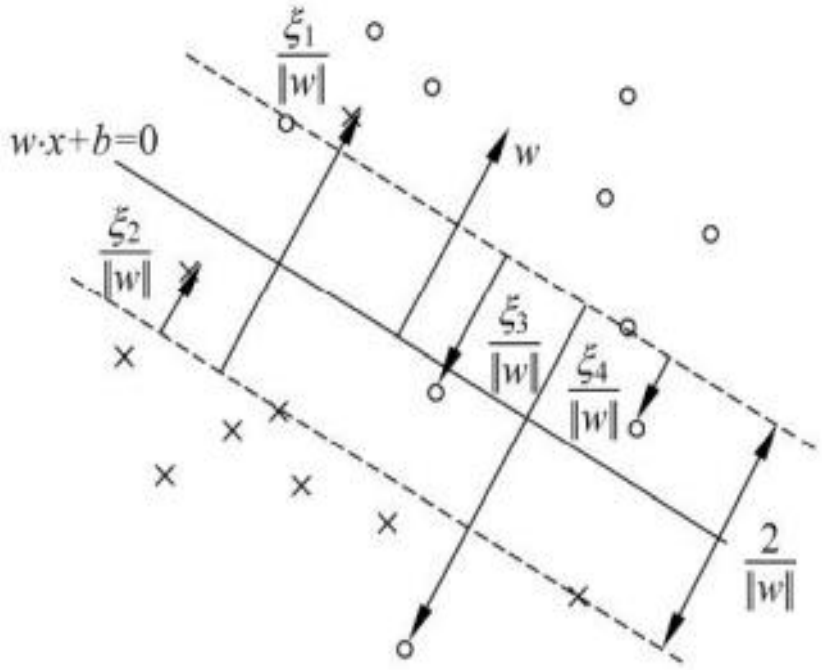
支持向量

- 在线性不可分的情况下，将对偶问题的解$\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_l^*)^T$中对应于$\alpha_i^* > 0$的样本点$(x_i, y_i)$的实例$x_i$称为支持向量(软间隔的支持向量)。
- 图中，分离超平面由实线表示，间隔边界由虚线表示，正例点由“o”表示，负例点由“x”表示。图中还标出了实例x_i到间隔边界的距离$\frac{\xi_i}{\|w\|}$。



支持向量

- 软间隔的支持向量x_i或者在间隔边界上，或者在间隔边界与分离超平面之间，或者在分离超平面误分一侧。
- 若$\alpha_i^* < C$，则$\xi_i = 0$，支持向量x_i恰好落在间隔边界上；
- 若$\alpha_i^* = C$，则$0 < \xi_i < 1$，则分类正确，x_i在间隔边界与分离超平面之间；
- 若$\alpha_i^* = C$，则$\xi_i = 0$，则x_i在分离超平面上；
- 若$\alpha_i^* = C$，则$\xi_i > 1$，则x_i位于分离超平面误分一侧。



合页损失函数

对于对于线性支持向量机学习来说，其模型为分离超平面w*·x + b*=0及决策函数f(x) = sign(w*· x + b*)，其学习策略为软间隔最大化，学习算法为凸二次规划。
线性支持向量机学习还有另一种解释，就是最小化以下目标函数：

$$\sum_{i=1}^{N}[1-y_i(w\cdot x_i+b)]_+ +\lambda\|w\|^2 \qquad (4.50)$$

目标函数的第1项是经验损失或经验风险，函数

$$L\big(y(w\cdot x+b)\big) == [1-y(w\cdot x+b)]_+ \qquad (4.51)$$

称为合页损失函数。下标“+”表示以下取正值的函数。

$$[z]_+ = \begin{cases} z, z>0 \\ 0, z\le 0 \end{cases} \qquad (4.52)$$

线性支持向量机原始最优化问题

定理4.4 线性支持向量机原是最优化问题：

$$\min_{w,b,\xi} \frac{1}{2}\|w\|^2 + C\sum_{i=1}^{N}\xi_i \quad (4.53)$$

$$s.t.\ \ y_i(w\cdot x_i + b) \geq 1 - \xi_i,, i = 1,2,\dots,N \quad (4.54)$$

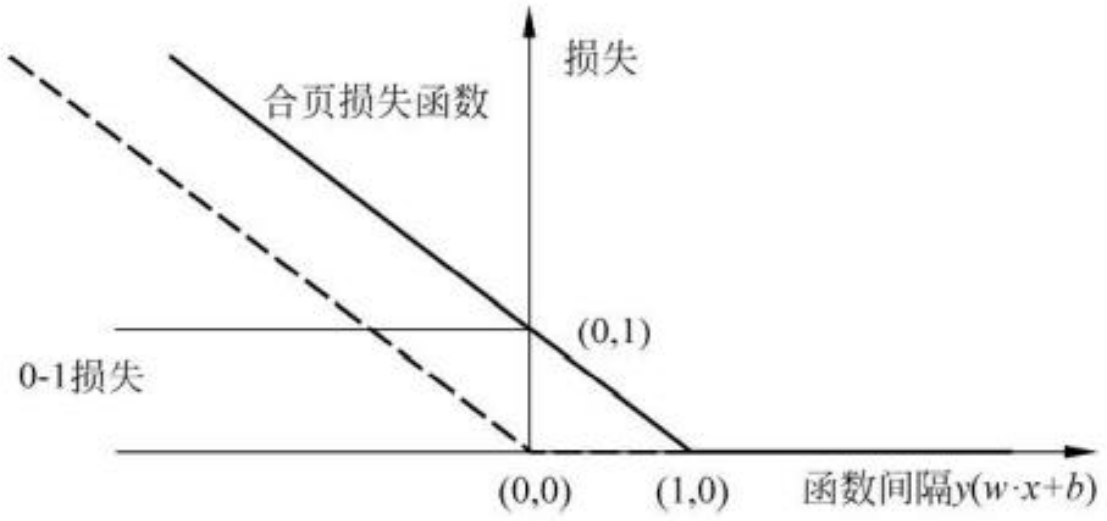
$$\xi_i \geq 0, i = 1,2,\dots,N \quad (4.55)$$

等价于最优化问题

$$\min_{w,b} \sum_{i=1}^{N}[1 - y_i(w\cdot x_i + b)]_+ + \lambda\|w\|^2 \quad (4.56)$$

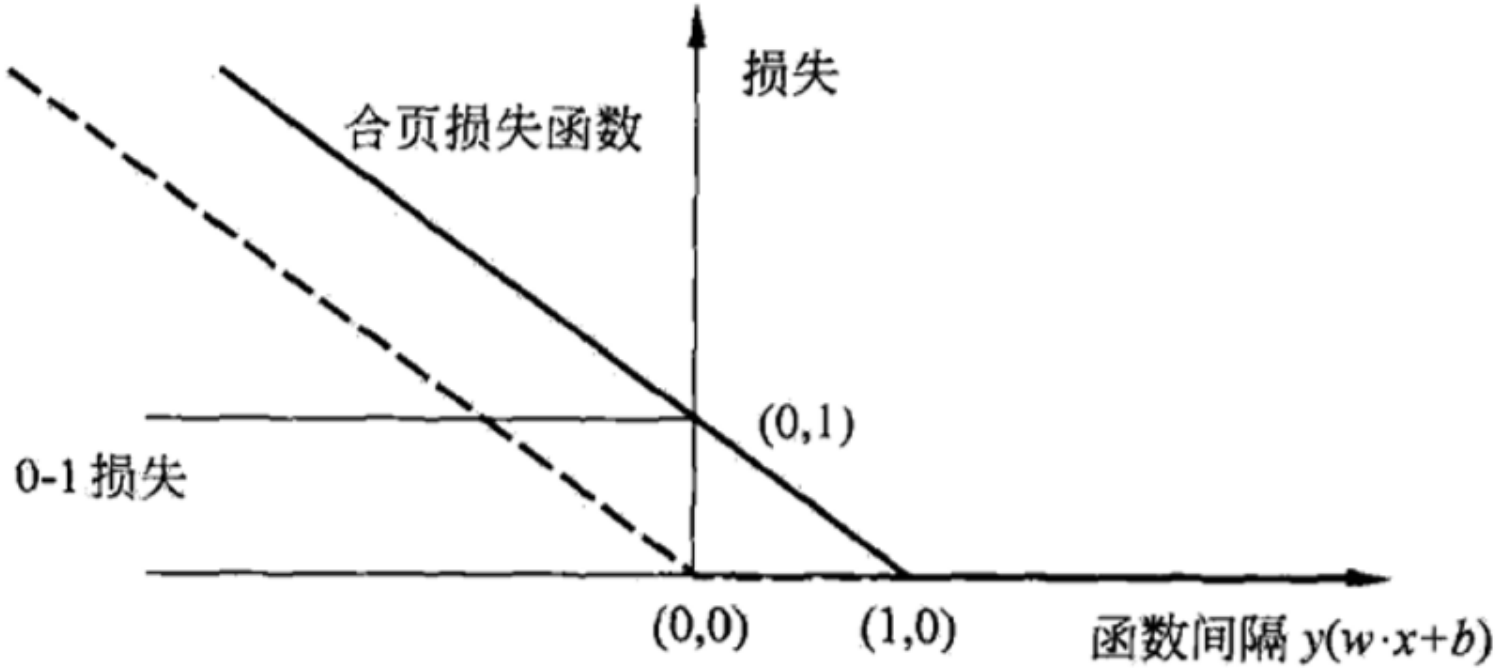
合页损失函数

- 合页损失函数的图形如图所示，横轴是函数间隔y(w·x + b)，纵轴是损失。由于函数形状像一个合页，故名合页损失函数。
- 0-1损失函数可以认为是二类分类问题的真正的损失函数，而合页损失函数是0-1损失函数的上界。由于0-1损失函数不是连续可导的，直接优化由其构成的目标函数比较困难，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。
- 上界损失函数又称为代理损失函数。



合页损失函数

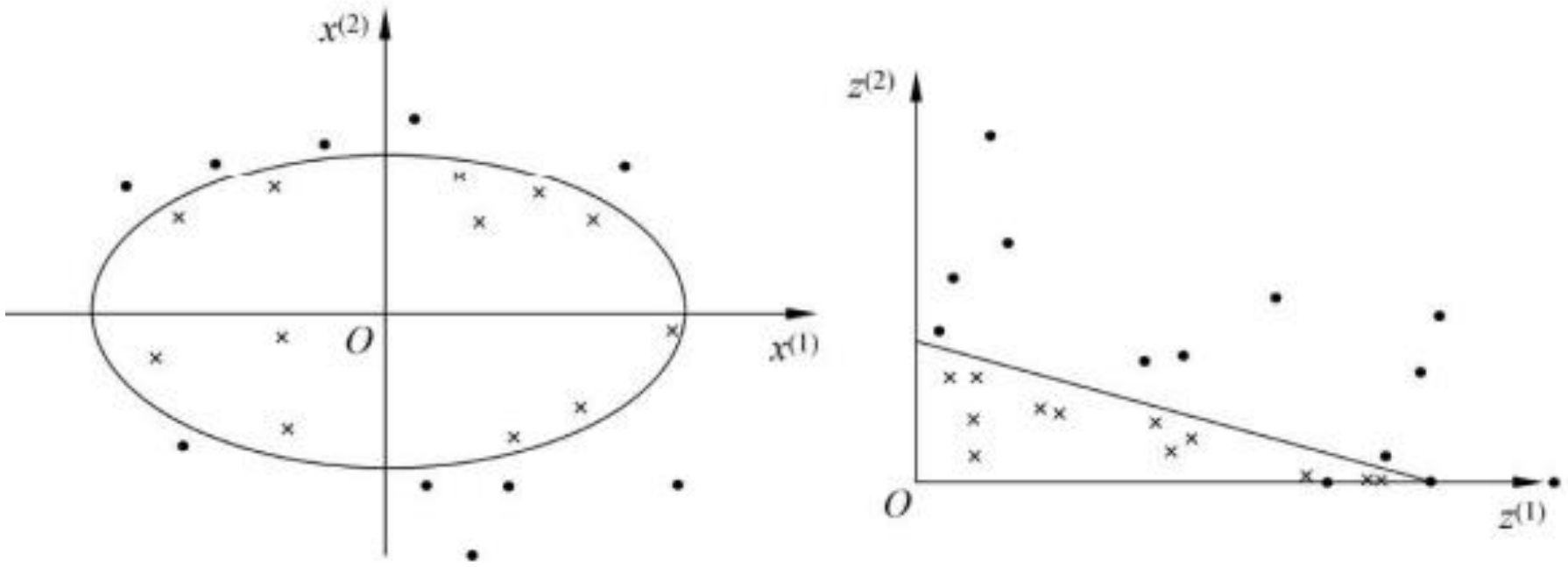
图中虚线显示的是感知机的损失函数$[-y_i(w \cdot x_i + b)]_+$。这时，当样本点$(x_i, y_i)$被正确分类时，损失是0，否则损失是$-y_i(w \cdot x_i + b)$。相比之下，合页损失函数不仅要分类正确，而且确信度足够高时损失才是0。也就是说，合页损失函数对学习有更高的要求。



目录

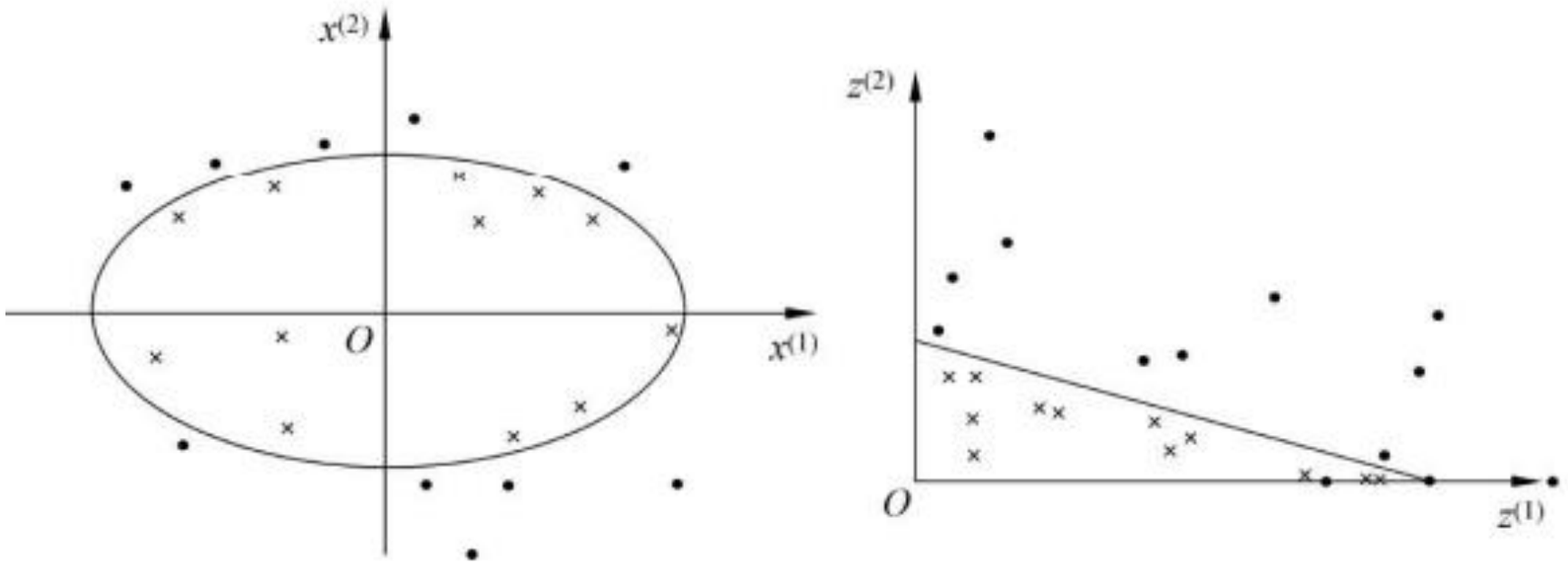
非线性分类问题

- 非线性分类问题是指通过利用非线性模型才能很好地进行分类的问题。
- 例子:如左图，是一个分类问题，图中“·”表示正实例点,“×”表示负实例点。
  - 由图可见，无法用直线(线性模型)将正负实例正确分开，但可以用一条椭圆曲线(非线性模型)将它们正确分开。



非线性分类问题

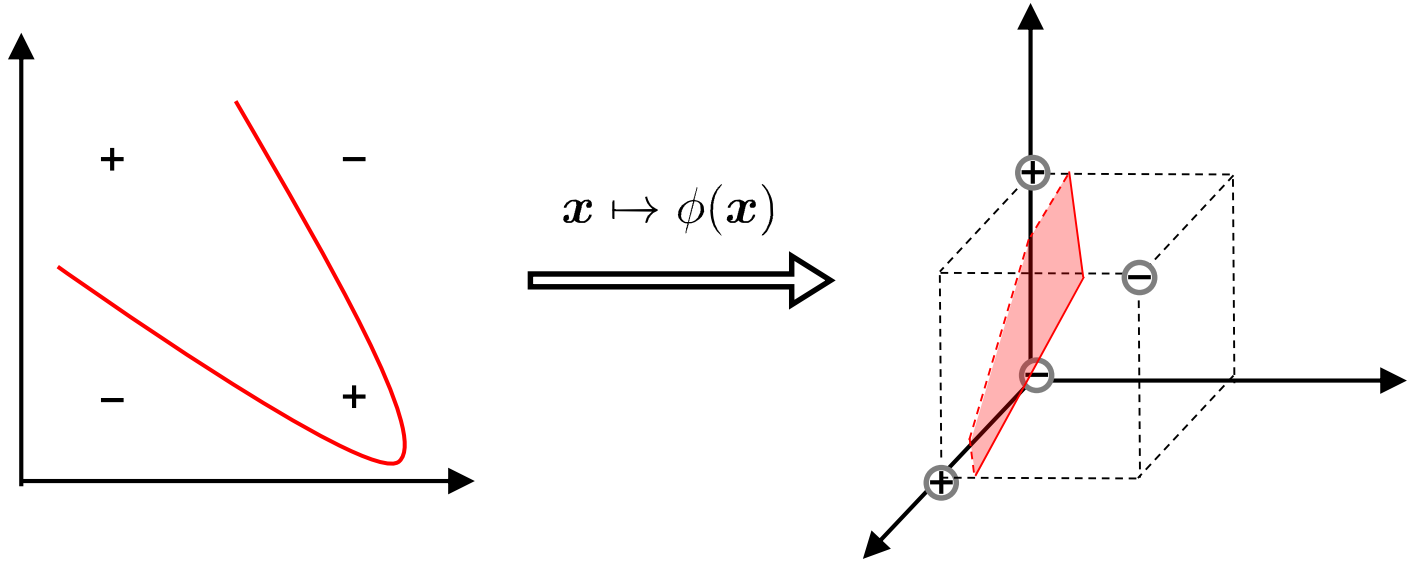
- 一般来说，对给定的一个训练数据集 $T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)\}$ ，其中，实例x_i属于输入空间， $x_i \, \epsilon X = R^n$ ，对应的标记有两类$y_i \epsilon Y = \{-1, +1\}, i = 1,2, \dots, N$。如果能用$R^n$中的一个超曲面将正负例正确分开，则称这个问题为非线性可分问题。



线性不可分

-Q:若不存在一个能正确划分两类样本的超平面,怎么办?

-A:将样本从原始空间映射到一个更高维的特征空间,使得样本在这个特征空间内线性可分。



线性不可分

设原空间为$X \subset R^2, x = \left(x^{(1)}, x^{(2)}\right)^{\mathrm{T}} \in X$, 新空间为$Z \epsilon R^2, z = (z^{(1)}, z^{(2)})^{\mathrm{T}} \epsilon Z$, 定义从原空间到新空间的变换（映射）：

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

经过变换$z = \emptyset(x)$, 原空间$x \subset R^2$变换为新空间$\mathrm{Z} \subset R^2$，原空间中的点相应地变换为新空间中的点，原空间中的椭圆

$$w_1(x^{(1)})^2 + w_2(x^{(2)})^2 + b = 0$$

变换成为新空间中的直线

$$w_1 z^{(1)} + w_2 z^{(2)} + b = 0$$

在变换后的新空间里，直线$w_1 z^{(1)} + w_2 z^{(2)} + b = 0$可以将变换后的正负实例点正确分开。这样，原空间的非线性可分问题就变成了新空间的线性可分问题。

核技巧

- 用线性分类方法求解非线性分类问题分为两步:
  - 首先使用一个变换将原空间的数据映射到新空间;然后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。
- 核技巧就属于这样的方法
- 核技巧应用到支持向量机，其基本想法：
  - 通过一个非线性变换将输入空间(欧氏空间R^n或离散集合）对应于一个特征空间(希尔伯特空间H)，使得在输入空间R^n中的超曲面模型对应于特征空间H中的超平面模型(支持向量机)。
  - 分类问题的学习任务通过在特征空间中求解线性支持向量机就可以完成。

核函数

定义4.6（核函数）设X是输入空间（欧式空间R^n的子集或离散集合），又设H为特征空间（希尔伯特空间），如果存在一个从X到H的映射

$$\phi(x):\ X \to H \quad (4.57)$$

使得对所有x,zϵX，函数K(x,z)满足条件

$$K(x,z) = \phi(x) \cdot \phi(z) \quad (4.58)$$

则称K(x,z)为核函数，ϕ(x)为映射函数，式中ϕ(x)· ϕ(z)为ϕ(x)和ϕ(z)的内积。

核函数

- 基本想法：不显式地设计映射, 而是设计核函数。

$$K(x_i, x_j) = \phi(x_i)^{\mathrm{T}} \phi(x_j)$$

绕过显式考虑特征映射、以及计算高维内积的困难

- Mercer定理：任何半正定对称函数都可以作为核函数。

- 任何一个核函数，都隐式地定义了一个RKHS（Reproducing Kernel Hilbert Space,再生核希尔伯特空间）。核函数的选择成为决定支持向量机性能的关键。

核函数

- 线性核，表示原空间内积，适用于线性可分问题
- 高斯核，适用于没有先验经验的非线性分类。
- 多项式核，适用于没有先验经验的分类。
- Sigmoid核，此时SVM实现的就是一种多层感知器神经网络。

核函数

口常用核函数：

名称	表达式	参数
线性核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \boldsymbol{x}_i^\top \boldsymbol{x}_j$	
多项式核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = (\boldsymbol{x}_i^\top \boldsymbol{x}_j)^d$	$d \geq 1$为多项式的次数
高斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|^2}{2\delta^2}\right)$	$\delta > 0$为高斯核的带宽(width)
拉普拉斯核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \exp\left(-\frac{\|\boldsymbol{x}_i - \boldsymbol{x}_j\|}{\delta}\right)$	$\delta > 0$
Sigmoid核	$\kappa(\boldsymbol{x}_i, \boldsymbol{x}_j) = \tanh(\beta \boldsymbol{x}_i^\top \boldsymbol{x}_j + \theta)$	tanh为双曲正切函数, $\beta > 0, \theta < 0$

非线性支持向量分类机

定义（非线性支持向量机）从非线性分类训练集，通过核函数与软间隔最大化，或凸二次规划，学习得到的分类决策函数

$$f(x) = sing(\sum_{i=1}^{N} \alpha_i^* y_i K(x, x_i) + b^*)$$

称为非线性支持向量机，K(x,z)是正定核函数。

非线性支持向量机学习算法

算法4（非线性支持向量机学习算法）

输入：训练数据集$T = \{(x_1, y_1), (x_2, y_2), \dots, (x_N, y_N)$

其中 $x_i \, \epsilon X = R^n, y_i \epsilon Y = \{-1, +1\}, i = 1,2, \dots, N$。

输出 ：分类决策函数

非线性支持向量机学习算法

(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

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j\, y_i y_j K(x,x_i) - \sum_{i=1}^{N}\alpha_i \qquad (4.62)$$

$$s.t. \sum_{i=1}^{N}\alpha_i\alpha_j = 0 \qquad (4.63)$$

$$0 \le \alpha_i \le C, \qquad i = 1,2,\dots,N \qquad (4.64)$$

求得最优解求得最优解 $\alpha^* = \left(\alpha_1^*, \alpha_2^*, \dots, \alpha_N^*\right)^{\mathrm{T}}$

非线性支持向量机学习算法

（2）并选择α^*的一个正分量$0 < \alpha_j^* < C$,计算

$$b^* = y_j - \sum_{i=1}^{N} \alpha_i^* y_i K(x_i \cdot x_j),$$

（3）构造决策函数：

$$f(x) = sign(\sum_{i=1}^{N} \alpha_i^* y_i K(x_i \cdot x_j) + b^*)$$

当K(x,z)是正定核函数时，问题(4.62)~(4.64)是凸二次规划问题，解是存在的。

目录

SMO

序列最小最优化(sequential minimal optimization SMO)算法：1998年由Platt提出。

动机：

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，并且有许多最优化算法可以用于这一问题的求解；

但是当训练样本容量很大时，这些算法往往变得非常低效，以致无法使用，所以，如何高效地实现支持向量机学习就成为一个重要的问题。

求解方法 - SMO

- 基本思路：不断执行如下两个步骤直至收敛.
  - 第一步：选取一对需更新的变量α_i和α_j.
  - 第二步：固定α_i和α_j以外的参数, 求解对偶问题更新α_i和α_j.

- 仅考虑α_i和α_j时, 对偶问题的约束变为

$$\alpha_i y_i + \alpha_j y_j = -\sum_{k \neq i,j} \alpha_k y_k, \quad \alpha_i \geq 0, \quad \alpha_j \geq 0.$$

用一个变量表示另一个变量, 回代入对偶问题可得一个单变量的二次规划, 该问题具有闭式解.

- 偏移项b：通过支持向量来确定.
  - 对任意支持向量$(\mathbf{x}_i, y_i)$ 有 $y_i(\mathbf{w}^\top \mathbf{x}_i + b) = 1$

SMO

SMO算法要解如下凸二次规划的对偶问题：

$$\min_{\alpha} \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j\, y_i y_j K(x, x_i) - \sum_{i=1}^{N}\alpha_i \qquad (4.65)$$

$$s.t. \sum_{i=1}^{N}\alpha_i\alpha_j = 0 \qquad (4.66)$$

$$0 \le \alpha_i \le C, \qquad i = 1,2,\dots,N \qquad (4.67)$$

SMO

- 启发式算法，基本思路：
- 如果所有变量的解都满足此最优化问题的KKT条件，那么得到解；
- 否则，选择两个变量，固定其它变量，针对这两个变量构建一个二次规划问题，称为子问题，可通过解析方法求解，提高了计算速度。
- 子问题的两个变量：一个是违反KKT条件最严重的那个，另一个由约束条件自动确定。
- 注意，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，$\alpha_3, \alpha_4, \ldots, \alpha_N$ 固定，那么由等式约束(4.66)可知$\alpha_1 = -y_i \sum_{i=2}^{N} \alpha_i y_i$，如果$\alpha_2$确定，那么$\alpha_1$也随之确定。所以子问题中同时更新两个变量。

SMO

- SMO算法包括两个部分：
  - 求解两个变量二次规划的解析方法
  - 选择变量的启发式方法

两个变量二次规划的求解方法

不失一般性，假设选择的两个变量是α_1, α_2，其他变量$\alpha_i (i = 3,4, \ldots, N)$是固定的。于是SMO的最优化问题(4.65)~(4.67)的子问题可以写成：

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

$$+ y_1\alpha_1 \sum_{i=3}^{N} y_i\alpha_i K_{i1} + y_2\alpha_2 \sum_{i=3}^{N} y_i\alpha_i K_{i2} \quad (4.68)$$

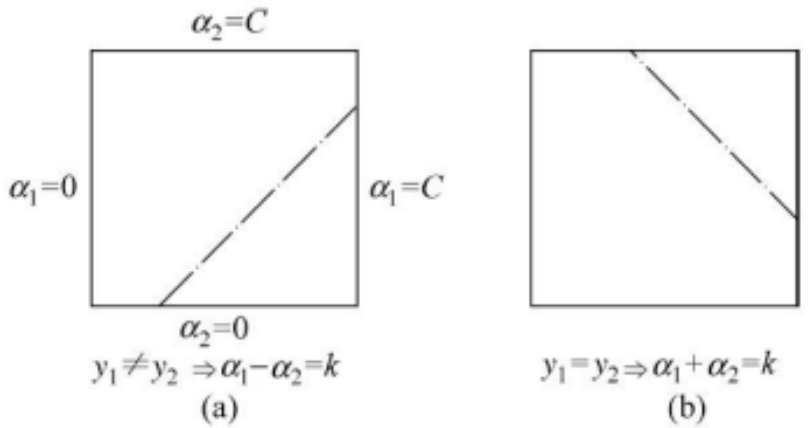
$$s.t. \quad \alpha_1 y_1 + \alpha_2 y_2 = -\sum_{i=3}^{N} y_i\alpha_i = \varsigma \quad (4.69)$$

$$0 \le \alpha_i \le C, \qquad i = 1,2 \quad (4.70)$$

其中，$K_{ij} = K(x_i, x_j), i, j = 1,2, \ldots, N$，$\varsigma$是常数。目标函数式(4.68)省略了不含$\alpha_1$，$\alpha_2$的常数项。

两个变量二次规划的求解方法

- 为了求解两个变量的二次规划问题(4.68)~(4.70),首先分析约束条件，然后在此约束条件下求极小。由于只有两个变量(α_1, α_2),约束可以用二维空间中的图形表示（如图）。
- 不等式约束(4.70)使得(α_1,α_2)在盒子[0,C]× [0,C]内，等式约束(4.69)使(α_1,α_2)在平行于盒子[0,C]×[0,C]的对角线的直线上。因此要求的是目标函数在一条平行于对角线的线段上的最优值。这使得两个变量的最优化问题成为实质上的单变量的最优化问题，不妨考虑为变量α_2的最优化问题。



两个变量二次规划的求解方法

- 假设(4.68)~(4.70)问题的初始可行解为α_1^{old}, α_2^{old} ，最优解为α_1^{new},α_2^{new} ，并且假设在沿着约束方向未经剪辑时α_2的最优解为$\alpha_2^{new,unc}$。
- 由于α_2^{new}需满足不等式约束(4.70)，所以最优值α_2^{new}的取值范围必须满足条件 $L \le \alpha_2^{new} \ll H$，其中，L与H是$\alpha_2^{new}$所在的对角线段端点的界。

如果$y_1 \neq y_2$, 则

$$L = \max\left(0, \alpha_2^{old} - \alpha_1^{old}\right), H = \min(C, C + \alpha_2^{old} - \alpha_1^{old})$$

如果$y_1 = y_2$, 则

$$L = \max\left(0, \alpha_2^{old} + \alpha_1^{old} - C\right), H = \min(C, \alpha_2^{old} + \alpha_1^{old})$$

两个变量二次规划的求解方法

下面，首先求沿着约束方向未经剪辑即未考虑不等式约束时α_2的最优解$\alpha_2^{\text{new},unc}$；然后再求剪辑后$\alpha_2$的解$\alpha_2^{\text{new}}$。我们用定理来叙述这个结果。为了叙述简单，记

$$g(x)=\sum_{i=1}^{N}\alpha_i y_i K(x_i\cdot x)+\text{b} \qquad (4.71)$$

令

$$E_{\text{i}}=g(x_i)-y_i=\left(\sum_{j=1}^{N}\alpha_j y_j K\left(x_j,x_i\right)+b\right)-y_i, i=1,2 \qquad (4.72)$$

当i=1,2时，E_i为函数g(x)对输入x_i的预测值与真实输出y_i之差。

两个变量二次规划的求解方法

定理 最优化问题(4.68)~(4.70)沿着约束方向未经剪辑时的解是

$$\alpha_2^{\text{new},unc} = \alpha_2^{old} + \frac{y_2(E_1 - E_2)}{\eta} \quad (4.73)$$

其中，

$$\eta = K_{11} + K_{22} - 2K_{12} = \|\Phi(x_1) - \Phi(x_2)\|^2 \quad (4.74)$$

$\phi(x)$是输入空间到特征空间的映射，$E_i, i = 1,2$由式(4.72)给出，经剪辑后α_2的解是

$$\alpha_2^{new} = \begin{cases} H, \alpha_2^{\text{new},unc} > H \\ \alpha_2^{\text{new},unc}, L \le \alpha_2^{\text{new},unc} \le H \\ L, \alpha_2^{\text{new},unc} < L \end{cases} \quad (4.75)$$

由α_2^{new}求得α_1^{new}是

$$\alpha_1^{new} = \alpha_1^{old} + y_1 y_2 \left(\alpha_2^{old} - \alpha_2^{new}\right) \quad (4.76)$$

变量的选择方法

1.第1个变量的选择

- SMO称选择第1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，并将其对应的变量作为第1个变量。具休地，检验训练样本点(x_i, y_i)是否满足KKT条件，即

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

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

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

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

- 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。

变量的选择方法

2. 第2个变量的选择

- SMO称选择第2个变量的过程为内层循环。假设在外层循环中已经找到第1个变量α_1，现在要在内层循环中找第2个变量α_2。第2个变量选择的标准是希望能使α_2有足够大的变化。

- 由式(4.73)和式(4.75)可知，α_2是依赖于|E- E_2|的，为了加快计算速度，一种简单的做法是选择α_2 ，使其对应的|E- E_2|最大。因为α_1已定， E_1也确定了。如果E_1是正的，那么选择最小的E_i作为E_2。如果E_1是负的，那么选择最大的E_i作为E_2 。为了节省计算时间，将所有E_i值保存在一个列表中。

变量的选择方法

2. 第2个变量的选择

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。

变量的选择方法

3.计算阈值b和差值E_i

在每次完成两个变量的优化后，都要重新计算阈值b。当$0 < \alpha_1^{new} < C$时，由KKT条件可知：

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

于是$b_1^{new} = y_1 - \sum_{i=3}^{N} \alpha_i y_i K_{i1} - \alpha_1^{new} y_1 K_{11} - \alpha_2^{new} y_2 K_{21}$ (4.77)

由E_1的定义式(4.72)有 $E_1 = \sum_{i=3}^{N} \alpha_i y_i K_{i1} - \alpha_1^{old} y_1 K_{11} - \alpha_2^{old} y_2 K_{21} - y_1$

则式(4.77)的前两项可写成：$y_1 - \sum_{i=3}^{N} \alpha_i y_i K_{i1} = -E_1 + \alpha_1^{old} y_1 K_{11} - \alpha_2^{old} y_2 K_{21} + b^{old}$

变量的选择方法

3.计算阈值b和差值E_i

代入式(4.77)，得

$$b_1^{new} = -E_1 + y_1 K_{11}\left(\alpha_1^{new} - \alpha_1^{old}\right) - y_2 K_{21}\left(\alpha_2^{new} - \alpha_2^{old}\right) + b^{old} \quad (4.78)$$

同样，如果$0 < \alpha_2^{new} < C$,那么,

$$b_2^{new} = -E_2 + y_1 K_{12}\left(\alpha_1^{new} - \alpha_1^{old}\right) - y_2 K_{22}\left(\alpha_2^{new} - \alpha_2^{old}\right) + b^{old} \quad (4.79)$$

变量的选择方法

- 如果α_1^{new}，α_2^{new}同时满足条件$0 < \alpha_i^{new} < C$， $i = 1,2$，那么$b_1^{new} = b_2^{new}$。如果α_1^{new}，α_2^{new}是0或者C，那么b_1^{new}和b_2^{new}以及它们之间的数都是符合KKT条件的阈值，这时选择它们的中点作为b^{new}。
- 在每次完成两个变量的优化之后，还必须更新对应的E_i值，并将它们保存在列表中。E_i的值更新要用到b^{new}值，以及所有支持向量对应的α_i：

$$E_i^{new} = \sum_S y_j \alpha_j K(x_i, x_j) + b^{new} - y_i \qquad (4.80)$$

其中， S是所有支持向量x_j的集合。

SMO算法

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

输出：近似值$\hat{\alpha}$

(1)去初值 $\alpha^{(0)} = 0$ 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(4.68)~(4.70)，求取最优解$\alpha_1^{(k+1)}$，$\alpha_2^{(k+1)}$，更新α为$\alpha^{(k+1)}$;

SMO算法

(3)若在精度ε范围内满足停机条件

$$\sum_{\mathrm{i}=1}^{N} \alpha_i y_i = 0\,, 0 \le \alpha_i \le C \;\; i = 1,2 \ldots N$$

$$y_i \cdot \mathrm{g}(x_i) = \begin{cases} \ge 1, \{x_i | \alpha_i = 0\} \\ = 1, \{x_i | 0 < \alpha_i < C\} \\ \le 1, , \{x_i | \alpha_i = C\} \end{cases}$$

其中，

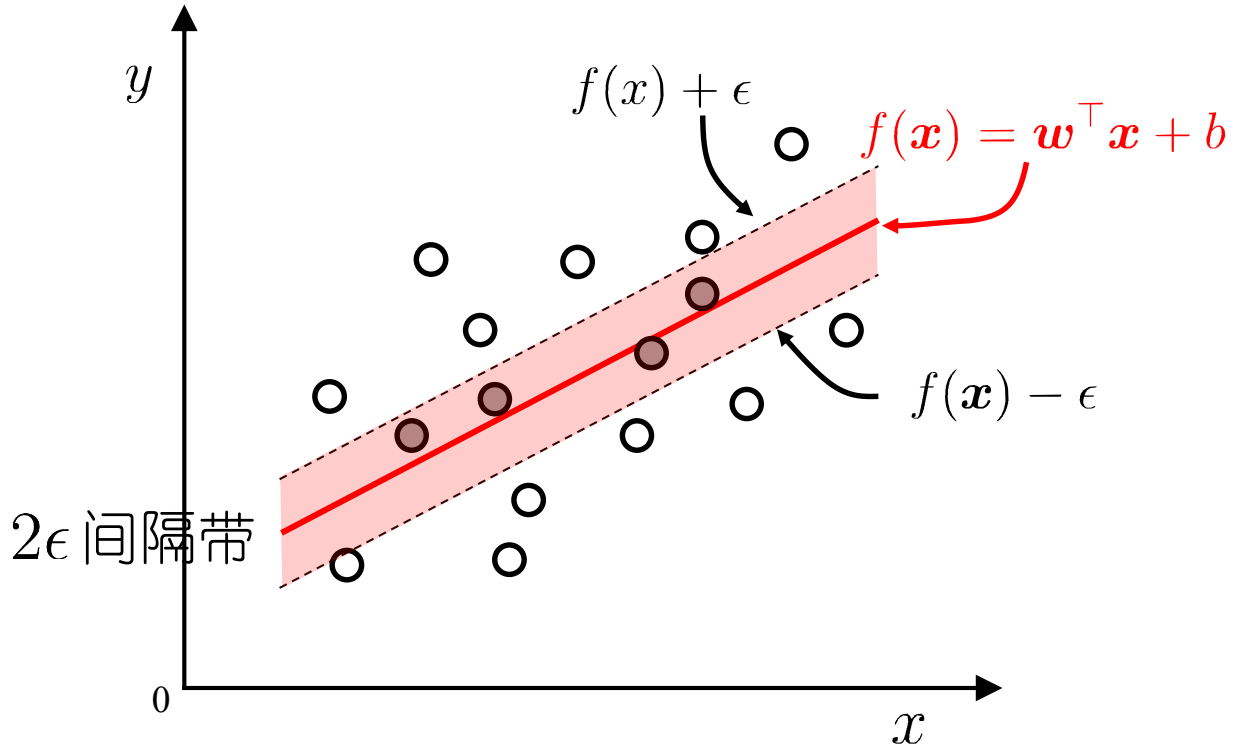
$$g(x_i) = \sum_{j=1}^{N} \alpha_j y_j K\left(x_j, x_i\right) + b$$

则转（4）;否则令k=k+1 ,转（2）;

（4）取$\hat{\alpha} = \alpha^{\mathrm{k+1}}$。

支持向量回归

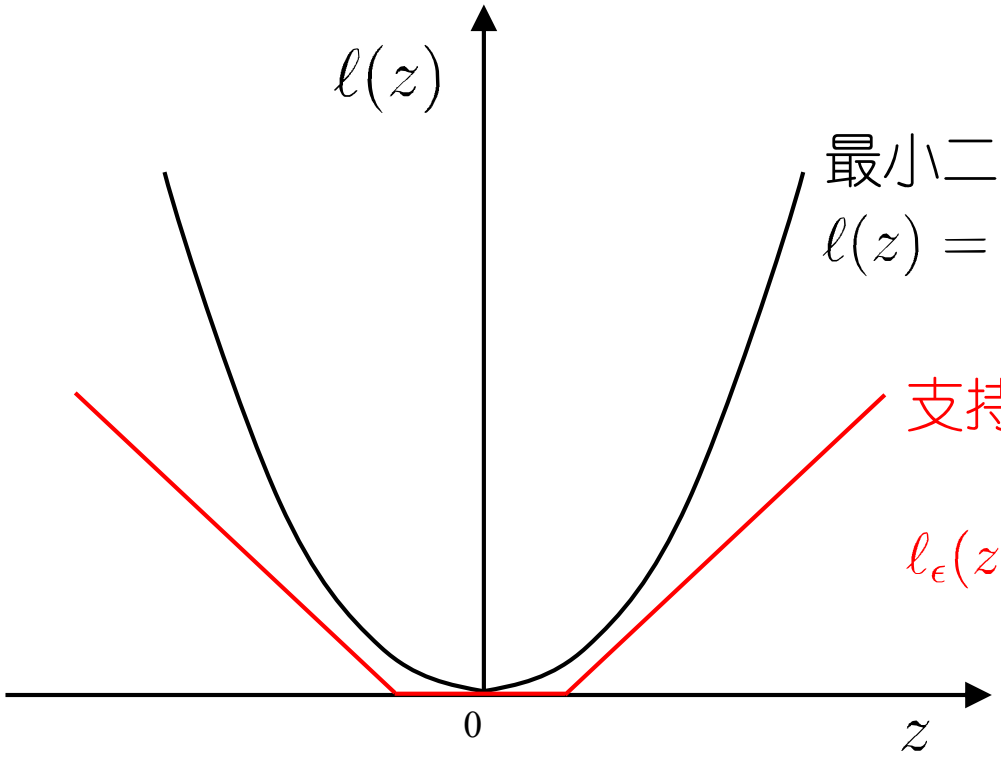
特点: 允许模型输出和实际输出间存在ϵ的偏差.



损失函数

落入中间2ϵ 间隔带的样本不计算损失, 从而使得模型获得稀疏性

.



最小二乘损失函数

$$\ell(z) = z^2$$

支持向量回归损失函数

$$\ell_\epsilon(z) = \begin{cases} 0 & \text{if } |z| \leq \epsilon \\ |z| - \epsilon & \text{otherwise} \end{cases}$$

形式化

原始问题

$$\min_{w,b,\xi_i,\hat{\xi}_i} \frac{1}{2}\|\boldsymbol{w}\|^2 + C\sum_{i=1}^{m}\left(\xi_i + \hat{\xi}_i\right)$$
$$\text{s.t. } f\left(\boldsymbol{x}_i\right) - y_i \leqslant \epsilon + \xi_i$$
$$y_i - f\left(\boldsymbol{x}_i\right) \leqslant \epsilon + \hat{\xi}_i$$
$$\xi_i \geqslant 0, \hat{\xi}_i \geqslant 0, i = 1, 2, \ldots, m$$

对偶问题

$$\max_{\boldsymbol{\alpha},\hat{\boldsymbol{\alpha}}} \sum_{i=1}^{m} y_i\left(\hat{\alpha}_i - \alpha_i\right) - \epsilon\left(\hat{\alpha}_i + \alpha_i\right)$$
$$-\frac{1}{2}\sum_{i=1}^{m}\sum_{j=1}^{m}\left(\hat{\alpha}_i - \alpha_i\right)\left(\hat{\alpha}_j - \alpha_j\right)\boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{x}_j$$
$$\text{s.t. } \sum_{i=1}^{m}\left(\hat{\alpha}_i - \alpha_i\right) = 0$$
$$0 \leqslant \alpha_i, \hat{\alpha}_i \leqslant C$$

预测

$$f(\boldsymbol{x}) = \boldsymbol{w}^{\top}\phi(\boldsymbol{x}) + b = \sum_{i=1}^{m}(\hat{\alpha}_i - \alpha_i)y_i\kappa(\boldsymbol{x}_i, \boldsymbol{x}) + b$$

Take Home Message

- 支持向量机的“最大间隔”思想
- 对偶问题及其解的稀疏性
- 通过向高维空间映射解决线性不可分的问题
- 引入“软间隔”缓解特征空间中线性不可分的问题
- 将支持向量的思想应用到回归问题上得到支持向量回归
- 将核方法推广到其他学习模型

成熟的SVM软件包

- LIBSVM
  http://www.csie.ntu.edu.tw/~cjlin/libsvm/
- LIBLINEAR
  http://www.csie.ntu.edu.tw/~cjlin/liblinear/
- SVM^{light}、SVM^{perf}、SVM^{struct}
  http://svmlight.joachims.org/svm_struct.html
- Pegasos
  http://www.cs.huji.ac.il/~shais/code/index.html

参考文献

[1] CORTES C, VAPNIK V. Support-vector networks[J]. Machine learning, 1995, 20(3): 273–297.

[2] Andrew Ng. Machine Learning[EB/OL]. Stanford University,2014. https://www.coursera.org/course/ml

[3] 李航. 统计学习方法[M]. 清华大学出版社,2019.

[4] Hastie T., Tibshirani R., Friedman J. The Elements of Statistical Learning[M]. Springer, New York, NY, 2001.

[5] CHRISTOPHER M. BISHOP. Pattern Recognition and Machine Learning[M]. Springer,2006.

[6] Stephen Boyd, Lieven Vandenberghe, Convex Optimization[M]. Cambridge University Press, 2004.

[7] PLATT J C. Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines[J]. 1998: 22.

谢 谢！