推荐书目

《*Pattern Recognition and Machine Learning*》

《*Machine Learning A Probabilistic Perspective*》

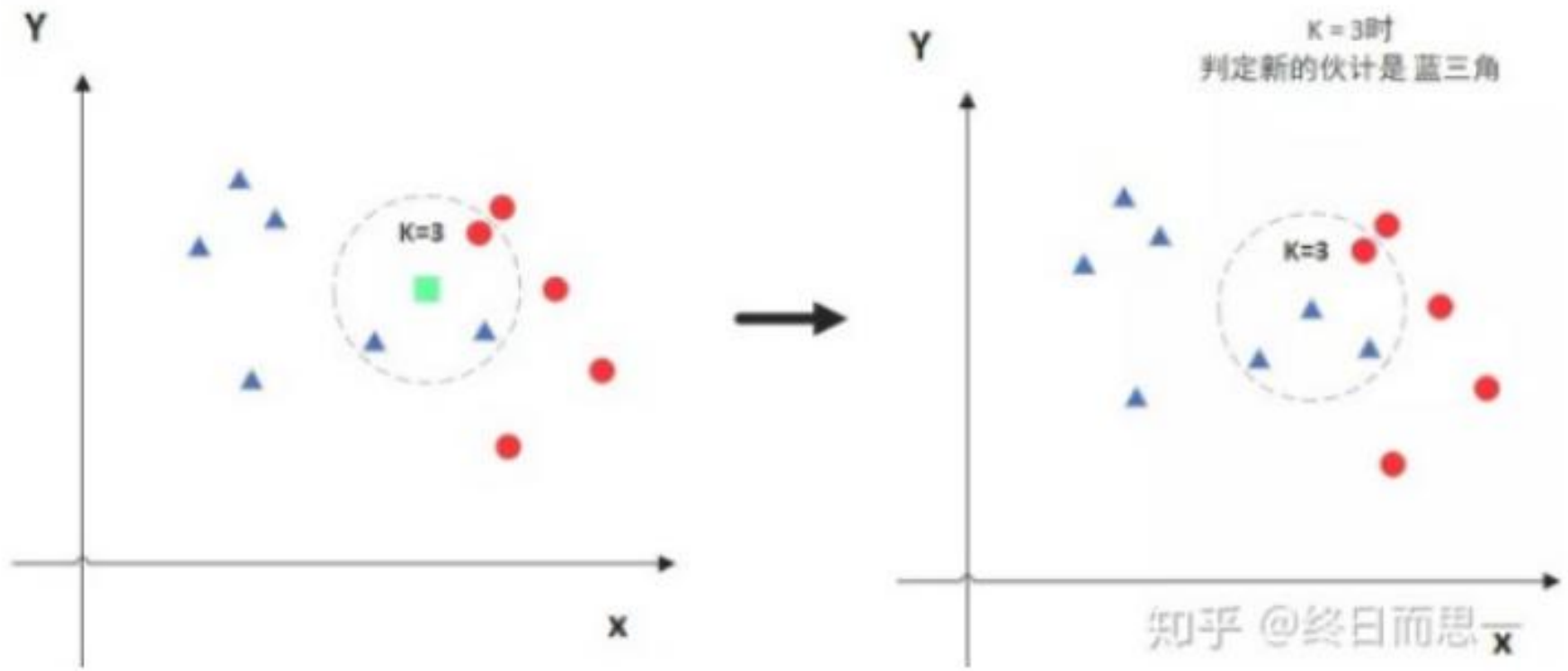
《*The Elements of Statistical Learning*》

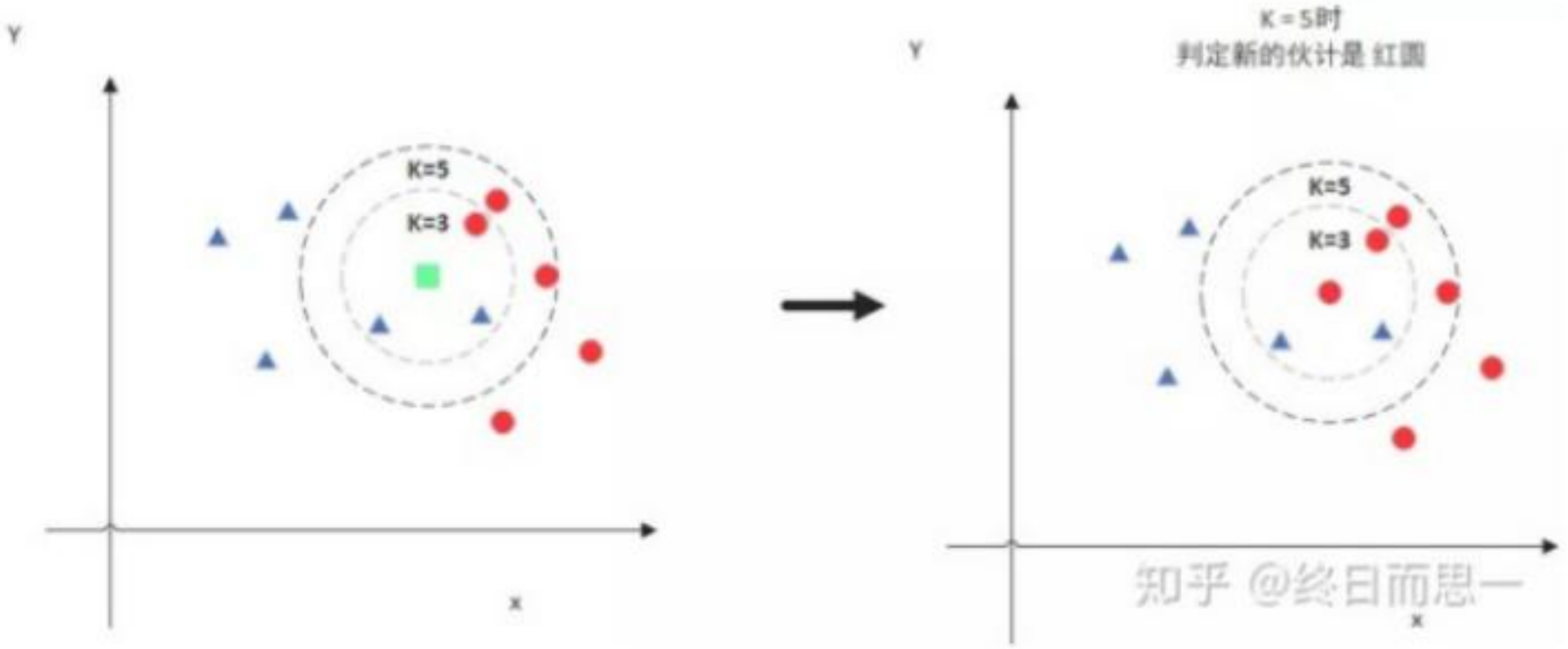
《*Machine Learning in Action*》

《统计学习方法》李航

5. *KNN（k-Nearest Neighbor）*

KNN可以说是最简单的分类算法之一，注意KNN算法是有监督学习中的分类算法。KNN的全称是K Nearest Neighbors，意思是K个最近的邻居，从这个名字我们就能看出一些KNN算法的蛛丝马迹了。K个最近邻居，毫无疑问，K的取值肯定是至关重要的。那么最近的邻居又是怎么回事呢？**其实啊，KNN的原理就是当预测一个新的值x的时候，根据它距离最近的K个点是什么类别来判断x属于哪个类别。**




Y
K=5
K=3
x
K = 5时
判定新的伙计是 红圆
Y
K=5
K=3
x
知乎 @终日而思一

欧式距离

二维平面上两个点(x_1, y_1), (x_2, y_2)间的欧式距离

$$d = \sqrt{(x_1 - x_2)^2 + (y_1 - y_2)^2}$$

曼哈顿距离

二维平面上两个点(x_1, y_1), (x_2, y_2)间的曼哈顿距离

$$d = |x_1 - x_2| + |y_1 - y_2|$$

切比雪夫距离

二维平面上两个点(x_1, y_1), (x_2, y_2)间的切比雪夫距离

$$d = max\{|x_1 - x_2|, |y_1 - y_2|\}$$

闵可夫斯基距离(*Minkowski Distance*)

二维平面上两个点(x_1, y_1), (x_2, y_2)间的闵可夫斯基距离

$$d = \sqrt[p]{(|x_1 - x_2|)^p + (|y_1 - y_2|)^p}$$

余弦距离

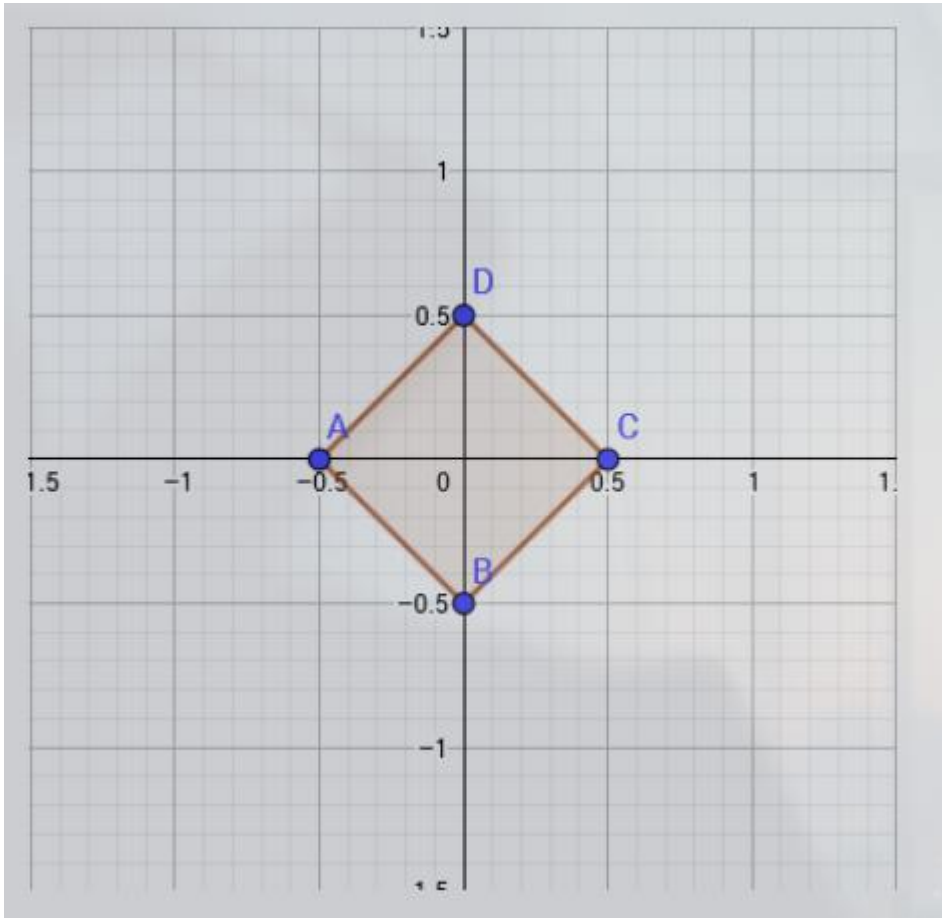
二维平面上两个点$(x_1, y_1), (x_2, y_2)$间的余弦距离

$$d = cos\theta = \frac{x_1 x_2 + y_1 y_2}{\sqrt{x_1^2 + y_1^2}\sqrt{x_2^2 + y_2^2}}$$

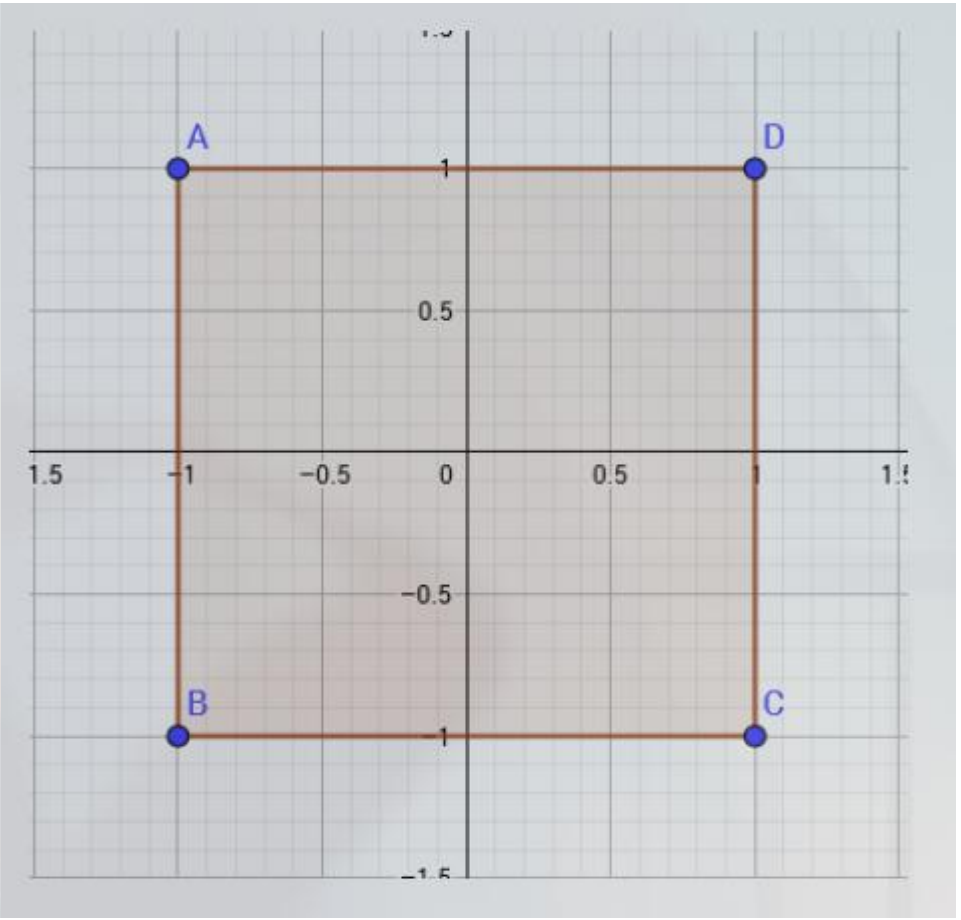
汉明距离

表示两个（相同长度）字对应位不同的数量，我们以d（x,y）表示两个字x,y之间的汉明距离。对两个字符串进行异或运算，并统计结果为1的个数，那么这个数就是汉明距离。

1011101 与 1001001 之间的汉明距离是 2。
2143896 与 2233796 之间的汉明距离是 3。
“toned” 与 “roses” 之间的汉明距离是 3。

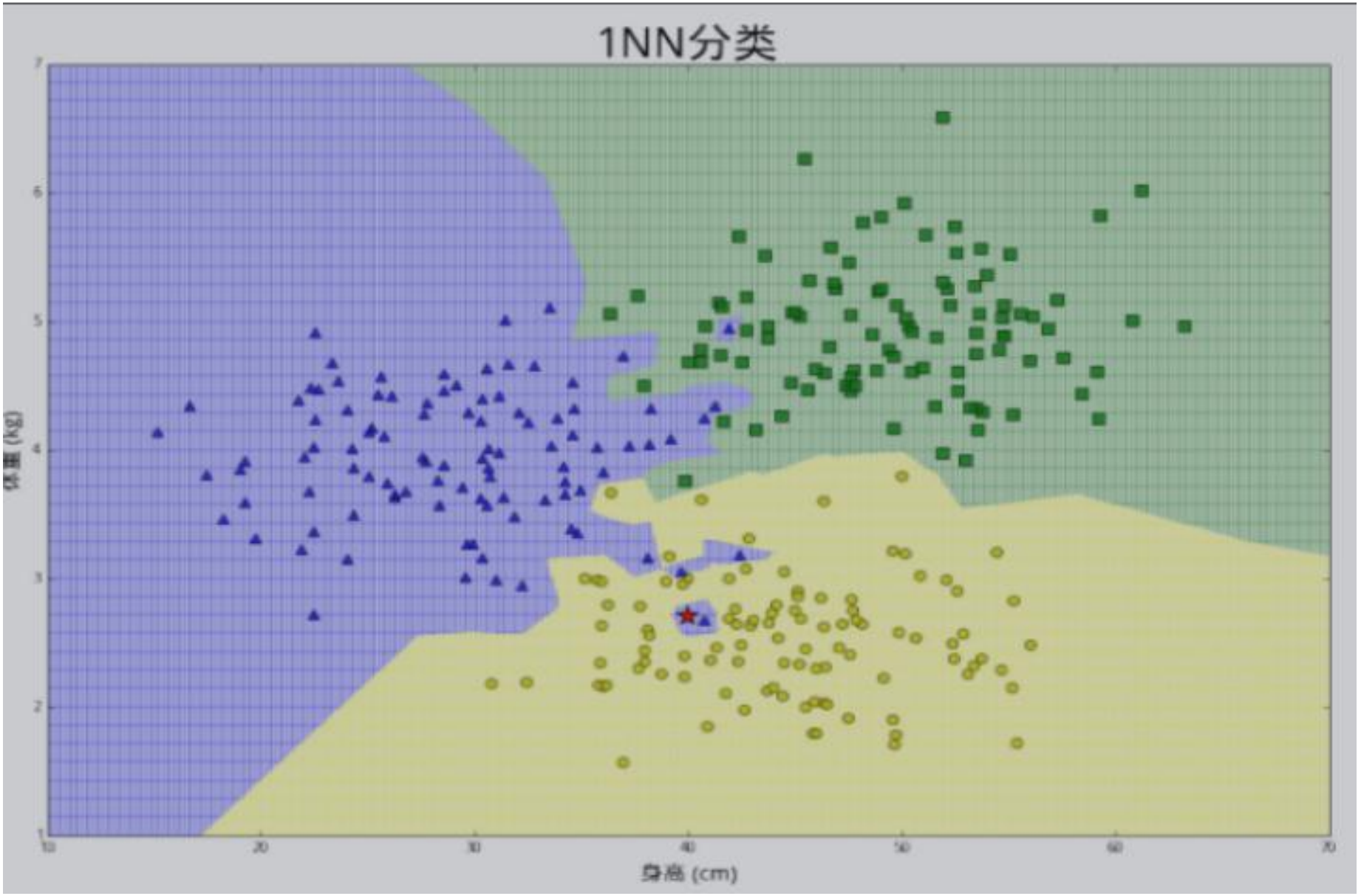


曼哈顿距离

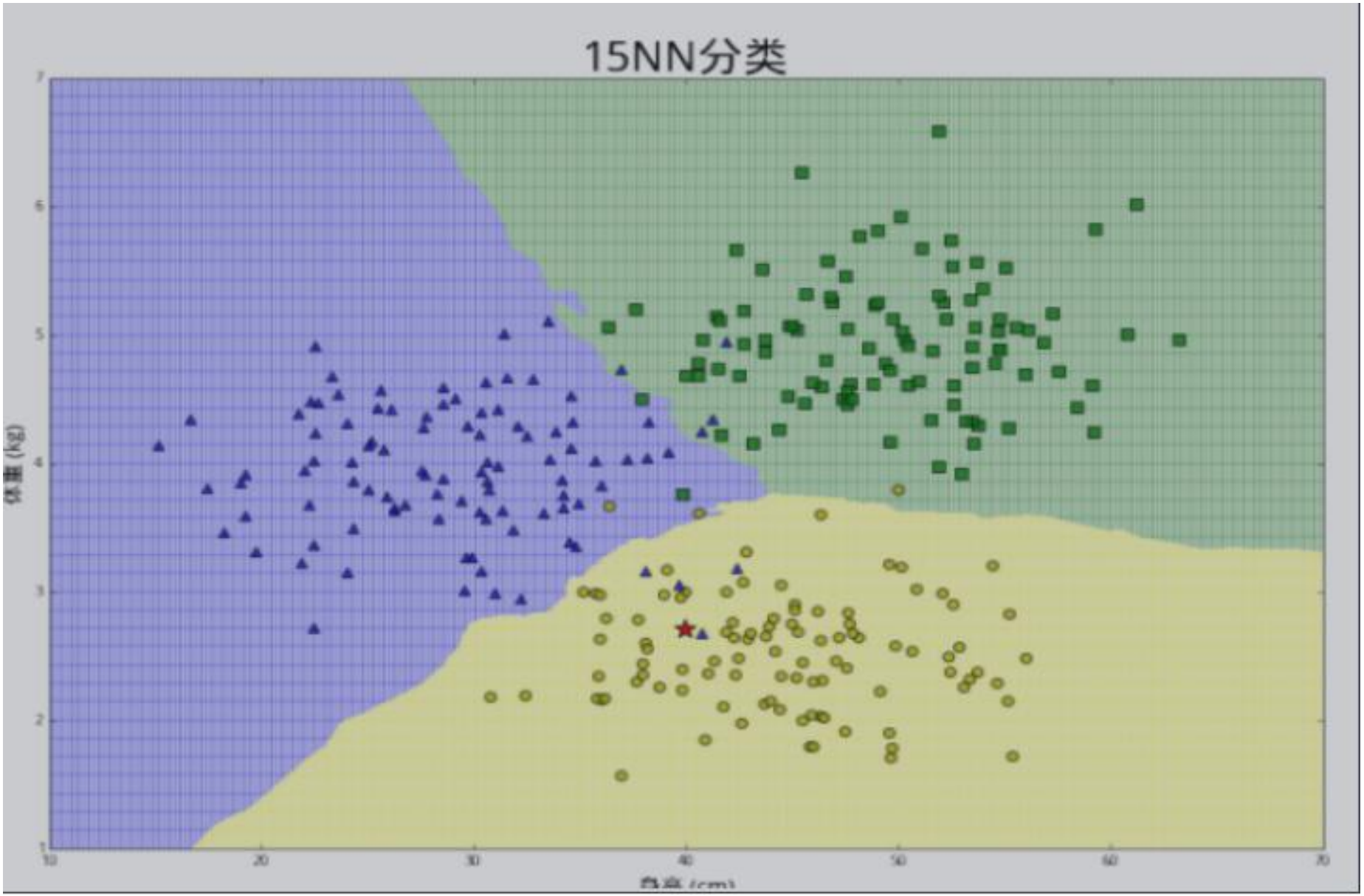


切比雪夫距离

K的选择



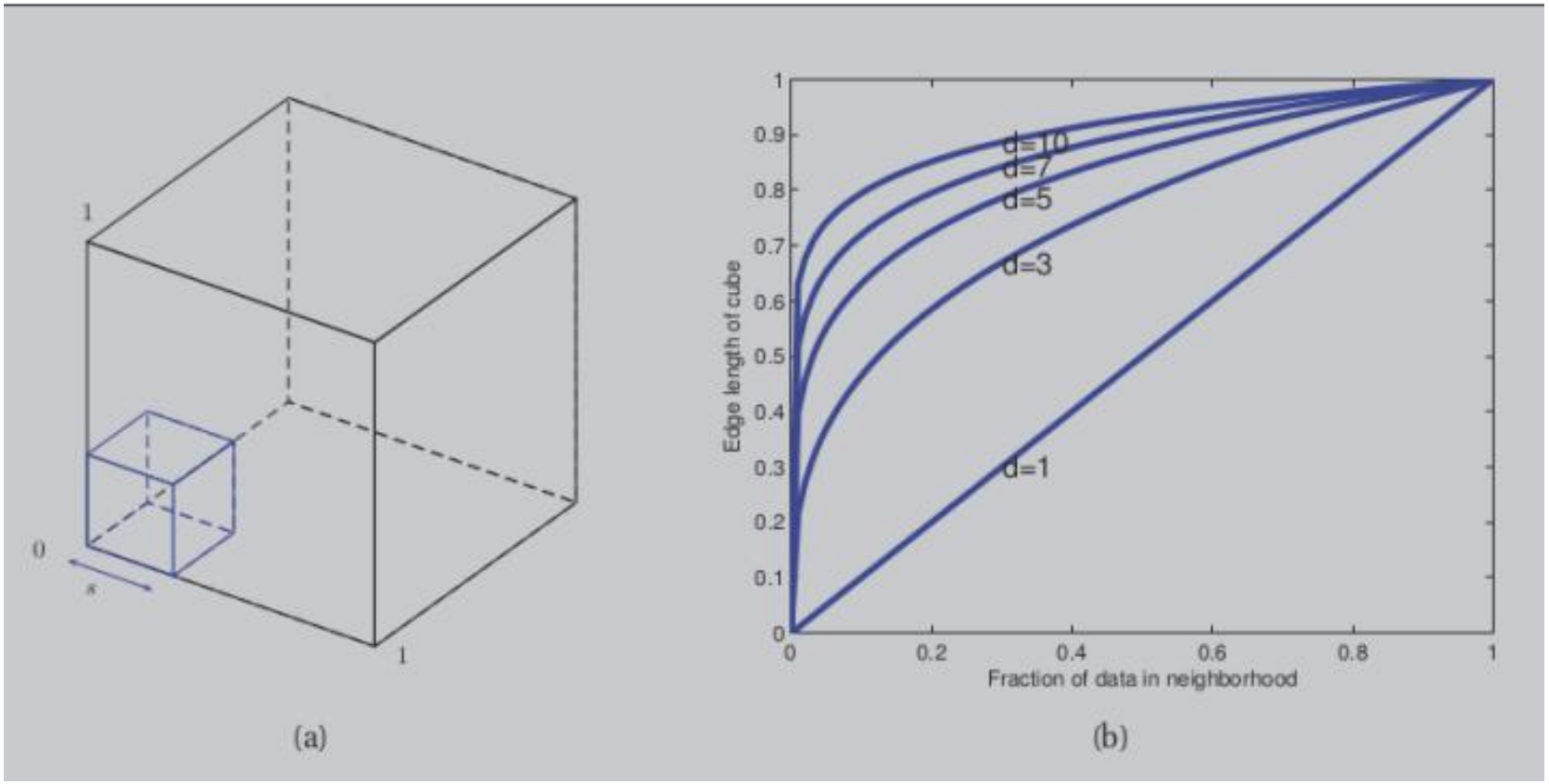
K的选择



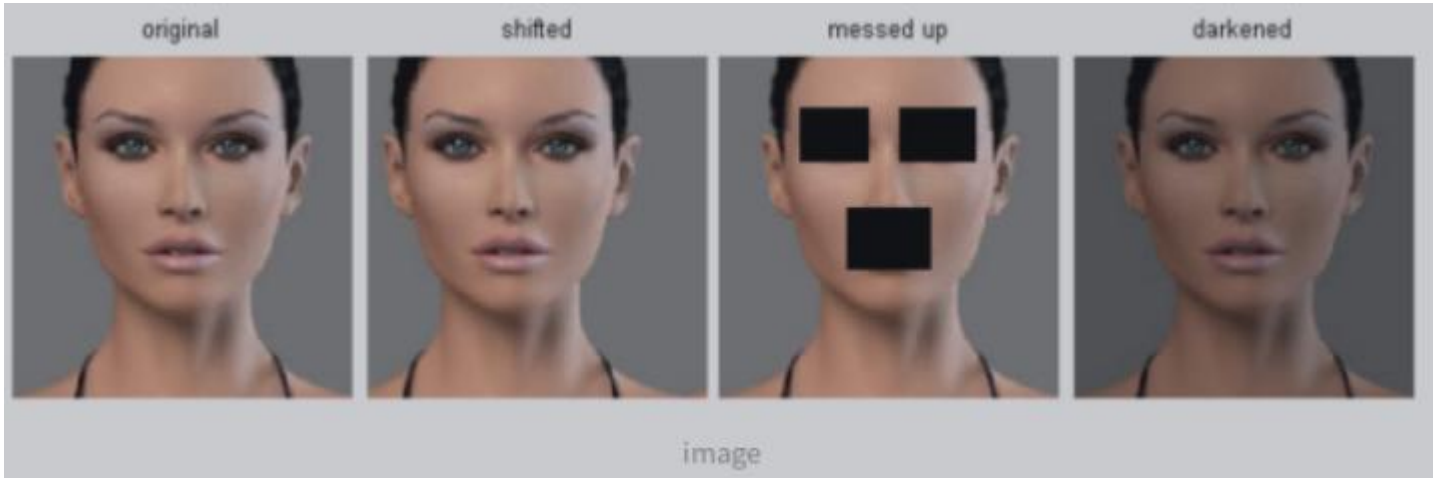
维度灾难

在高维空间中，距离将会发生一些奇妙的变化，例如我们在一个$D = 10$维的单位超立方体（边长为1）中均匀分布着一些样本点，我们想利用一个新数据点周边$f = 10$的点来判定这个点的归属类别，那么我需要囊括这个数据点周边的超立方体的边长接近为$e_d(f) = f^{1/D} = 0.8$。

此时最邻近已经失去了它的局部性，因为他所考虑的点大多都距离自己很远



比如下图中, 最左边的是原图, 右边三张图看上去差别很大, 但和原图都有着相同的欧氏距离.



避免维度灾难的方式:

PCA, 特征选择, 增加数据集等

KNN的优缺点

优点：

1.KNN分类方法是一种非参数的分类技术，简单直观，易于实现！**只要让预测点分别和训练数据求距离，挑选前k个即可，非常简单直观。**

2.KNN是一种在线技术，新数据可以直接加入数据集而**不必进行重新训练**

缺点及改进：

1.当样本不平衡时，**比如一个类的样本容量很大，其他类的样本容量很小**，输入一个样本的时候，K个邻近值大多数都是大样本容量的那个类，这时可能会导致分类错误。

改进方法：对K邻近点进行加权，也就是距离近的权值大，距离远的点权值小。

2.计算量较大，**每个待分类的样本都要计算它到全部点的距离**，根据距离排序才能求得K个临近点。

改进方法：采取kd树以及其它高级搜索方法BBF等算法减少搜索时间。

3.对维度灾难比较敏感

6. 支持向量机

支持向量机（support vector machine）是一种二类分类器，它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，这使它称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。

支持向量机学习方法包含构建由简至繁的模型：线性可分支持向量机、线性支持向量机和非线性支持向量机。简单模型是复杂模型的基础，也是复杂模型的特殊情况。

6.1 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类超平面为

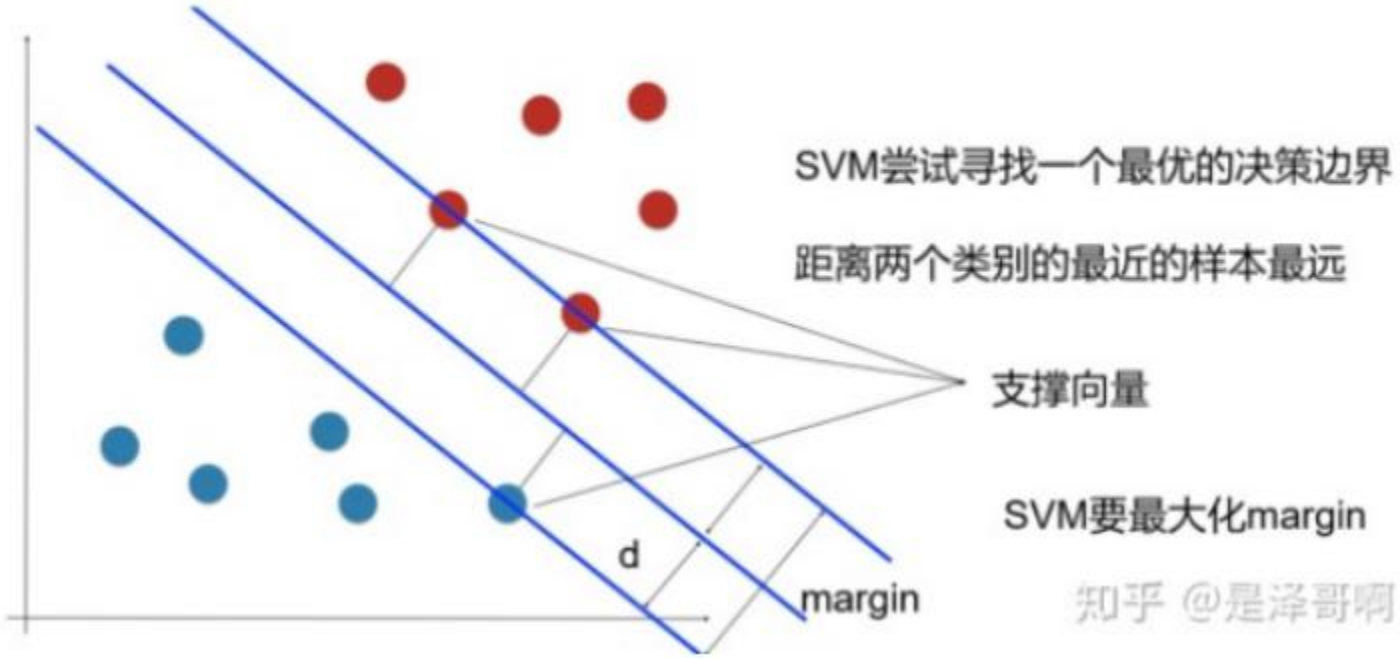
$$w^* x + b^* = 0$$

以及相应的分类决策函数

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

称为线性可分支持向量机。

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。对于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（与将要讨论的训练数据集近似线性可分时的软间隔最大化相对应）

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进行分类。也就是说，不仅将正负实例分开，而且对最难分的实例点（离超平面最近的点）也有足够大的确信度将他们分开。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。

下面考虑如何求得一个几何间隔最大的分离超平面，即最大间隔的分离超平面。具体的，这个问题可以表示为下面的约束最优化问题：

$$max_{w,b} \quad \gamma$$
$$s.t. \qquad y_i(\frac{w}{\|w\|}x_i + \frac{b}{\|w\|} \geq \gamma), \quad i = 1, 2, \ldots, N$$

即我们希望在最大化超平面(w, b)关于训练数据集的几何间隔γ，约束条件表示的是超平面(w, b)关于每个训练样本点的几何间隔至少是γ。

通过将上面的优化问题等价于如下的优化问题（具体推导可以参考《统计学习方法》李航）：

$$min_{w,b} \quad \frac{1}{2}\|w\|^2$$
$$s.t. \quad y_i(wx_i + b) \geq 1, i = 1, \ldots, N$$

这是一个凸二次规划问题。

凸优化问题是指约束最优化问题：

$$min_w \quad f(w)$$
$$s.t. \quad g_i(w) \leq 0, \quad i = 1, 2, .., k$$
$$h_i(w) = 0, \quad i = 1, 2, \ldots, l$$

其中，目标函数$f(w)$和约束函数$g_l(w)$都是R^n上的连续可微的凸函数，约束函数$h_l(w)$是R^n上的仿射函数。当目标函数$f(w)$是二次函数并且约束函数$g_l(w)$是仿射函数时，上述凸优化问题称为凸二次规划问题。

如果求出了上述约束问题的解w^*，b^*，那么就可以得到最大间隔超平面$w^*x + b^* = 0$及分类决策函数$f(x) = sign(w^*x + b^*)$，即线性可分支持向量机模型

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类超平面距离最近的样本点称为支持向量（support vector）。支持向量是使约束条件等号成立的点，即：

$$y_i(wx_i + b) - 1 = 0$$

对$y_i = 1$的正例点，支持向量在超平面

$$H_1 : wx + b = 1$$

上，对$y_i = -1$的负例点，支持向量在超平面

$$H_1 : wx + b = -1$$

上。如下图，在H_1和H_2上的点就是支持向量。

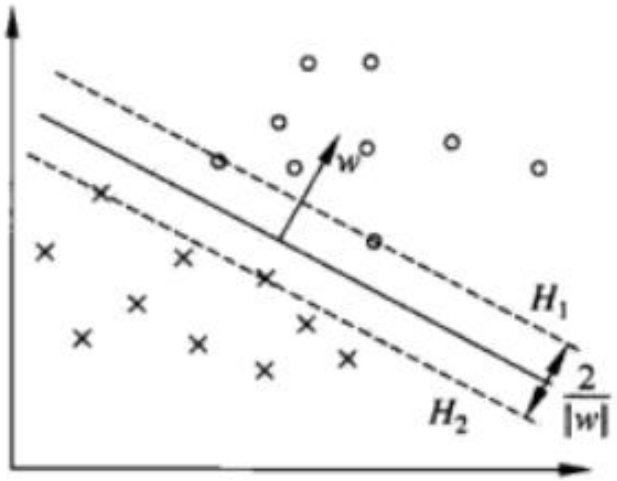


图 7.3　支持向量

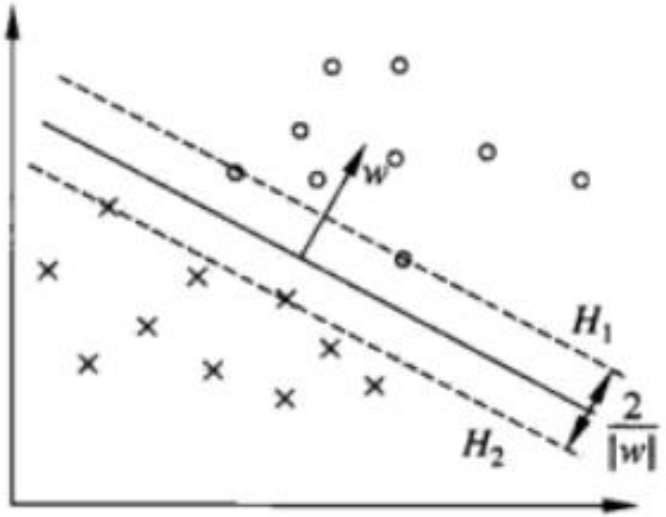


图 7.3　支持向量

注意到H_1和H_2平行，并且没有实例点落在它们中间。在H_1与H_2之间形成一条长带，分类超平面与它们平行且位于它们中央，长带的宽度，即H_1与H_2之间的距离称为间隔（margin），间隔依赖于分类超平面的法向量w，等于$\frac{2}{\|w\|}$，H_1和H_2称为间隔边界。

在决定分类超平面时只有支持向量起作用，而其它实例点并不起作用。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它的实例点，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类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。

为了求解这个优化问题，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解法。这样做的有点，一是对偶问题往往更容易求解，二是自然引入核函数，进而推广到非线性分类问题。

对偶问题的形式为（省区推导过程）：

$$\begin{aligned}\min_{\alpha} \quad & \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{N}\alpha_i \\ \text{s.t.} \quad & \sum_{i=1}^{N}\alpha_i y_i = 0 \\ & \alpha_i \geqslant 0, \quad i = 1,2,\cdots,N\end{aligned}$$

对线性可分数据集，假设对偶问题对α的解为$\alpha^* = (\alpha_1^*, \alpha_2^*, \ldots, \alpha_N^*)^T$，可以由$\alpha^*$求得原始最优化问题的解$w^*, b^*$。有下面的定理：

定理 7.2 设$\alpha^* = (\alpha_1^*, \alpha_2^*, \cdots, \alpha_l^*)^{\mathrm{T}}$是对偶最优化问题 (7.22) ～ (7.24) 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 (7.13) ～ (7.14) 的解 w^*, b^*：

$$w^* = \sum_{i=1}^{N}\alpha_i^* y_i x_i \tag{7.25}$$

$$b^* = y_j - \sum_{i=1}^{N}\alpha_i^* y_i (x_i \cdot x_j) \tag{7.26}$$

$i=1$

由次定理可知，分类超平面可以写成

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

分类决策函数可以写成

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

这就是说，分类决策函数只依赖于输入x和训练样本输入的内积，上式称为线性可分支持向量机的对偶形式。

算法 7.2（线性可分支持向量机学习算法）

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

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

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

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

$$\alpha_i\geqslant 0,\quad i=1,2,\cdots,N$$

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$.

（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

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

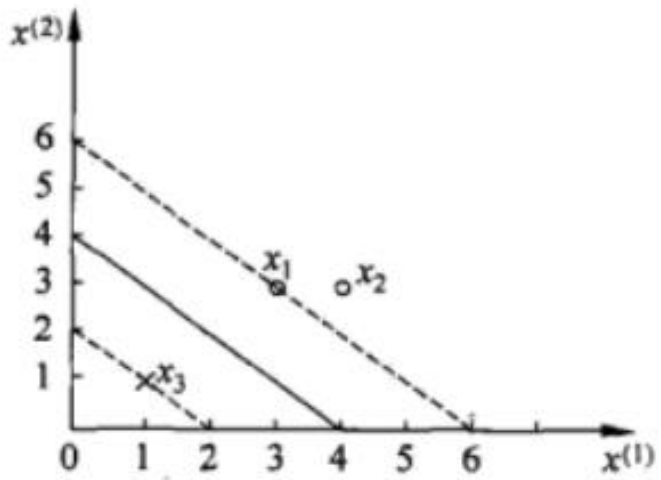
（3）求得分离超平面

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

分类决策函数：

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

例子：正例点$x_1 = (3,3)^T, x_2 = (4,3)^T$，负例点是$x_3 = (1,1)^T$。用上述算法求线性可分支持向量机



根据所给数据，对偶问题是、

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

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

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

$$\alpha_i \geqslant 0, \quad i = 1,2,3$$

解这一最优化问题。将$\alpha_3 = \alpha_1 + \alpha_2$代入目标函数得到

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

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

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

这样α_1^*, α_3^*对应的实例点x_1, x_3是支持向量，则$w_1^* = w_2^* = \frac{1}{2}$，$b^* = -2$.

分类超平面为

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

分类决策函数为

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

6.2 线性支持向量机

对于线性可分问题，上述线性可分支持向量机（硬间隔最大化）算法是完美的。但是，训练数据线性可分是理想的情形。在现实问题中，训练数据集往往是线性不可分的，即在样本中出现噪声点或者异常值。此时，我们将上述算法推广到更一般的情况

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

$$\text{s.t.} \quad y_i(w \cdot x_i + b) \geqslant 1 - \xi_i, \quad i = 1,2,\cdots,N$$

$$\xi_i \geqslant 0, \quad i = 1,2,\cdots,N$$

其对偶问题和求解算法与上述线性可分的场景类似，再次就不做展开。

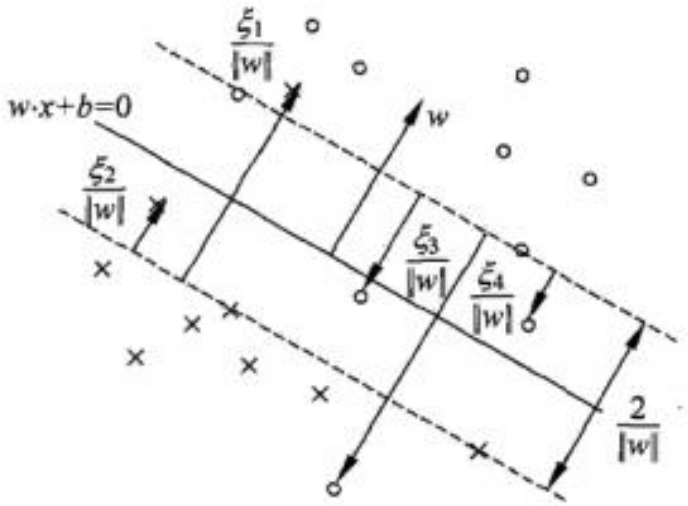
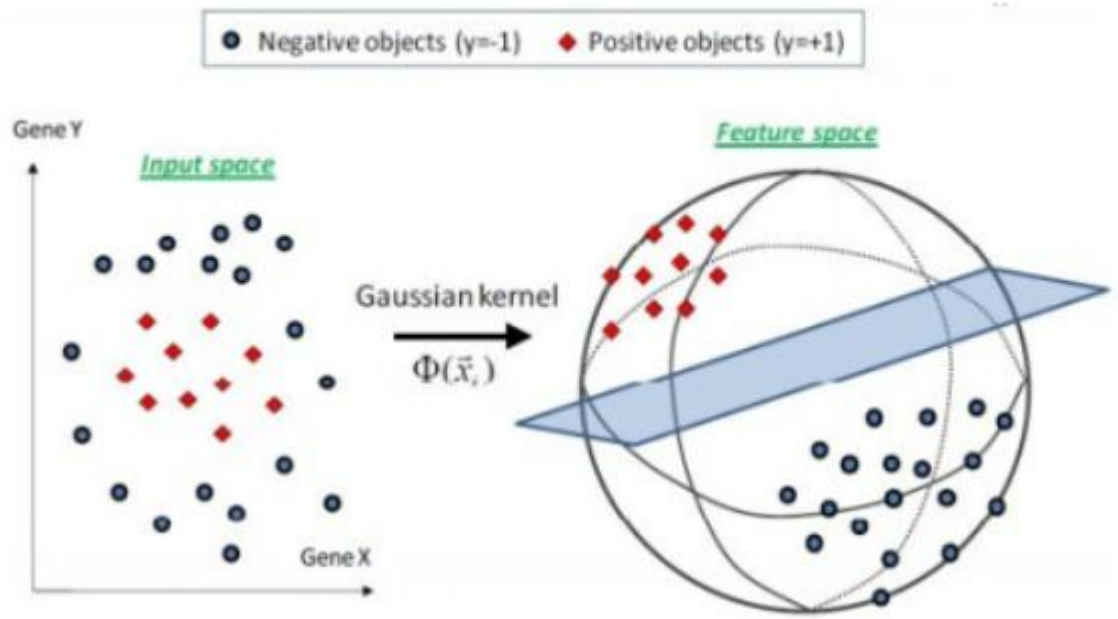
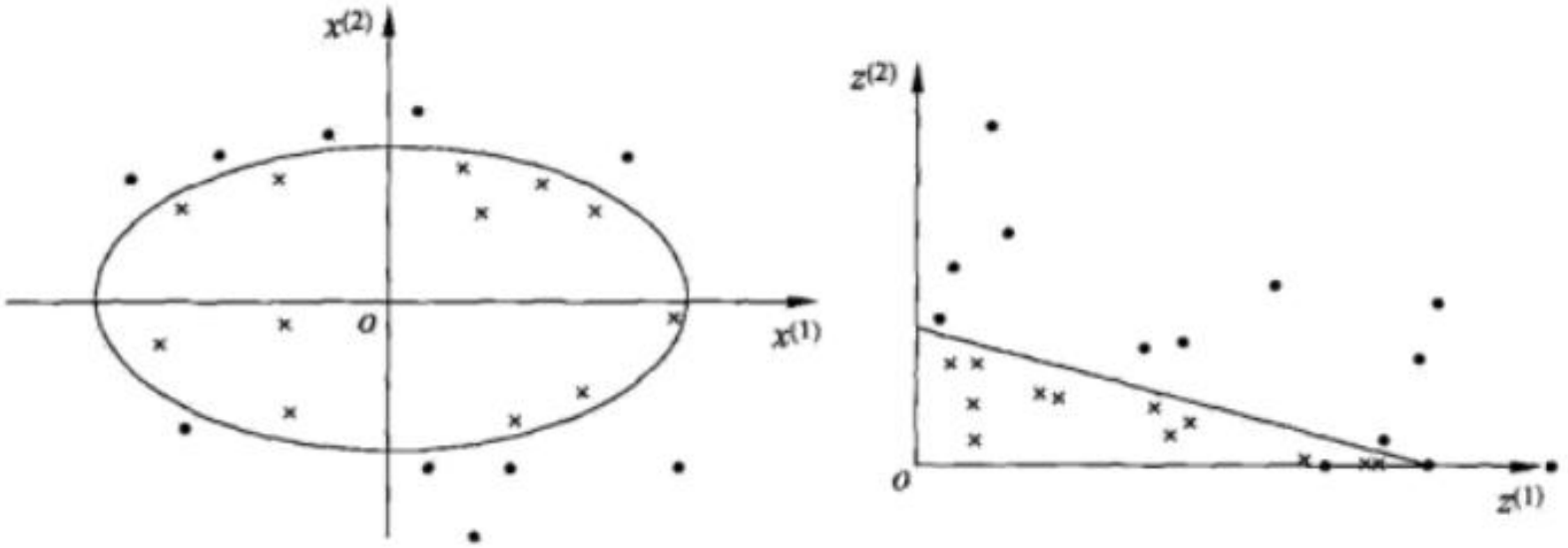


图 7.5　软间隔的支持向量

6.3 非线性支持向量机


Negative objects (y=-1)
Positive objects (y=+1)
Gene Y
Input space
Gene X
Gaussian kernel
$\Phi(\vec{x}_i)$
Feature space



非线性问题往往不好求解，所以希望能用解线性分类问题的方法解决这个问题。所采取的方法是进行一个非线性变换，将非线性问题变换为线性问题。通过解变换后的线性问题的方法求解原来的非线性问题。对应上图所示的例子，通过变换，将左图中椭圆变换成右图中的直线，将非线性分类问题变换为线性分类问题。

设原空间$x = (x^{(1)}, x^{(2)}) \in R^2$，新空间$z = (z^{(1)}, z^{(2)}) \in R^2$，定义从原空间到新空间的变换：

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

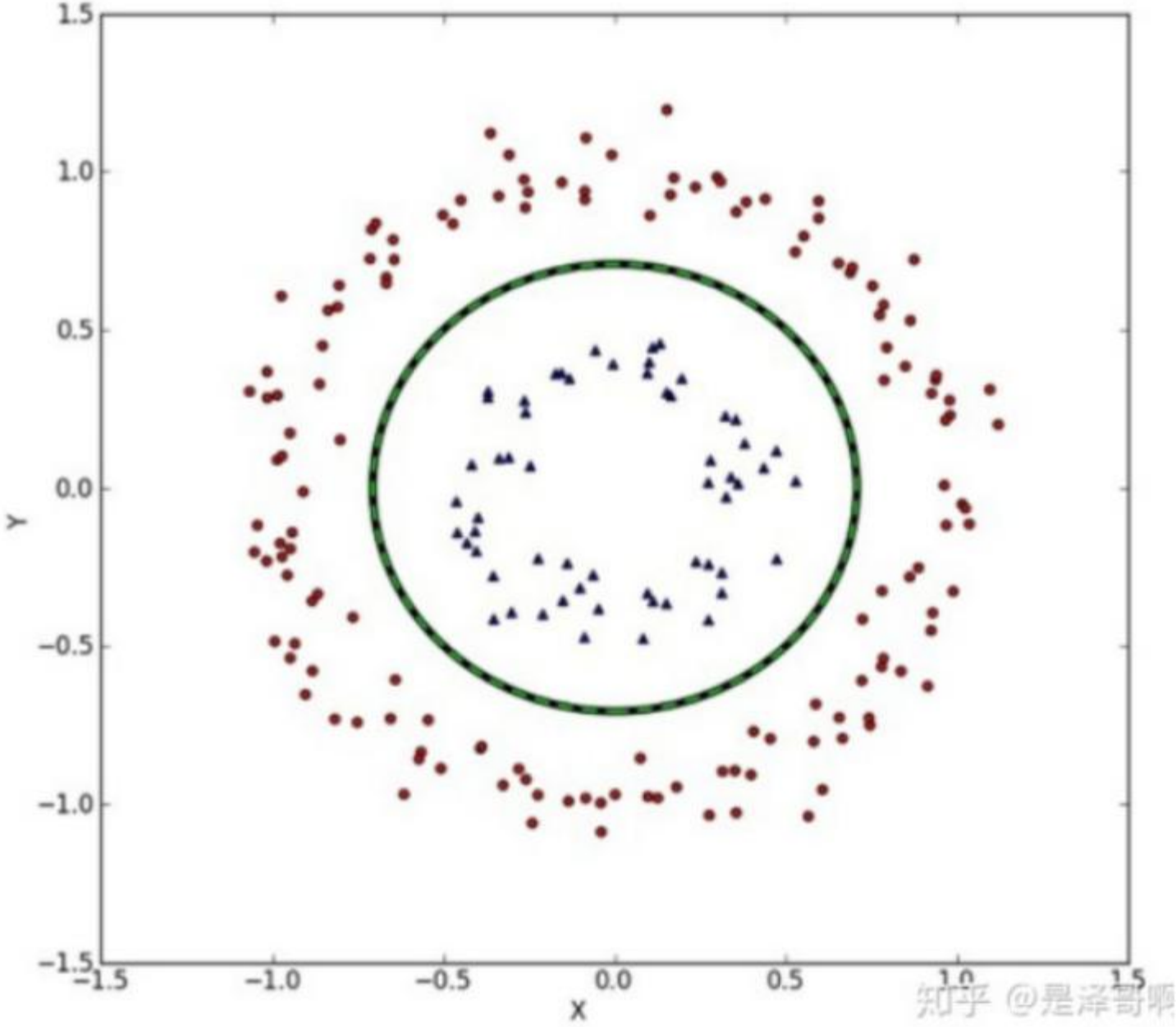
经过变换$z = \phi(x)$，原空间中的点相应地变换为新空间中的点，原空间中的椭圆

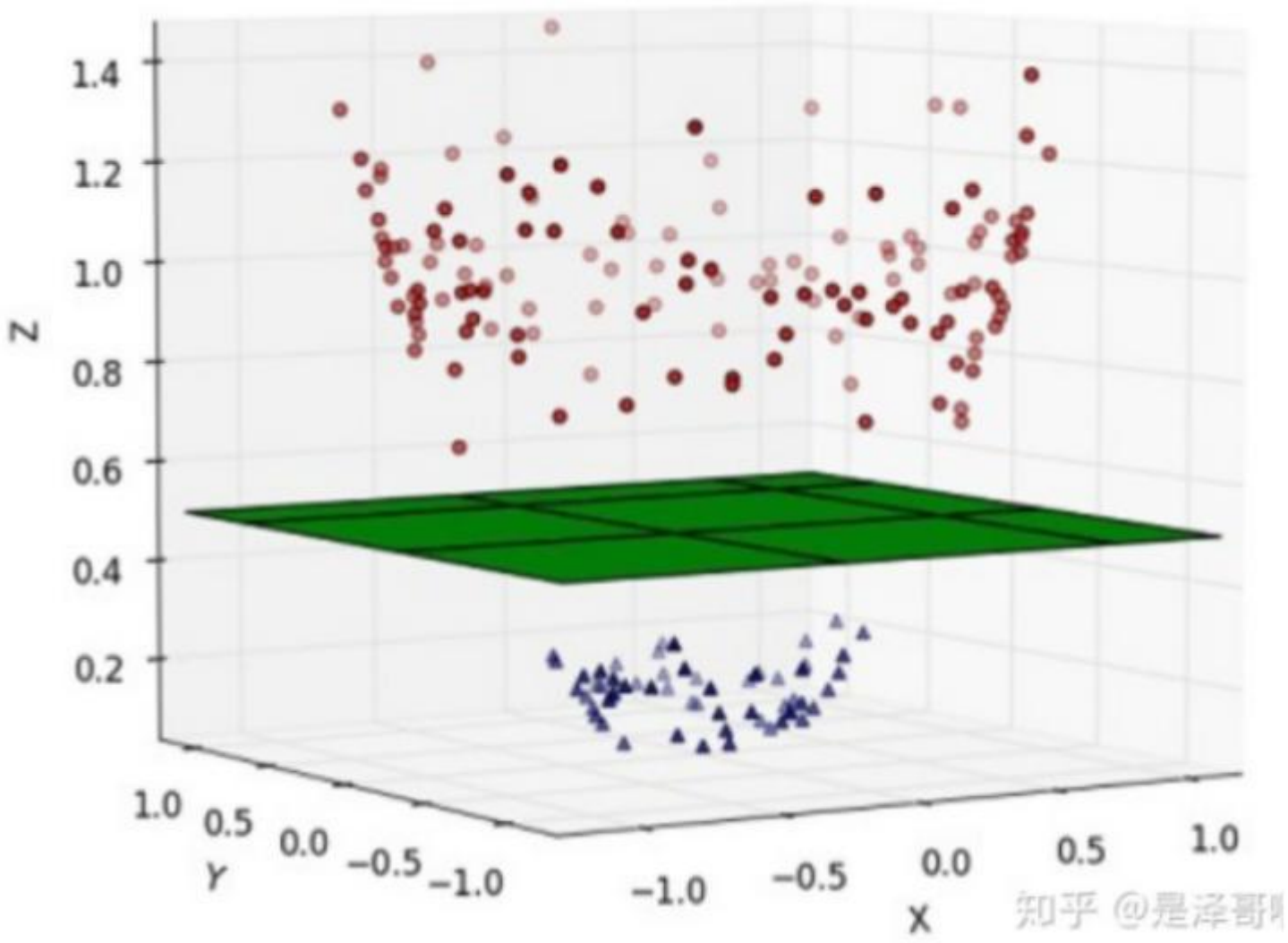
$$w_1((x^{(1)})^2) + w_2((x^{(2)})^2) + b = 0$$

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中的线性可分问题。




1.4
1.2
1.0
0.8
0.6
0.4
0.2
Z
1.0
0.5
0.0
−0.5
−1.0
Y
−1.0
−0.5
0.0
0.5
1.0
X
知乎 @是泽哥

上面的例子说明，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，其基本想法就是通过一个非线性变换将输入空间对应于一个特征空间，使得在输入空间中的超曲面模型对应于特征空间中的超平面模型。这样，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

设X是输入空间，又设H是特征空间，如果从在一个从X到H的映射

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

使得对所有的$x, z \in X$，函数$K(x, z)$满足条件

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

则称$K(x, z)$为核函数，$\phi(x)$为映射函数。

例如核函数是$K(x,z)=(x\cdot z)^2$，可以取映射

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

也可以取映射

$$\phi(x)=\frac{1}{\sqrt{2}}((x^{(1)})^2-(x^{(2)})^2,2x^{(1)}x^{(2)},(x^{(1)})^2+(x^{(2)})^2)^T$$

还可以取

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

我们注意到在支持向量机的对偶问题中，无论是目标函数还是决策函数（分类超平面）都只涉及输入实例之间的内积。在对偶问题的目标函数中的内积$x_i\cdot x_j$可以用核函数$K(x_i,x_j)=\phi(x_i)\cdot\phi(x_j)$来代替。此时对偶问题的目标函数称为:

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

同样，分类决策函数中的内积也可以用核函数代替，而分类决策函数成为

$$f(x)=\text{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i\phi(x_i)\cdot\phi(x)+b^*\right)=\text{sign}\left(\sum_{i=1}^{N_s}a_i^* y_i K(x_i,x)+b^*\right)$$

也就是说，在核函数给定的条件下，可以利用解线性分类问题的方法求解非线性分类问题的支持向量机。学习是隐式地在特征空间进行的，不需要显式的定义特征空间和映射函数。这样的技巧称为核技巧，它是巧妙的利用线性分类学习方法与核函数解决非线性问题的技术。在实际应用中，核函数的选择有很多种，有效性需要通过实验验证。

不是所有的函数都能成为核函数，有判定核函数的充要条件，这里不做展开，两个常用的核函数：

多项式核函数：

$$K(x,z)=(x\cdot z+1)^p$$

高斯核函数：

$$K(x,z)=exp(-\frac{\|x-z\|^2}{2\sigma^2})$$

核函数可以通过一些已知核函数的组合来构造，假设$K_1(x,z)$和$K_2(x,z)$是两个核函数，那么以下的函数也是核函数：

$$K(x,z)=cK_1(x,z)$$
$$K(x,z)=f(x)K_1(x,z)f(z)$$
$$K(x,z)=q(K_1(x,z))$$
$$K(x,z)=exp(K_1(x,z))$$
$$K(x,z)=K_1(x,z)+K_2(x,z)$$
$$K(x,z)=K_1(x,z)K_2(x,z)$$
$$K(x,z)=x^TAz$$

其中f是任意函数，q是一个具有非负参数的多项式，A是一个半正定对称矩阵。

非线性支持向量机的学习算法

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

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

$$\min_{\alpha}\quad \frac{1}{2}\sum_{i=1}^{N}\sum_{j=1}^{N}\alpha_i\alpha_j y_i y_j K(x_i,x_j)-\sum_{i=1}^{N}\alpha_i \tag{7.95}$$

$$\text{s.t.}\quad \sum_{i=1}^{N}\alpha_i y_i=0 \tag{7.96}$$

$$0\leqslant\alpha_i\leqslant C,\quad i=1,2,\cdots,N \tag{7.97}$$

求得最优解 $\alpha^*=(\alpha_1^*,\alpha_2^*,\cdots,\alpha_N^*)^{\mathrm{T}}$.

（2）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：

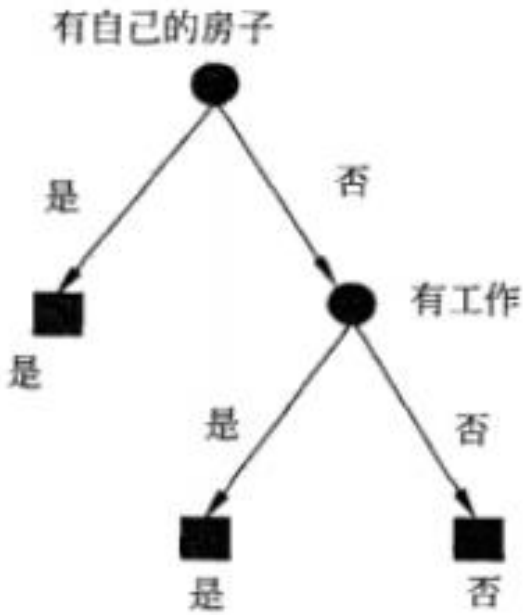
$$f(x)=\operatorname{sign}\left(\sum_{i=1}^{N}\alpha_i^* y_i K(x\cdot x_i)+b^*\right)$$ ■

当 $K(x,z)$ 是正定核函数时，问题 (7.95) ～ (7.97) 是凸二次规划问题，解是存在的.

求解支持向量机的凸二次规划问题，具有全局最优解，一般可以通过现有的工具来优化。但当训练样本非常多的时候，这些优化算法往往非常耗时低效，以致无法使用。从SVM提出到现在，也出现了很多优化训练的方法。其中，非常出名的一个是1982年由Microsoft Research的John C. Platt在论文《Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines》中提出的Sequential Minimal Optimization序列最小化优化算法，简称SMO算法。SMO算法的思想很简单，它将大优化的问题分解成多个小优化的问题。这些小问题往往比较容易求解，并且对他们进行顺序求解的结果与将他们作为整体来求解的结果完全一致。在结果完全一致的同时，SMO的求解时间短很多。具体算法在这里不做展开。

7. 决策树

决策树（decision tree）是一种基本的分类与回归方法，一般用于分类。决策树模型呈树形结构，在分类问题中，表示基于特征对实例进行分类的过程。它可以认为是if-then规则的集合，也可以认为是定义在特征空间与类空间上的条件概率分布。其主要有点是模型具有可读性，分类速度快。学习时，利用训练数据，根据损失函数最小化原则建立决策树模型。预测时，对新的数据，利用决策树模型进行分类。决策树学习通常包括3个步骤：特征选择、决策树的生成和决策树的修剪。这些决策树学习的思想主要来源于由Quinlan在1986年提出的ID3算法和1993年提出的C4.5算法，以及由Brieman等人在1984年提出的CART算法。



编号	色泽	根蒂	敲声	纹理	脐部	触感	好瓜
1	青绿	蜷缩	浊响	清晰	凹陷	硬滑	是
2	乌黑	蜷缩	沉闷	清晰	凹陷	硬滑	是
3	乌黑	蜷缩	浊响	清晰	凹陷	硬滑	是
4	青绿	蜷缩	沉闷	清晰	凹陷	硬滑	是
5	浅白	蜷缩	浊响	清晰	凹陷	硬滑	是
6	青绿	稍蜷	浊响	清晰	稍凹	软粘	是
7	乌黑	稍蜷	浊响	稍糊	稍凹	软粘	是
8	乌黑	稍蜷	浊响	清晰	稍凹	硬滑	是
9	乌黑	稍蜷	沉闷	稍糊	稍凹	硬滑	否
10	青绿	硬挺	清脆	清晰	平坦	软粘	否
11	浅白	硬挺	清脆	模糊	平坦	硬滑	否
12	浅白	蜷缩	浊响	模糊	平坦	软粘	否
13	青绿	稍蜷	浊响	稍糊	凹陷	硬滑	否
14	浅白	稍蜷	沉闷	稍糊	凹陷	硬滑	否
15	乌黑	稍蜷	浊响	清晰	稍凹	软粘	否
16	浅白	蜷缩	浊响	模糊	平坦	硬滑	否
17	青绿	蜷缩	沉闷	稍糊	稍凹	硬滑	否

7.1.1 基于信息增益的节点划分方法 ID3

信息熵

熵(entropy)是度量集合纯度的一种常用指标，计算方法为：

$$Entropy(D) = -\sum_{k=1}^{n} p_k log_2 p_k$$

其中n代表样本共有n类，p_k表示第k类样本所占的比例。

则上述好坏西瓜数据集标签的信息熵为：

注：计算熵时约定，若$p = 0$，则$plog_2 p = 0$

$Entropy(D)$的最小值是0，最大值是$log_2 n$

信息增益

对于离散特征a，有n个可能的取值，记为$\{a_1, a_2, \ldots, a_n\}$，若使用$a$对样本$D$进行划分，就会产生$n$个分支，则第$k$个分支包含了$D$中所有的在属性$a$上取值为$a_k$的样本，记为$D_k$。则用属性$a$对节点$D$进行划分所获得的信息增益(information gain):

$$Gain(D, a) = Entropy(D) - \sum_{k=1}^{n} \frac{|D^k|}{|D|} Entropy(D^k)$$

信息增益值越大，意味着用属性a来进行划分所获得的信息纯度提升越大。

用属性"色泽"对节点进行划分，可知该属性有3个可能的取值{ **青绿，　乌黑，　浅白** }，用该属性对D进行划分，可得3个分支节点：

D^1(色泽 = 青绿)包含6个样本，3个是好瓜，3个否

D^2(色泽 = 乌黑)包含6个样本，4个是好瓜，2个否

D^3(色泽 = 浅白)包含5个样本，1个是好瓜，4个否

分别计算他们的信息熵：

$$Entropy(D^1) = -(\frac{3}{6}) * log_2(\frac{3}{6}) - (\frac{3}{6}) * log_2(\frac{3}{6}) = 1.000$$

$$Entropy(D^2) = -(\frac{4}{6}) * log_2(\frac{4}{6}) - (\frac{2}{6}) * log_2(\frac{2}{6}) = 0.918$$

$$Entropy(D^3) = -(\frac{1}{5}) * log_2(\frac{1}{5}) - (\frac{4}{5}) * log_2(\frac{4}{5}) = 0.722$$

属性“色泽”的信息增益为：

$$
\begin{aligned}
Gain(D, \text{“色泽”}) &= Entropy(D) - \sum_{k=1}^{3} \frac{|D^k|}{|D|} Entropy(D^k) \\
&= 0.998 - \frac{6}{17} * 1.000 - \frac{6}{17} * 0.918 - \frac{5}{17} * 0.722 \\
&= 0.109
\end{aligned}
$$

同样的方法求出其他属性的信息增益：

$$Gain(D, \text{“根蒂”}) = 0.143$$

$$Gain(D, \text{“敲声”}) = 0.141$$

$$Gain(D, \text{“纹理”}) = 0.381$$

$$Gain(D, \text{“脐部”}) = 0.289$$

$$Gain(D, \text{“触感”}) = 0.006$$

此处，“纹理”的信息增益值较大，所以用“纹理”作为节点划分的特征。

第一步划分结束，然后再根据第一步划分出的节点一步步继续划分下去。

7.1.2 基于信息增益率的节点划分方法 C4.5

C4.5采用信息增益率作为数据分割的依据。

回到上边的例子，假如我们把数据集D中的第一列"编号"也作为一个属性，基于该属性划分节点，就会产生17个节点，我们可以求出

$$Gain(D, \text{“编号”}) = 0.998 - 0 * 7 = 0.998$$

远远大于其他的候选属性。**信息增益准则就是对可取数目较多的属性有所偏好**。

为了消除这种情况C4.5算法采用"增益率(gain ratio)"来选择最佳的划分属性。信息增益率公式：

$$Gain\_ratio(D, a) = \frac{Gain(D, a)}{IV(a)}$$

其中

$$IV(a) = -\sum_{k=1}^{n} \frac{|D^k|}{|D|} log_2 \frac{|D^k|}{|D|}$$

$IV(a)$称为属性的固有值，对于上面的数据集D，求得

$$IV(\text{色泽}) = -\frac{6}{17}log_2\frac{6}{17} - \frac{6}{17}log_2\frac{6}{17} - \frac{5}{17}log_2\frac{5}{17} = 1.580(n=3)$$

$$IV(\text{根蒂}) = -\frac{8}{17}log_2\frac{8}{17} - \frac{7}{17}log_2\frac{7}{17} - \frac{2}{17}log_2\frac{2}{17} = 1.402(n=3)$$

$$IV(\text{敲声}) = -\frac{5}{17}log_2\frac{5}{17} - \frac{2}{17}log_2\frac{2}{17} - \frac{10}{17}log_2\frac{10}{17} = 1.333(n=3)$$

$$IV(\text{纹理}) = -\frac{3}{17}log_2\frac{3}{17} - \frac{9}{17}log_2\frac{9}{17} - \frac{5}{17}log_2\frac{5}{17} = 1.447(n=3)$$

$$IV(\text{脐部}) = -\frac{7}{17}log_2\frac{7}{17} - \frac{4}{17}log_2\frac{4}{17} - \frac{6}{17}log_2\frac{6}{17} = 1.549(n=3)$$

$$IV(\text{触感}) = -\frac{5}{17}log_2\frac{5}{17} - \frac{12}{17}log_2\frac{12}{17} = 0.874(n=2)$$

$$IV(\text{编号}) = -\frac{1}{17}log_2\frac{1}{17} * 17 = 4.088(n=17)$$

得到各属性的信息增益率：

$$Gain\_ratio(D, \text{“色泽”}) = 0.069$$

$$Gain\_ratio(D, \text{“根蒂”}) = 0.102$$

$$Gain\_ratio(D, \text{“敲声”}) = 0.106$$

$$Gain\_ratio(D, \text{“纹理”}) = 0.263$$

$$Gain\_ratio(D, \text{“脐部”}) = 0.187$$

$$Gain\_ratio(D, \text{“触感”}) = 0.007$$

$$Gain\_ratio(D, \text{“编号”}) = 0.244$$

此处“纹理”的信息增益率最大，所以将该属性作为节点划分的属性。

7.1.3 CART(Classification And Regression Trees, 分类回归树)

CART既能是分类树，又能是回归树。

7.1.3.1 GINI指数构建分类树

GINI指数计算公式

$$Gini(D) = \sum_{k=1}^{n} \sum_{k' \neq k} p_k p_{k'} = 1 - \sum_{k=1}^{n} p_k^2$$

基尼指数表示了从数据集D中随机抽取两个样本，其类别标记不一致的概率，所以GINI值越小表示数据集的纯度越高。

对于数据集D的属性a的基尼指数定义为:

$$Gini\_index(D, a) = \sum_{k=1}^{n} \frac{|D^k|}{|D|} Gini(D^k)$$

在划分属性的时候，我们选择使得划分后的基尼指数最小的属性作为最优划分属性，即:

$$a_* = \underset{a \in A}{argmin}(Gini_index(D, a))$$

例，对于数据集D'

编号	色泽	脐部	好瓜
1	青绿	凹陷	是
2	乌黑	凹陷	是
3	乌黑	凹陷	是
4	青绿	凹陷	是
5	青绿	平坦	否
6	青绿	凹陷	否

用属性“色泽”划分

D'^1(色泽 = 青绿)包含4个样本，2个是好瓜，2个否

D'^2(色泽 = 乌黑)包含2个样本，2个是好瓜，0个否

$$Gini(D'^1) = 1 - (2/4)^2 - (2/4)^2 = 0.5$$

$$Gini(D'^2) = 1 - (2/2)^2 - (0)^2 = 0$$

最终，属性“色泽”的基尼指标为：

$$Gini(D', \text{色泽}) = \frac{4}{6} * 0.5 + \frac{2}{6} * 0 = 0.333$$

同样的方法可以求出：

$$Gini(D', \text{脐部}) = \frac{5}{6} * (1 - (\frac{4}{5})^2 - (\frac{1}{5})^2) + \frac{1}{6} * (1 - (\frac{1}{1})^2) = 0.267$$

属性“脐部”的基尼指标较小，所以就选它作为分裂节点。

7.1.3.2 样本方差最小值构建回归树

回归方差计算公式:

$$\sigma^2 = \sum_{i \in I}(x_i - \mu)^2 = \sum_{i \in I} x_i - n\mu^2$$

方差越大，表示该节点的数据越分散，预测的效果就越差。

$$Gain = SS_T - (SS_L + SS_R)$$

其中$SS_T = \sum_{i \in I}(x_i - \mu)^2$，$SS_L$和$SS_R$是左右节点的方差和。

例,对于数据集E

密度	含糖率
0.697	0.460
0.774	0.376
0.634	0.264
0.608	0.318
0.556	0.215
0.403	0.237
0.481	0.149
0.437	0.211
0.666	0.091

首先计算"含糖率"SS_T的方差和，然后依次选取"密度"各属性值作为分割点计算左右节点的方差和，当$SS_T - (SS_L + SS_R)$达到最大值时该点就是分割点。

7.2 分割停止

决策树节点停止分裂的一般性条件：

\1. 最小节点数

当节点的数据量小于一个指定的数量时，不继续分裂。有两点原因：一是数据量较少时，再做分裂容易强化噪声数据的作用；二是降低树生长的复杂性。提前结束分裂一定程度上有利于降低过拟合的影响。

\2. 熵或者基尼值小于阀值

由上述可知，熵和基尼值的大小表示数据的复杂程度，当熵或者基尼值过小时，表示数据的纯度比较大，如果熵或者基尼值小于一定程度数，节点停止分裂。

\3. 决策树的深度达到指定的条件

节点的深度可以理解为节点与决策树跟节点的距离，如根节点的子节点的深度为1，因为这些节点与跟节点的距离为1，子节点的深度要比父节点的深度大1。决策树的深度是所有叶子节点的最大深度，当深度到达指定的上限大小时，停止分裂。

\4. 所有特征已经使用完毕，不能继续进行分裂

被动式停止分裂的条件，当已经没有可分的属性时，直接将当前节点设置为叶子节点。