AI300学院

机器学习之监督学习

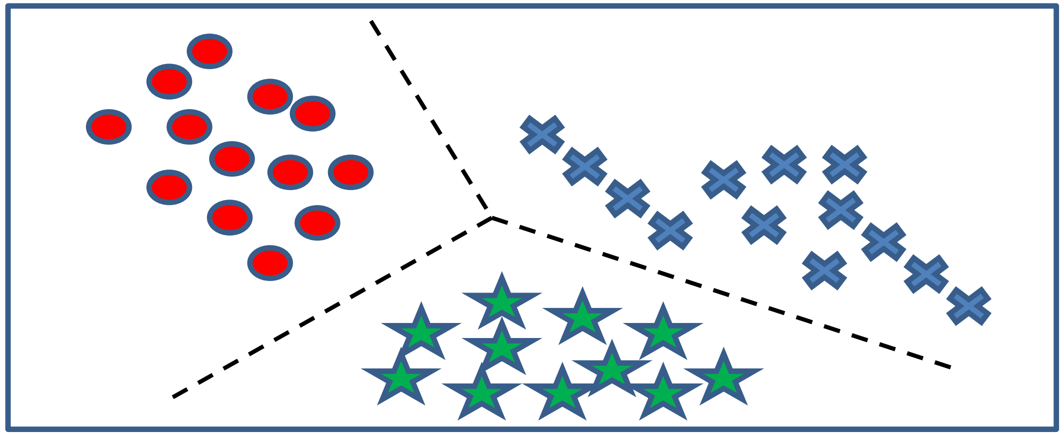
支持向量机(SVM)

倪冰冰
上海交通大学

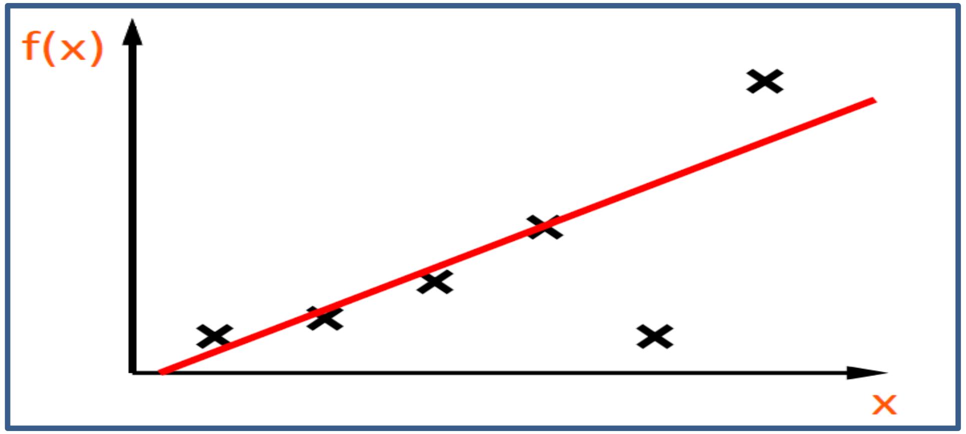
监督学习

- 给定一组数据，我们知道正确的输出结果应该是什么样子，并且知道在输入和输出之间有着一个特定的关系$f(x)$。

- 分类 vs 回归



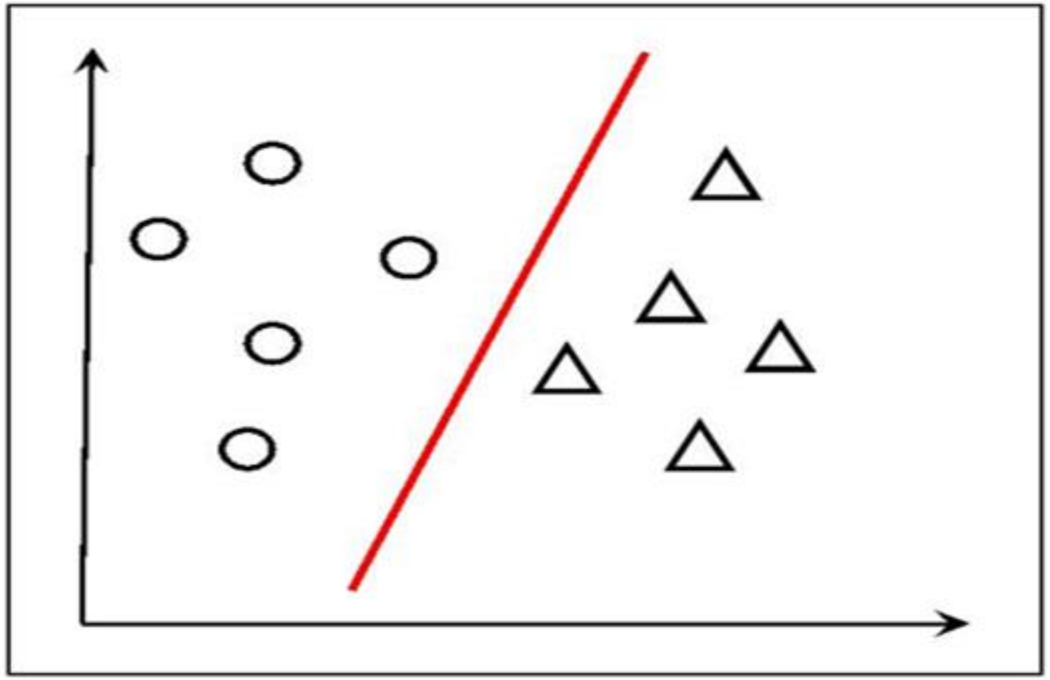
分类(Classification)



回归(Regression)

线性分类模型

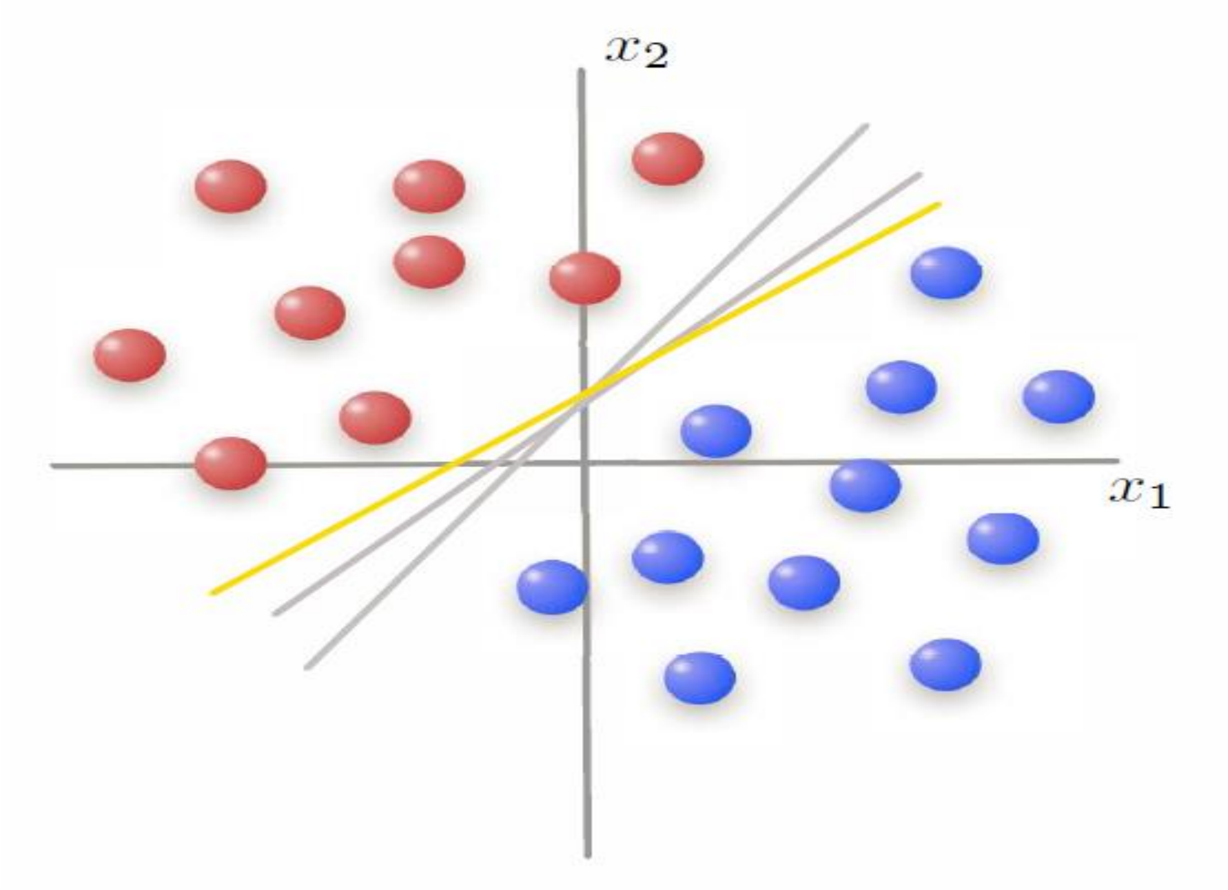
线性分类器



思路一：用超平面（线）将不同类样本分开

AI300学院

Support Vector Machine



Is there an "optimal" way to separate the data?

这么多分类界面，哪个是最优的呢？

Support Vector Machine

Invented by Vladimir Vapnik and
co-workers at AT&T Bell Labs in 1990s

Vladimir Vapnik
Professor
The Computer Learning Research Centre
University of London

线性分类器的巅峰-支持向量机（Support Vector Machine）

Support Vector Machine

Basic problem

Given N training data points:

$$\mathbf{x}_1, \ldots, \mathbf{x}_i, \ldots, \mathbf{x}_N \in R^m$$

each with a label of

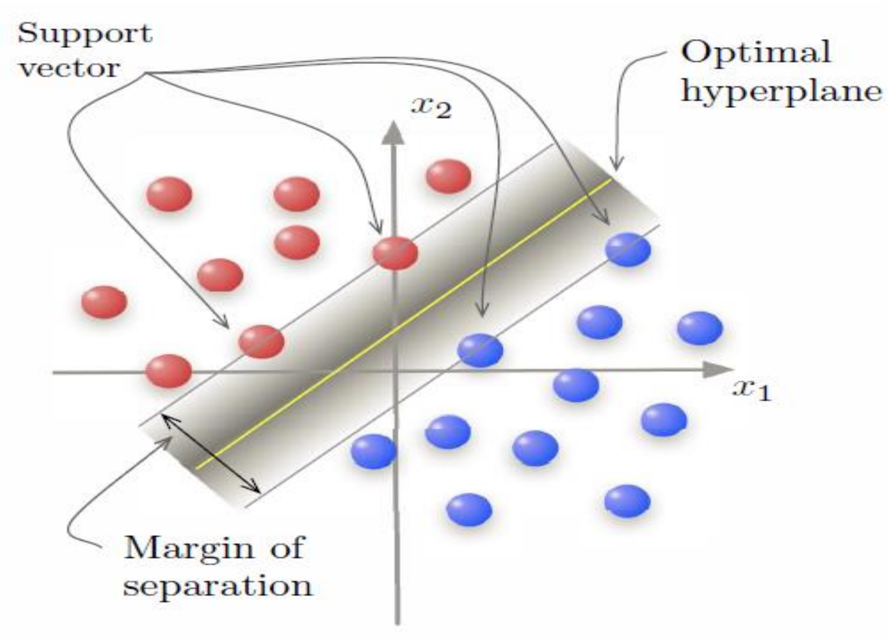
$$d_i = +1 \quad \text{or} \quad d_i = -1$$

find a hyperplane

$$\mathbf{w}^T\mathbf{x} + b = 0$$
$$(\mathbf{w} \in R^m,\ b \in R)$$

that separates data into two groups:

- those with $d_i = +1$
- those with $d_i = -1$



Hyperplane is optimal when margin is maximized

要将两边的“缓冲区”（margin）最大化！！！

Support Vector Machine

AI300学院

训练目的

Point $\mathbf{x}_i$ is classified by hyperplane $\mathbf{w}^T\mathbf{x} + b = 0$ as follows

$$\begin{cases} \mathbf{w}^T\mathbf{x}_i + b \geq 0 & \text{for } d_i = +1 \\ \mathbf{w}^T\mathbf{x}_i + b < 0 & \text{for } d_i = -1 \end{cases}$$

d_i为标签

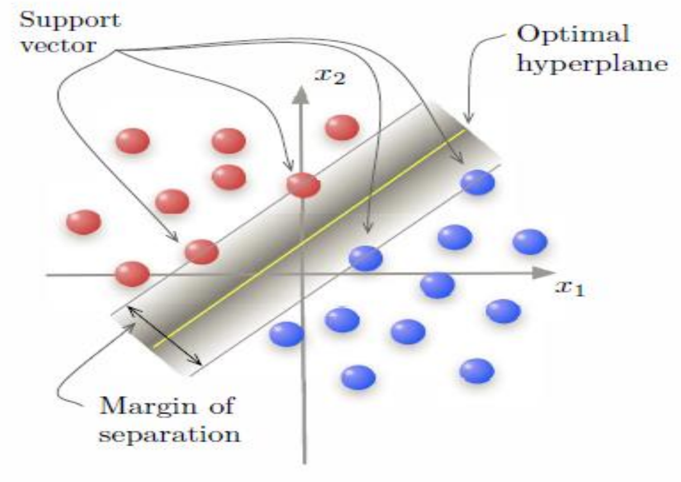
We want to find $\mathbf{w}_o$ and b_o that define the optimal hyperplane

线性判别函数： $g(\mathbf{x}) = \mathbf{w}_o^T\mathbf{x} + b_o = 0$

▸ Back

so that for training data $\mathbf{x}_i$

$$\begin{cases} g(\mathbf{x}_i) = \mathbf{w}_o^T\mathbf{x}_i + b_o \geq +1 & \text{for } d_i = +1 \\ g(\mathbf{x}_i) = \mathbf{w}_o^T\mathbf{x}_i + b_o \leq -1 & \text{for } d_i = -1 \end{cases}$$



or, in a compact form:

$$d_i\, g(\mathbf{x}_i) = d_i\left(\mathbf{w}_o^T\mathbf{x}_i + b_o\right) \geq 1$$

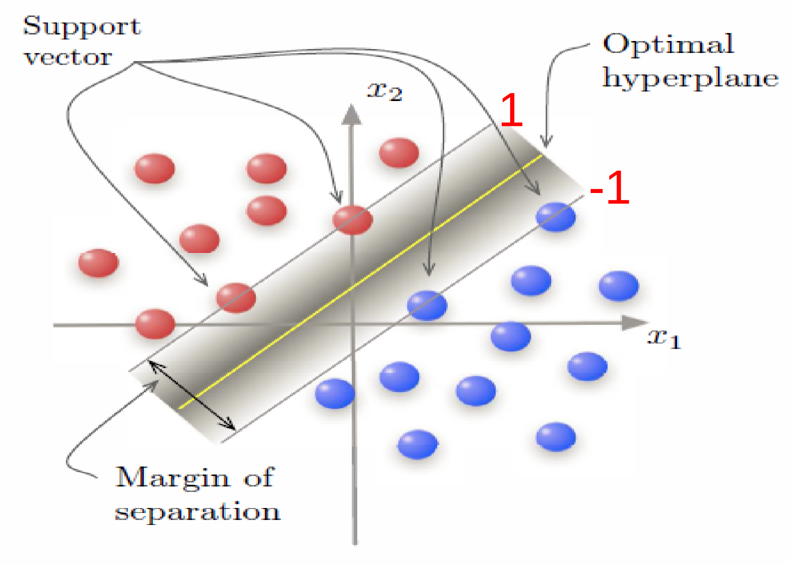
两种情况可以合写在一起

AI300学院

Support Vector Machine

- For training data $\mathbf{x}_i$

$$\begin{cases} g(\mathbf{x}_i) = \mathbf{w}_o^T \mathbf{x}_i + b_o \geq +1 & \text{for } d_i = +1 \\ g(\mathbf{x}_i) = \mathbf{w}_o^T \mathbf{x}_i + b_o \leq -1 & \text{for } d_i = -1 \end{cases}$$



Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_o^T \mathbf{x} + b_o$$

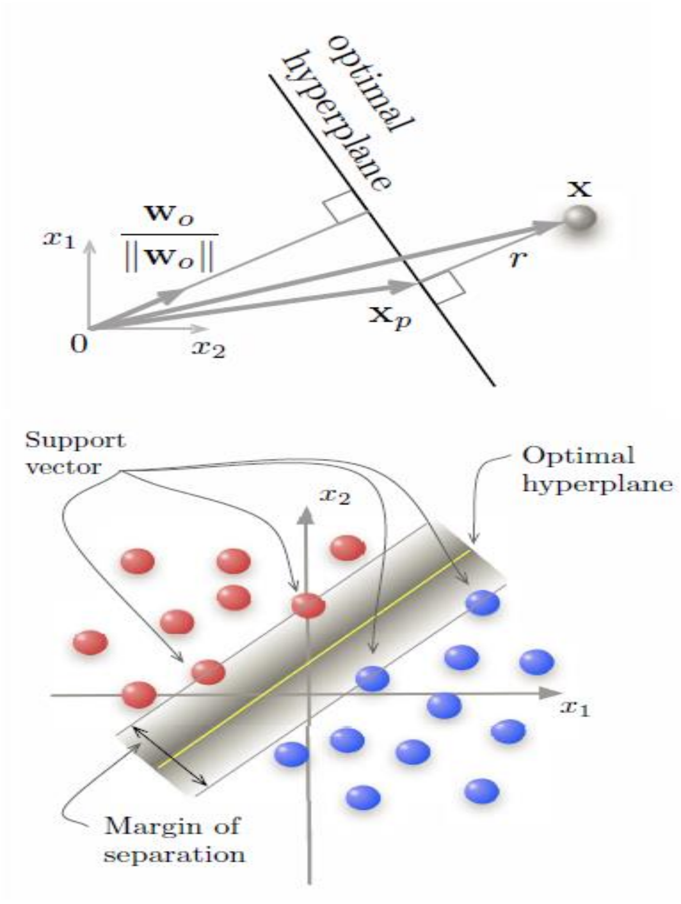
Support vector: $\mathbf{x}_i$ that satisfies

$$g(\mathbf{x}_i) = \pm 1$$

支持向量：即margin的边界上的点

Support Vector Machine

Optimal hyperplane is determined by minimizing $\mathbf{w}$



Key arguments:

- Margin of separation ρ is proportional to r

$$\rho \propto r$$

- r is inversely proportional to $\|\mathbf{w}\|$

$$r \propto \frac{1}{\|\mathbf{w}\|}$$

Margin反比于平面系数w

- Maximizing margin implies minimizing $\mathbf{w}$

$$\text{Max } \rho \rightarrow \text{Min } \|\mathbf{w}\|$$

Support Vector Machine

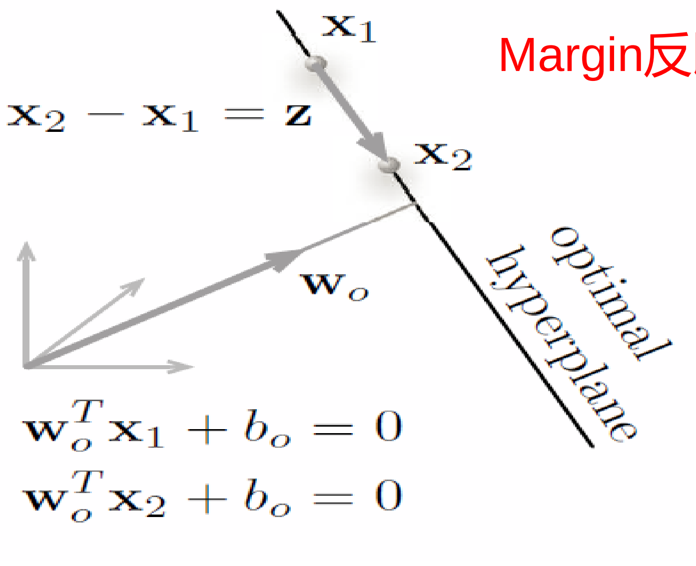
Optimal hyperplane: $\mathbf{w}_o^T \mathbf{x} + b_o = 0$

Claim: $\mathbf{w}_o \perp$ optimal hyperplane

Proof: Given vectors $\mathbf{u}$ and $\mathbf{v}$, we have the facts

1. if $\mathbf{u} \cdot \mathbf{v} = 0$ then $\mathbf{u} \perp \mathbf{v}$
2. $\mathbf{u} \cdot \mathbf{v} = \mathbf{u}^T \mathbf{v}$

Consider a vector $\mathbf{z}$ defined by two points $\mathbf{x}_1$ and $\mathbf{x}_2$ on hyperplane



Margin反比于平面系数$\boldsymbol{w}$

$$\begin{aligned} \mathbf{w}_o \cdot \mathbf{z} &= \mathbf{w}_o \cdot (\mathbf{x}_2 - \mathbf{x}_1) = \mathbf{w}_o \cdot \mathbf{x}_2 - \mathbf{w}_o \cdot \mathbf{x}_1 \\ &= \mathbf{w}_o^T \mathbf{x}_2 - \mathbf{w}_o^T \mathbf{x}_1 = -b_o - (-b_o) = 0 \end{aligned}$$

Support Vector Machine

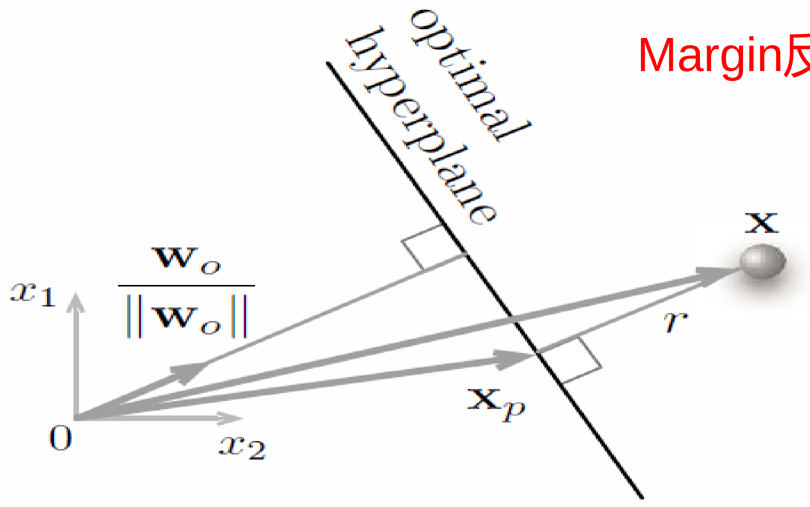
$$\mathbf{x} = \mathbf{x}_p + r\left(\frac{\mathbf{w}_o}{||\mathbf{w}_o||}\right)$$

$\mathbf{x}_p$ is on optimal hyperplane:

$g(\mathbf{x}_p) = \mathbf{w}_o^T\mathbf{x}_p + b_o = 0$ ▸ View

So

$$\mathbf{w}_o^T\mathbf{x}_p = -b_o$$



Margin反比于平面系数*w*

$$\begin{aligned} g(\mathbf{x}) &= \mathbf{w}_o^T\mathbf{x} + b_o = \mathbf{w}_o^T\left(\mathbf{x}_p + r\left(\frac{\mathbf{w}_o}{||\mathbf{w}_o||}\right)\right) + b_o \\ &= \mathbf{w}_o^T\mathbf{x}_p + r\left(\frac{\mathbf{w}_o^T\mathbf{w}_o}{||\mathbf{w}_o||}\right) + b_o = -b_o + r\left(\frac{||\mathbf{w}_o||^2}{||\mathbf{w}_o||}\right) + b_o \\ &= r||\mathbf{w}_o|| \end{aligned}$$

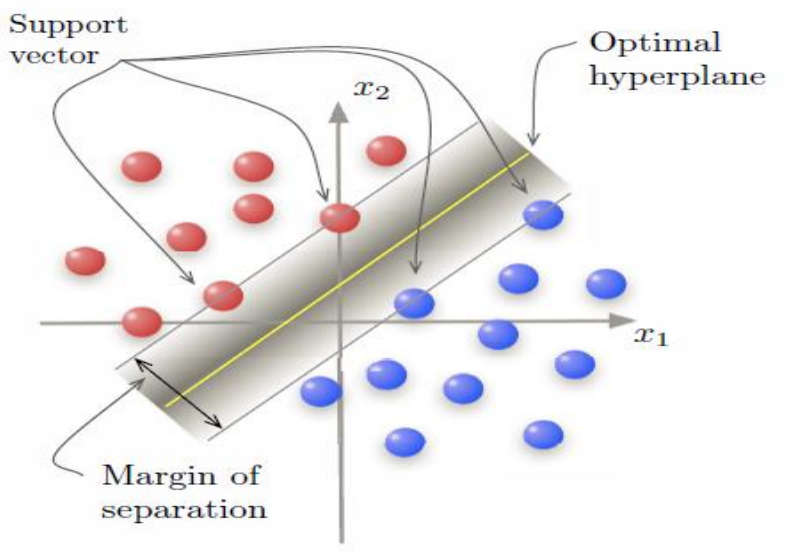
Support Vector Machine

For a support vector $\mathbf{x}^{(s)}$, $g\left(\mathbf{x}^{(s)}\right) = \pm 1$. So

$$r = \frac{g\left(\mathbf{x}^{(s)}\right)}{\|\mathbf{w}_o\|} = \begin{cases} \dfrac{1}{\|\mathbf{w}_o\|} & \text{if } d^{(s)} = +1 \\ -\dfrac{1}{\|\mathbf{w}_o\|} & \text{if } d^{(s)} = -1 \end{cases}$$

- Margin of separation

$$\rho = 2r = \frac{2}{\|\mathbf{w}_o\|}$$



Support Vector Machine

Primal problem 支持向量机的原问题建模

Given data set : $\{(\mathbf{x}_i, d_i)\}$, $i = 1, 2, \ldots, N$

Find : $\mathbf{w}$ and b

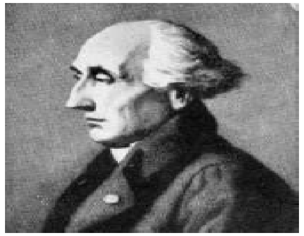
Minimizing : $\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to : $d_i\,(\mathbf{w}^T\mathbf{x}_i + b) \geq 1$

Well known algorithms exist for solving this problem

- R., Fletcher, *Practical Methods of Optimization*. 2nd ed., John Wiley & Sons, NY, 1987
- Arnold Neumaier, Complete search in continuous global optimization and constraint satisfaction. *Acta Numerica*, Vol. 13, 271-369, 2004

Alternative formulation using method of Lagrange multipliers



Joseph-Louis Lagrange
French mathematician
1736–1813

AI300学院

Support Vector Machine

Optimization problem: 约束优化经典方法：拉格朗日乘子

$$\boxed{\begin{array}{rl} \text{Maximize :} & f(x,y) \\ \text{Subject to :} & g(x,y) = c \end{array}}$$

$$\begin{array}{rl} \text{Find :} & \mathbf{w} \text{ and } b \\ \text{Minimizing :} & \Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \\ \text{Subject to :} & d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 \end{array}$$

Lagrangian function:

$$J(x,y,\alpha) = \underbrace{f(x,y)}_{\text{to optimize}} - \alpha \underbrace{\left(g(x,y) - c\right)}_{\text{from constraint}}$$

Solution must satisfy Karush-Kuhn-Tucker conditions:

$$\begin{array}{rcl} \frac{\partial J}{\partial w} & = & 0 \\ \frac{\partial J}{\partial b} & = & 0 \end{array} \qquad \begin{array}{rcl} g(x) & \geq & 0 \\ \alpha & \geq & 0 \\ \alpha g(x) & = & 0 \end{array}$$

KKT条件-约束优化的经典工具！

Support Vector Machine

Primal problem with Lagrange multipliers:

$$J(\mathbf{w}, b, \alpha) = \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1\right)$$

Karush-Kuhn-Tucker conditions

$$\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial \mathbf{w}} = \mathbf{0} \Rightarrow \mathbf{w} = \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{x}_i$$

$$\frac{\partial J(\mathbf{w}, b, \alpha)}{\partial b} = 0 \Rightarrow \sum_{i=1}^{N} \alpha_i \, d_i = 0$$

$$d_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$$

$$\alpha_i \left(d_i \left(\mathbf{w}^T\mathbf{x}_i + b\right) - 1\right) = 0$$

$$\alpha_i \geq 0$$

where

$$\frac{\partial(\cdot)}{\partial \mathbf{w}} = \begin{bmatrix} \frac{\partial(\cdot)}{\partial w_1} \\ \frac{\partial(\cdot)}{\partial w_2} \\ \vdots \\ \frac{\partial(\cdot)}{\partial w_m} \end{bmatrix}$$

Support Vector Machine

$$
\begin{aligned}
J(\mathbf{w}, b, \alpha) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right) \\
&= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \mathbf{x}_i - b \sum_{i=1}^{N} \alpha_i \, d_i + \sum_{i=1}^{N} \alpha_i
\end{aligned}
$$

From the KKT conditions, we have

$$
\mathbf{w} = \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{x}_i, \quad \sum_{i=1}^{N} \alpha_i \, d_i = 0
$$

So $\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \mathbf{x}_i = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T \mathbf{x}_j$

Let $Q(\alpha) \equiv J(\mathbf{w}, b, \alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i=j}^{N} \alpha_i \, \alpha_j \, d_i \, d_j \, \mathbf{x}_i^T \mathbf{x}_j$

AI300学院

Support Vector Machine

Dual problem 支持向量机的对偶问题建模

Given : $\{(\mathbf{x}_i, d_i)\}, \quad i \in \{1, \ldots, N\}$

Find : Lagrange multipliers $\{\alpha_i\}$

Maximizing : $Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i=j}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$

Subject to : (1) $\sum_{i=1}^{N} \alpha_i d_i = 0$

(2) $\alpha_i \geq 0$

- α_i are the only unknowns
- Algorithms and software available for solving this problem

简单二次规划问题，有现成的工具包可以简单求解，主要基于SMO算法!

Support Vector Machine

KKT condition:

$$\alpha_i \left(d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) - 1 \right) = 0$$

- For data point $\mathbf{x}_i$ that is not a support vector

$$d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) > 1 \quad \Rightarrow \quad \alpha_i = 0$$

- For data point $\mathbf{x}_i$ that is a support vector

$$d_i \left(\mathbf{w}^T \mathbf{x}_i + b \right) = 1 \quad \Rightarrow \quad \alpha_i \neq 0$$

如果$\alpha_i > 0$说明是支持向量!

AI300学院

Support Vector Machine

	Primal	Dual
Find :	$\mathbf{w}\,,\,b$	α_i
Minimizing :	$\Phi(\mathbf{w})$	$\mathbf{w}$
Maximizing :	$-$	$Q(\alpha)$
Subject to :	$d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$	$\sum_{i=1}^{N} \alpha_i d_i = 0$
		$\alpha_i \geq 0$

$$\Phi\left(\mathbf{w},\xi\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} \qquad Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i=j}^{N}\alpha_i\,\alpha_j\,d_i\,d_j\,\mathbf{x}_i^T\mathbf{x}_j$$

After $\alpha_{o,i}$ is obtained, we can calculate $\mathbf{w}_o$ and b_o as follows:

$$\mathbf{w}_o = \sum_{i=1}^{N}\alpha_{o,i}\,d_i\,\mathbf{x}_i\,, \quad b_o = 1 - \mathbf{w}_o^T\mathbf{x}^{(s)}$$

where $\mathbf{x}^{(s)}$ is a support vector with $d^{(s)} = +1$

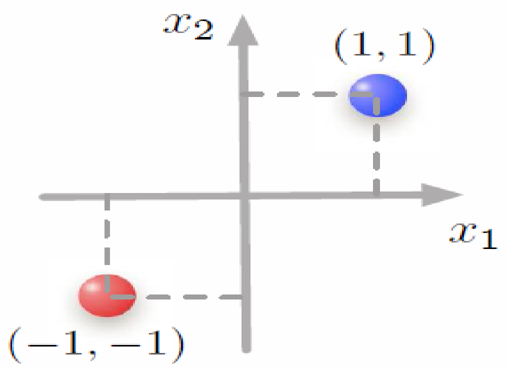
AI300学院

Support Vector Machine

举个例子

$$\mathbf{x}_i = \begin{bmatrix} x_1 \\ x_2 \end{bmatrix}, \quad \mathbf{x}_1 = \begin{bmatrix} -1 \\ -1 \end{bmatrix}, \quad \mathbf{x}_2 = \begin{bmatrix} 1 \\ 1 \end{bmatrix}$$

$$d_1 = -1, \qquad d_2 = +1$$



$$\begin{aligned}
Q(\alpha) &= \sum_{i=1}^{2} \alpha_i - \frac{1}{2}\left(\sum_{i=1}^{2}\sum_{i=1}^{2} \alpha_i \alpha_j d_i d_j \mathbf{x}^T \mathbf{x}\right) \\
&= \alpha_1 + \alpha_2 - \frac{1}{2}\Bigg(\alpha_1\alpha_1(-1)(-1)[-1 \ \ -1]\begin{bmatrix} -1 \\ -1 \end{bmatrix} + \\
&\quad \alpha_1\alpha_2(-1)(1)[-1 \ \ -1]\begin{bmatrix} 1 \\ 1 \end{bmatrix} + \cdots\Bigg) \\
&= \alpha_1 + \alpha_2 - \frac{1}{2}\left(2\alpha_1^2 + 4\alpha_1\alpha_2 + 2\alpha_2^2\right) \\
&= \alpha_1 + \alpha_2 - (\alpha_1 + \alpha_2)^2
\end{aligned}$$

AI300学院

Support Vector Machine

举个例子 Optimization problem

$$\text{Maximizing :} \quad Q(\alpha) = \alpha_1 + \alpha_2 - (\alpha_1 + \alpha_2)^2$$
$$\text{Subject to :} \quad -\alpha_1 + \alpha_2 = 0$$
$$\alpha_1 \geq 0\,, \quad \alpha_2 \geq 0$$

To find α_i manually for this simple example

$\boxed{1}$ $\dfrac{\partial Q(\alpha)}{\partial \alpha_1} = \dfrac{\partial Q(\alpha)}{\partial \alpha_2} = 0$

$\boxed{2}$ $\sum_{i=1}^{2} \alpha_i d_i = -\alpha_1 + \alpha_2 = 0$

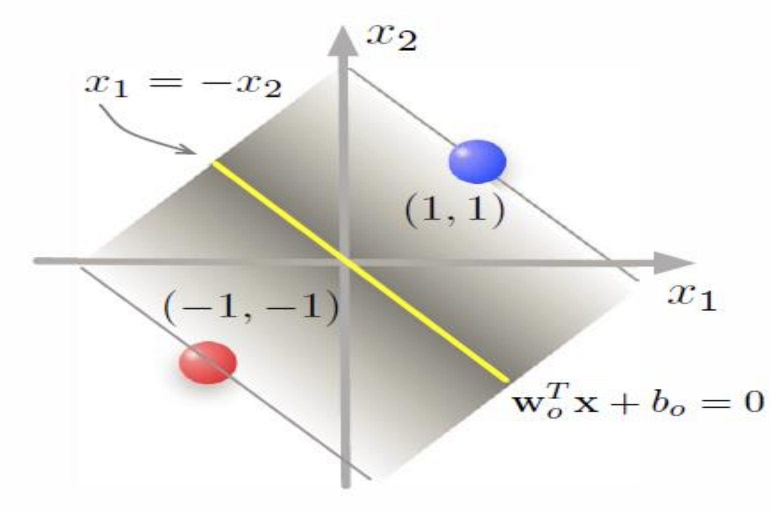
$$\left.\begin{aligned} \frac{\partial Q(\alpha)}{\partial \alpha_1} &= 1 - 2(\alpha_1 + \alpha_2) = 0 \\ \frac{\partial Q(\alpha)}{\partial \alpha_2} &= 1 - 2(\alpha_1 + \alpha_2) = 0 \end{aligned}\right\} \quad \alpha_1 + \alpha_2 = \frac{1}{2} \qquad \boxed{1}$$

$$\left.\sum_{i=1}^{2} \alpha_i d_i = 0 \right\} \quad -\alpha_1 + \alpha_2 = 0 \qquad \boxed{2}$$

Support Vector Machine

举个例子

$$\alpha_{o,1} = \alpha_{o,2} = \frac{1}{4}$$



$$\begin{aligned}
\mathbf{w}_o &= \sum_{i=1}^{N} \alpha_{o,i} d_1 \mathbf{x}_i = \alpha_{o,1} d_1 \mathbf{x}_1 + \alpha_{o,2} d_2 \mathbf{x}_2 \\
&= \frac{1}{4}(-1)\begin{bmatrix} -1 \\ -1 \end{bmatrix} + \frac{1}{4}(1)\begin{bmatrix} 1 \\ 1 \end{bmatrix} = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix} \\
b_o &= 1 - \mathbf{w}_o^T \mathbf{x}^{(s)} = 1 - [0.5\ 0.5]\begin{bmatrix} 1 \\ 1 \end{bmatrix} = 0
\end{aligned}$$

Support Vector Machine

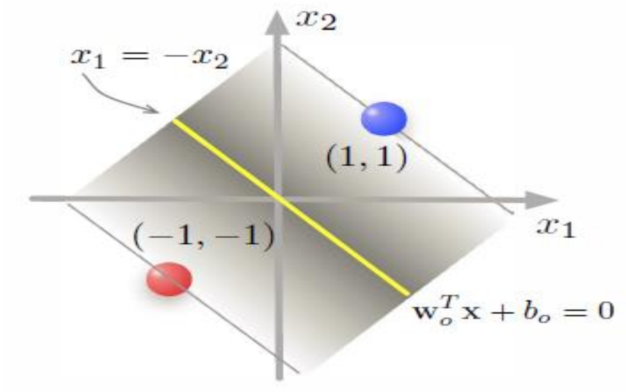
举个例子

Classify a "new" data point x_a as +1 or −1 based on

$$\begin{aligned} d_a &= \text{sgn}\left[g(\mathbf{x}_a)\right] \\ &= \text{sgn}\left[\mathbf{w}_o^T \mathbf{x}_a + b_o\right] \end{aligned}$$

where

$$\text{sgn}[u] = \begin{cases} +1 & \text{if } u > 0 \\ 0 & \text{if } u = 0 \\ -1 & \text{if } u < 0 \end{cases}$$



Example: Given a SVM with

$$\mathbf{w}_o = \begin{bmatrix} 0.5 \\ 0.5 \end{bmatrix}, \; b_o = 0$$

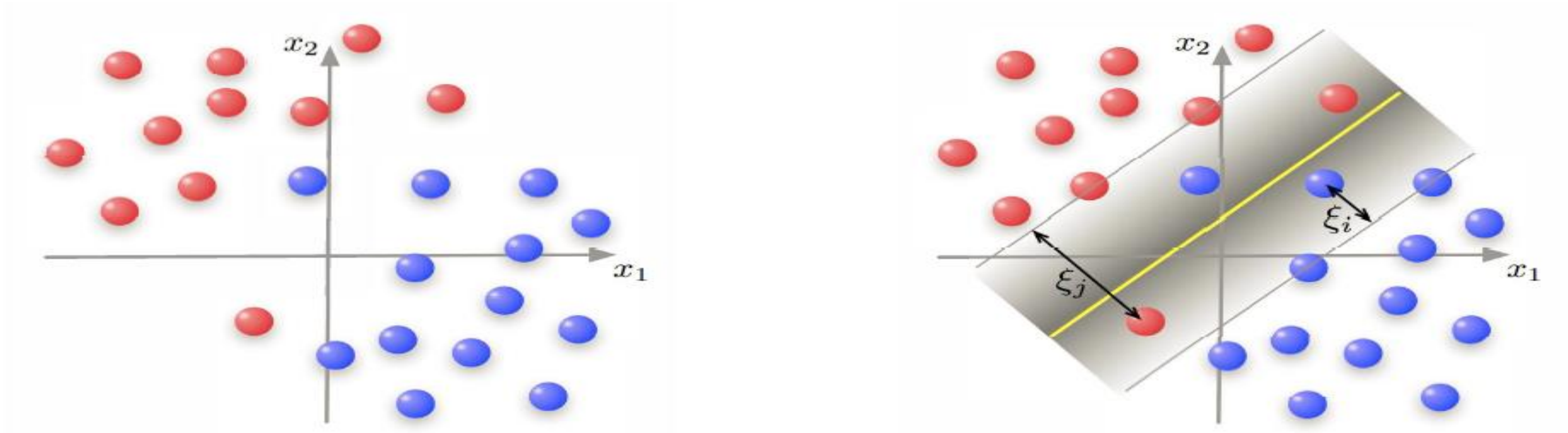
To classify $\mathbf{x}_a = \begin{bmatrix} 2 \\ 2 \end{bmatrix}$

$$\begin{aligned} d_a &= \text{sgn}\left[g(\mathbf{x}_a)\right] \\ &= \text{sgn}\left[\mathbf{w}_o^T \mathbf{x}_a + b_o\right] \\ &= \text{sgn}\left[[0.5 \; 0.5] \begin{bmatrix} 2 \\ 2 \end{bmatrix}\right] \\ &= +1 \end{aligned}$$

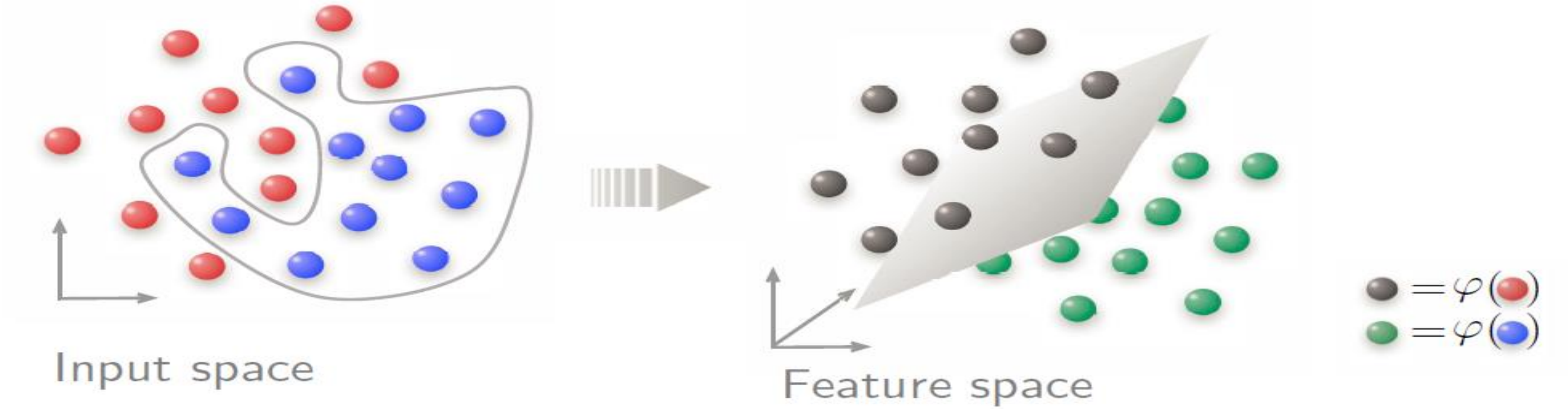
Support Vector Machine

若线性不可分呢?

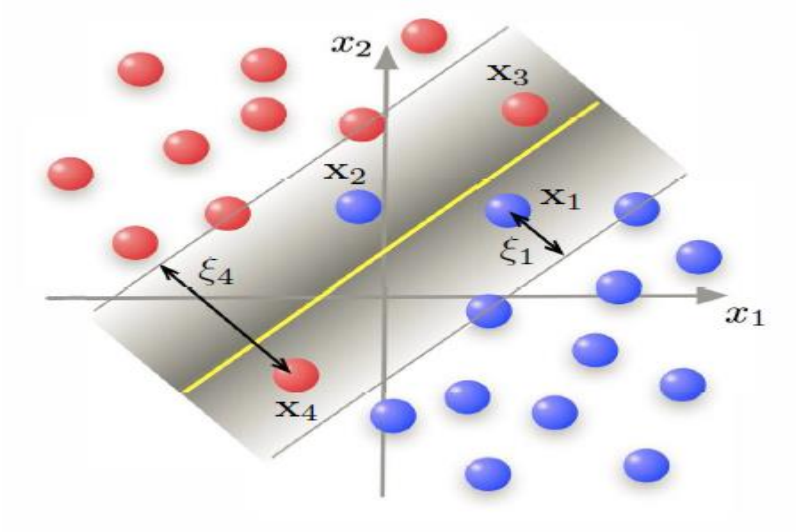
1. Find optimal hyperplane to minimize classification error



2. Transform data into higher dimension space for separation



Support Vector Machine



核心idea：引入松弛变量

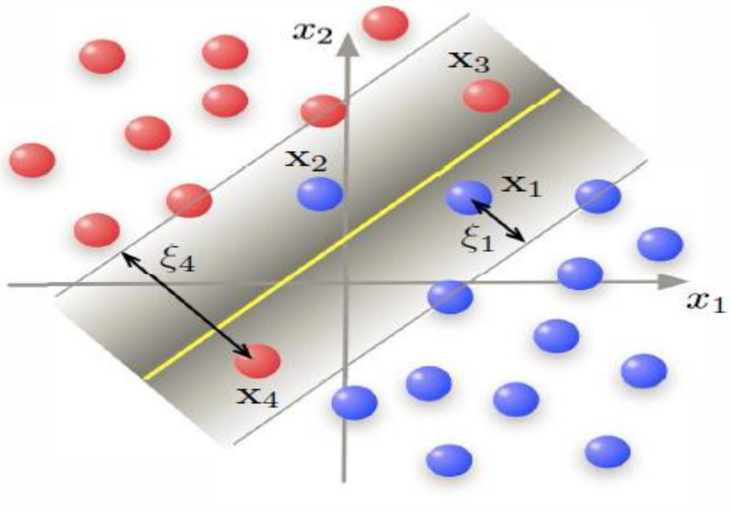
Nonnegative slack variables

$$\xi_i\,,\quad i = 1, \ldots, N$$

$$d_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1 - \xi_i$$
$$\xi_i \geq 0$$

$\xi_i = 0$	Data point outside region of separation and correctly separated
$0 \leq \xi_i \leq 1$	Data point in region of separation and on correct side of hyperplane (e.g., ξ_1 and ξ_3)
$\xi_i > 1$	Data point in region of separation but on wrong side of hyperplane (e.g., ξ_2 and ξ_4)

Support Vector Machine



Optimal hyperplane must also minimize error penalty

$$\sum_{i=1}^{N} \xi_i$$

New function to be minimized

$$\Phi(\mathbf{w}, \xi) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N} \xi_i$$

- Trade off between a large margin and a small error penalty
- Value of $C > 0$ set by user to control trade-off

AI300学院

Support Vector Machine

线性可分

Find : $\mathbf{w}$ and b

Minimizing : $\Phi(\mathbf{w}) = \frac{1}{2}\mathbf{w}^T\mathbf{w}$

Subject to : $d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1$

线性不可分

Find: $\boldsymbol{w}$ and b

Min: $\phi(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i$

S.T: $d_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1 - \xi_i$

$\xi_i \geq 0$

新的拉格朗日函数

$$J(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i - \sum_i \alpha_i\left(d_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) - 1 + \xi_i\right) - \sum_i \beta_i\xi_i$$

新的KKT条件

$$\frac{\partial J}{\partial \boldsymbol{w}} = 0 \qquad \frac{\partial J}{\partial b} = 0 \qquad \frac{\partial J}{\partial \xi_i} = 0$$

$d_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) \geq 1 - \xi_i$ $\qquad$ $\xi_i \geq 0$

$\alpha_i \geq 0$ $\qquad$ $\beta_i \geq 0$

$\alpha_i(d_i\left(\boldsymbol{w}^T\boldsymbol{x}_i + b\right) - 1 + \xi_i) = 0$ $\qquad$ $\beta_i\xi_i = 0$

AI300学院

Support Vector Machine

原问题

Find: $\boldsymbol{w}$ and b

Min: $\phi(\boldsymbol{w}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i$

S.T: $d_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) \geq 1 - \xi_i$

$\xi_i \geq 0$

新的拉格朗日函数

$$J(\boldsymbol{w}, b, \boldsymbol{\alpha}, \boldsymbol{\beta}, \boldsymbol{\xi}) = \frac{1}{2}\boldsymbol{w}^T\boldsymbol{w} + C\sum_i \xi_i - \sum_i \alpha_i\big(d_i(\boldsymbol{w}^T\boldsymbol{x}_i + b) - 1 + \xi_i\big) - \sum_i \beta_i\xi_i$$

$$\frac{\partial J}{\partial \boldsymbol{w}} = 0 \Rightarrow \boldsymbol{w} = \sum_i \alpha_i d_i \boldsymbol{x}_i$$

$$\frac{\partial J}{\partial b} = 0 \Rightarrow \sum_i \alpha_i d_i = 0$$

$$\frac{\partial J}{\partial \xi_i} = 0 \Rightarrow \alpha_i + \beta_i = C$$

$$\left.\begin{array}{l} \alpha_i + \beta_i = C \\ \alpha_i \geq 0 \quad \beta_i \geq 0 \end{array}\right\} 0 \leq \alpha_i \leq C$$

新的对偶问题

Min: $Q(\alpha) = \sum_i \alpha_i - \frac{1}{2}\sum_i\sum_j \alpha_i\alpha_j d_i d_j \boldsymbol{x}_i^T\boldsymbol{x}_j$

S.T: $\sum_i \alpha_i d_i = 0$

$0 \leq \alpha_i \leq C$

AI300学院

Support Vector Machine

	Primal	Dual (Tutorial problem $\diamond$)
Find :	$\mathbf{w}\,, b$	α_i
Minimizing :	$\Phi\left(\mathbf{w}, \xi\right)$	$\mathbf{w}$
Maximizing :	$-$	$Q(\alpha)$
Subject to :	$d_i\left(\mathbf{w}^T\mathbf{x}_i + b\right) \geq 1 - \xi_i$	$\sum_{i=1}^{N} \alpha_i d_i = 0$
	$\xi_i \geq 0$	$0 \leq \alpha_i \leq C$

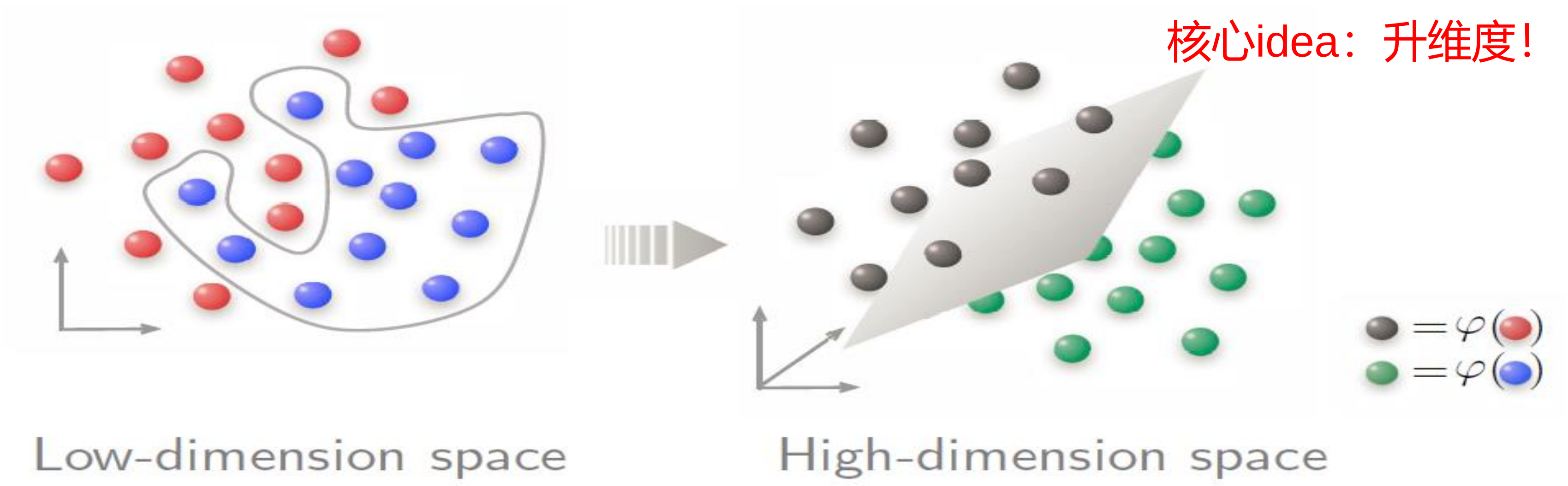
$$\Phi\left(\mathbf{w}, \xi\right) = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\sum_{i=1}^{N}\xi_i \qquad Q(\alpha) = \sum_{i=1}^{N}\alpha_i - \frac{1}{2}\sum_{i=1}^{N}\sum_{i=j}^{N}\alpha_i\,\alpha_j\,d_i\,d_j\,\mathbf{x}_i^T\mathbf{x}_j$$

After $\alpha_{o,i}$ is obtained, we can calculate $\mathbf{w}_o$ and b_o as follows:

$$\mathbf{w}_o = \sum_{i=1}^{N}\alpha_{o,i}\,d_i\,\mathbf{x}_i\,, \quad b_o = 1 - \mathbf{w}_o^T\mathbf{x}^{(s)}$$

where $\mathbf{x}^{(s)}$ is a support vector with $d^{(s)} = +1$

Support Vector Machine

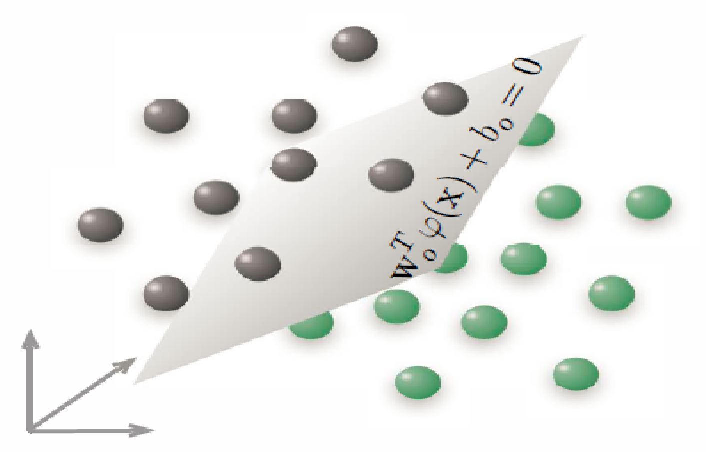


Cover's Theorem

Non-linearly-separable patterns may be transformed into a new feature space in which they are linearly separable, provided that

1. transformation φ is nonlinear
2. dimension of feature space is high enough

Support Vector Machine



Optimal hyperplane in feature space

$$g(\mathbf{x}) = \mathbf{w}_o^T\varphi(\mathbf{x}) + b_o = 0$$

For training data $\mathbf{x}_i$

$$\begin{cases} g(\mathbf{x}_i) = \mathbf{w}_o^T\varphi(\mathbf{x}_i) + b_o \geq +1 & \text{for } d_i = +1 \\ g(\mathbf{x}_i) = \mathbf{w}_o^T\varphi(\mathbf{x}_i) + b_o \leq -1 & \text{for } d_i = -1 \end{cases}$$

or, in a compact form:

$$d_i\, g(\mathbf{x}_i) = d_i\left(\mathbf{w}_o^T\varphi(\mathbf{x}_i) + b_o\right) \geq 1$$

数学形式都没有变!只是x变成$\varphi(x)$!

Support Vector Machine

$$
\begin{aligned}
J(\mathbf{w}, b, \alpha) &= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \left(d_i \left(\mathbf{w}^T \varphi(\mathbf{x}_i) + b \right) - 1 \right) \\
&= \frac{1}{2}\mathbf{w}^T\mathbf{w} - \sum_{i=1}^{N} \alpha_i \, d_i \, \mathbf{w}^T \varphi(\mathbf{x}_i) - b \sum_{i=1}^{N} \alpha_i \, d_i + \sum_{i=1}^{N} \alpha_i
\end{aligned}
$$

From ▸ KKT conditions: $\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \varphi(\mathbf{x}_i)$ and $\sum_{i=1}^{N} \alpha_i \, d_i = 0$

数学形式都没有变!只是x变成$\varphi(x)$!

So $\mathbf{w}^T\mathbf{w} = \sum_{i=1}^{N} \alpha_i d_i \mathbf{w}^T \varphi(\mathbf{x}_i) = \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \varphi^T(\mathbf{x}_i)\varphi(\mathbf{x}_j)$

Let $Q(\alpha) \equiv J(\mathbf{w}, b, \alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{i=j}^{N} \alpha_i \alpha_j \, d_i d_j \underbrace{\varphi^T(\mathbf{x}_i)\varphi(\mathbf{x}_j)}_{K(\mathbf{x}_i, \mathbf{x}_j)}$

Kernel: $\underbrace{K(\mathbf{x}_i, \mathbf{x}_j) = K(\mathbf{x}_j, \mathbf{x}_i)}_{\text{symmetric}} = \varphi^T(\mathbf{x}_i)\varphi(\mathbf{x}_j) = \varphi^T(\mathbf{x}_j)\varphi(\mathbf{x}_i)$

Support Vector Machine

Dual problem with soft margin

Find : α_i

Maximize : $Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \mathbf{x}_i^T \mathbf{x}_j$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0\,,\ 0 \leq \alpha_i \leq C$

Dual problem with soft margin and transformation

Find : α_i

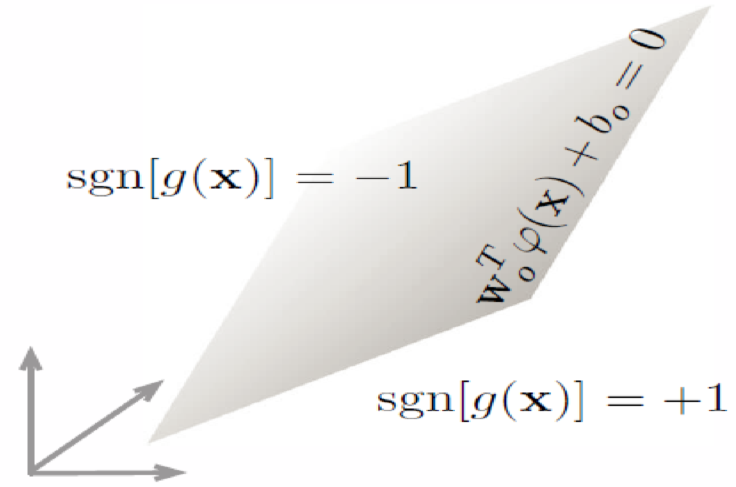
Maximize : $Q(\alpha) = \sum_{i=1}^{N} \alpha_i - \frac{1}{2} \sum_{i=1}^{N} \sum_{j=1}^{N} \alpha_i \alpha_j d_i d_j \boldsymbol{\varphi}^T(\mathbf{x}_i) \boldsymbol{\varphi}(\mathbf{x}_j)$

Subject to : $\sum_{i=1}^{N} \alpha_i d_i = 0\,,\ 0 \leq \alpha_i \leq C$

Support Vector Machine

Given optimal value $\alpha_{o,i}$

$$\begin{cases} \mathbf{w}_o = \sum_{i=1}^{N} \alpha_{o,i}\, d_i \boldsymbol{\varphi}(\mathbf{x}_i) \\ b_o = 1 - \mathbf{w}_o \mathbf{x}^{(s)} \\ (\mathbf{x}^{(s)} \text{ is a SV with } d^{(s)} = 1) \end{cases}$$



Discriminant function

$$g(\mathbf{x}) = \mathbf{w}_o^T \boldsymbol{\varphi}(\mathbf{x}) + b_o = \sum_{i=1}^{N} \alpha_{o,i} d_i \underbrace{\boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{x})}_{K(\mathbf{x}_i,\mathbf{x})} + b_o$$

To classify a new data point $\mathbf{x}_a$

$$d_a = \mathrm{sgn}\,[g(\mathbf{x}_a)]$$

例如径向基函数

$$k(x_i, x_j) = e^{-\|x_i - x_j\|_2^2/\sigma^2}$$

Support Vector Machine

If we know $\varphi(\cdot)$, then

$$K(\cdot,\cdot) = \boldsymbol{\varphi}^T(\cdot)\boldsymbol{\varphi}(\cdot)$$

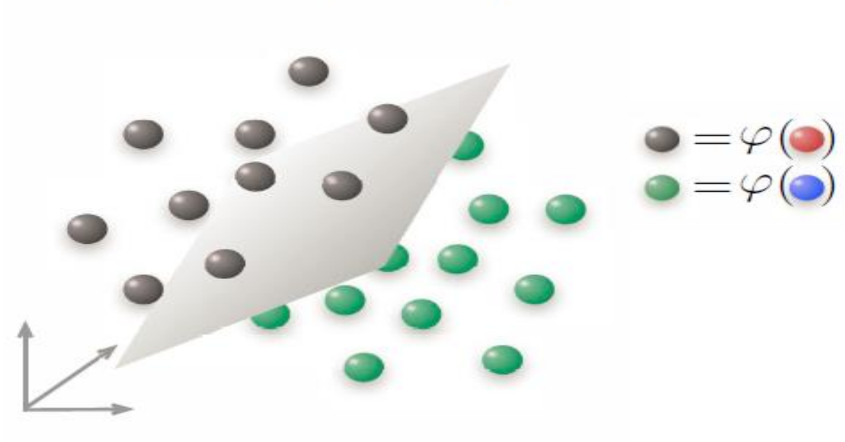
Suppose for a 2D vector $\mathbf{x}_i = [x_{i,1}\ x_{i,2}]^T$, we have:

$$\varphi(\mathbf{x}_i) = \begin{bmatrix} 1 \\ x_{i,1}^2 \\ \sqrt{2}x_{i,1}x_{i,2} \\ x_{i,2}^2 \\ \sqrt{2}x_{i,1} \\ \sqrt{2}x_{i,2} \end{bmatrix}$$

For two vectors $\mathbf{x}_i$ and $\mathbf{x}_j$

$$\begin{aligned} & K(\mathbf{x}_i, \mathbf{y}_j) \\ = \ & \boldsymbol{\varphi}^T(\mathbf{x}_i)\boldsymbol{\varphi}(\mathbf{y}_j) \\ = \ & (1 + x_{i,1}x_{j,1} + x_{i,2}x_{j,2})^2 \end{aligned}$$

Solution requires φ



Finding explicit φ is difficult

Kernel trick:

Find $K(\cdot,\cdot)$ directly

Support Vector Machine

Kernel trick Find an expression for $K(\cdot,\cdot)$ directly without knowing $\varphi(\cdot)$

Procedure Choose an expression for $K(\cdot,\cdot)$. If this expression satisfies the Mercer's Condition, then it can be used as a kernel

Mercer's condition

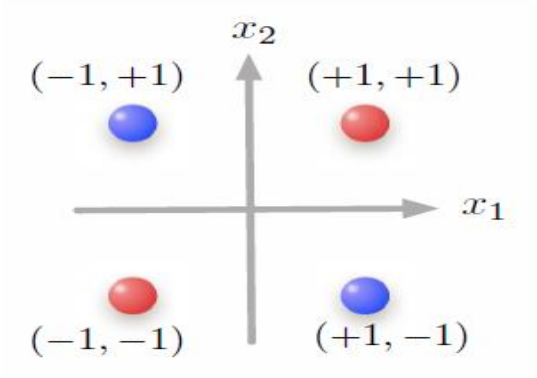
For data set $\{\mathbf{x}_i\}$, $i = 1, 2, \ldots, N$, the Gram matrix

$$\mathbf{K} = \begin{bmatrix} K(\mathbf{x}_1, \mathbf{x}_1) & \ldots & K(\mathbf{x}_1, \mathbf{x}_N) \\ \vdots & \ddots & \vdots \\ K(\mathbf{x}_N, \mathbf{x}_1) & \ldots & K(\mathbf{x}_N, \mathbf{x}_N) \end{bmatrix} \in R^{N \times N}$$

is positive semi-definite (i.e., its eigenvalues are nonnegative)

Support Vector Machine

举个例子



i	$\mathbf{x}_i$	d_i
1	$[-1,-1]^T$	-1
2	$[-1,+1]^T$	$+1$
3	$[+1,-1]^T$	$+1$
4	$[+1,+1]^T$	-1

Choose kernel:

$$\begin{aligned} K(\mathbf{x}_i,\mathbf{x}_j) &= (1+\mathbf{x}_i^T\mathbf{x}_j)^2 \\ &= 1 + x_{i,1}^2 x_{j,1}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} \\ &+ x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2} \end{aligned}$$

The Gram matrix is

$$\mathbf{K} = \begin{bmatrix} 9 & 1 & 1 & 1 \\ 1 & 9 & 1 & 1 \\ 1 & 1 & 9 & 1 \\ 1 & 1 & 1 & 9 \end{bmatrix}$$

The eigenvalues are

In *Mathematica*

```
In[1] = Eigenvalues[
{{9, 1, 1, 1}, {1, 9, 1, 1},
 {1, 1, 9, 1}, {1, 1, 1, 9}}]
Out[1]  = {12, 8, 8, 8}
```

Support Vector Machine

举个例子

$$
\begin{aligned}
Q(\alpha) &= \sum_{i=1}^{4}\alpha_i - \frac{1}{2}\sum_{i=1}^{4}\sum_{j=1}^{4}\alpha_i\alpha_j d_i d_j K(\mathbf{x}_i,\mathbf{x}_j) \\
&= \alpha_1+\alpha_2+\alpha_3+\alpha_4 - \\
&\quad \frac{1}{2}\Big(9\alpha_1^2 - 2\alpha_1\alpha_2 - \alpha_1\alpha_3 + 2\alpha_1\alpha_4 + \\
&\quad 9\alpha_2^2 + 2\alpha_2\alpha_3 - 2\alpha_2\alpha_4 + 9\alpha_3^2 - 2\alpha_3\alpha_4 + 9\alpha_4^2\Big)
\end{aligned}
$$

$$
\left.\begin{aligned}
9\alpha_1 - \alpha_2 - \alpha_3 + \alpha_4 &= 1 \\
-\alpha_1 + 9\alpha_2 + \alpha_3 - \alpha_4 &= 1 \\
-\alpha_1 + \alpha_2 + 9\alpha_3 - \alpha_4 &= 1 \\
\alpha_1 - \alpha_2 - \alpha_3 + 9\alpha_4 &= 1
\end{aligned}\right\} \quad \frac{\partial Q}{\partial \alpha_i} = 0
$$

$$
-\alpha_1 + \alpha_2 + \alpha_3 - \alpha_4 = 0 \;\} \quad \text{from KKT conditions}
$$

$$
\alpha_{o,1} = \alpha_{o,2} = \alpha_{o,3} = \alpha_{o,4} = \frac{1}{8} \qquad (\text{all } \mathbf{x}_i \text{ are SVs})
$$

Support Vector Machine

举个例子

Discriminant function ▸ View

$$g(\mathbf{x}) = \sum_{i=1}^{N} \alpha_{o,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_o$$

$$= \alpha_{o,1} d_1 K(\mathbf{x}, \mathbf{x}_1) + \alpha_{o,2} d_2 K(\mathbf{x}, \mathbf{x}_2) + \alpha_{o,3} d_3 K(\mathbf{x}, \mathbf{x}_3) + \alpha_{o,4} d_4 K(\mathbf{x}, \mathbf{x}_4) + b_o$$

General vector: $\mathbf{x} = [x_1, x_2]^T$

Training data: $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$

i	$\mathbf{x}_i$	d_i
1	$[-1, -1]^T$	-1
2	$[-1, +1]^T$	$+1$
3	$[+1, -1]^T$	$+1$
4	$[+1, +1]^T$	-1

$$K(\mathbf{x}_i, \mathbf{x}_j) = 1 + x_{i,1}^2 x_{j,1}^2 + 2x_{i,1}x_{i,2}x_{j,1}x_{j,2} + x_{i,2}^2 x_{j,2}^2 + 2x_{i,1}x_{j,1} + 2x_{i,2}x_{j,2}$$

For example, $K(\mathbf{x}, \mathbf{x}_1)$ is calculated as follows:

$$\begin{aligned}
& 1 + x_1^2 x_{1,1}^2 + 2x_1 x_2 x_{1,1} x_{1,2} + x_2^2 x_{1,2}^2 + 2x_1 x_{1,1} + 2x_2 x_{1,2} \\
= & 1 + x_1^2(-1)^2 + 2x_1 x_2(-1)(-1) + x_2^2(-1)^2 + 2x_1(-1) + 2x_2(-1) \\
= & 1 + x_1^2 + 2x_1 x_2 + x_2^2 - 2x_1 - 2x_2
\end{aligned}$$

AI300学院

Support Vector Machine

举个例子

$$
\begin{aligned}
g(\mathbf{x}) &= \sum_{i=1}^{N} \alpha_{o,i} d_i K(\mathbf{x}, \mathbf{x}_i) + b_o \\
&= \alpha_{o,1} d_1 K(\mathbf{x}, \mathbf{x}_1) + \\
&\quad \alpha_{o,2} d_2 K(\mathbf{x}, \mathbf{x}_2) + \\
&\quad \alpha_{o,3} d_3 K(\mathbf{x}, \mathbf{x}_3) + \\
&\quad \alpha_{o,4} d_4 K(\mathbf{x}, \mathbf{x}_4) + b_o \\
&= \frac{1}{8}(-8x_1x_2) + b_o \\
&= -x_1x_2 + b_o
\end{aligned}
$$

General vector: $\mathbf{x} = [x_1, x_2]^T$

Training data: $\mathbf{x}_i = [x_{i,1}, x_{i,2}]^T$

i	$\mathbf{x}_i$	d_i	$\alpha_{o,i}$
1	$[-1, -1]^T$	-1	$1/8$
2	$[-1, +1]^T$	$+1$	$1/8$
3	$[+1, -1]^T$	$+1$	$1/8$
4	$[+1, +1]^T$	-1	$1/8$

To find b_o, pick a support vector, say, $\mathbf{x}_2 = [x_{2,1}\ x_{2,2}]$, then

$$
\begin{aligned}
g(\mathbf{x}_2) &= -x_{2,1}x_{2,2} + b_0 = 1 \\
b_o &= 1 + x_{2,1}x_{2,2} = 1 + (-1)1 = 0
\end{aligned}
$$

SVM分类实战

SVM scikit-learn的python实现

【对scikit-learn中SVM概述】

在scikit-learn中， 导入SVM操作为from sklearn.svm import SVC

参数总结:

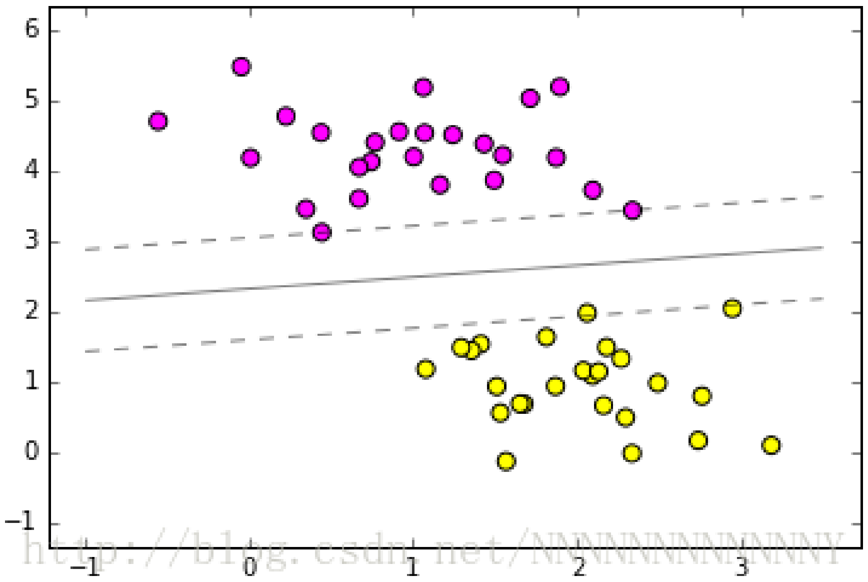
- kernel：核函数 默认为rbf型，其他可选的有linear，poly，sigmoid，precomputed，以及可调用自定义形式callable；
- gamma： ‘rbf’,‘poly’ 和 ‘sigmoid’的核函数参数。默认是’ auto’；
- probability： 是否采用概率估计？.默认为False；
- tol： 停止训练的误差值大小，默认为1e-3；
- decision_function（X）： 样本X与分离超平面的距离。

SVM分类实战

SVM scikit-learn的python实现

【线性 SVM 分类器】

```
from sklearn.svm import SVC
clf = SVC(kernel='linear')
clf.fit(X, y)
def plot_svc_decision_function(clf, ax=None):
if ax is None:
        ax = plt.gca()
    x = np.linspace(plt.xlim()[0], plt.xlim()[1], 30)
    y = np.linspace(plt.ylim()[0], plt.ylim()[1], 30)
    Y, X = np.meshgrid(y, x)
    P = np.zeros_like(X)
    for i, xi in enumerate(x):
        for j, yj in enumerate(y):
            P[i, j] = clf.decision_function([xi, yj])
```



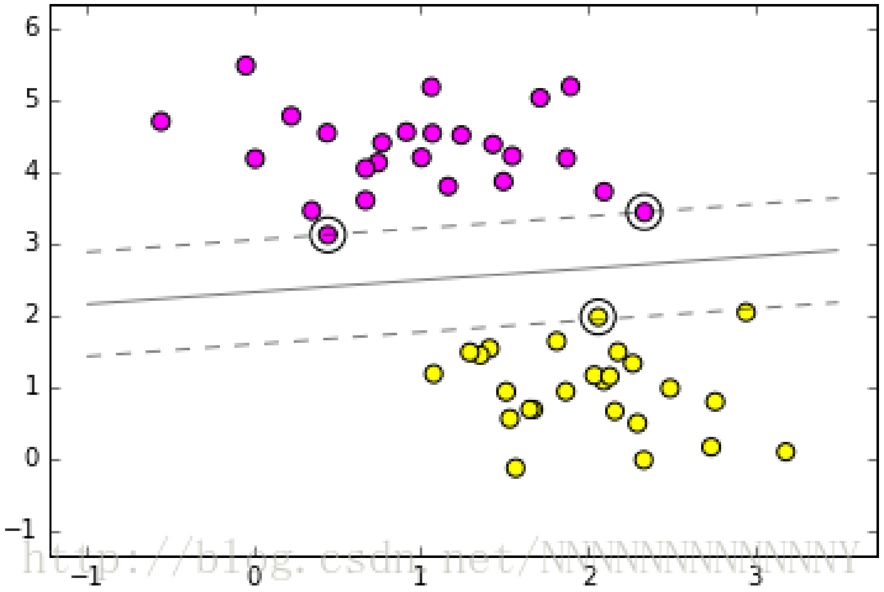
绘制SVM决策边界

SVM分类实战

SVM scikit-learn的python实现

【线性 SVM 分类器】

```
plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='spring')
plot_svc_decision_function(clf)
plt.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1],
          s=200, facecolors='none');
```



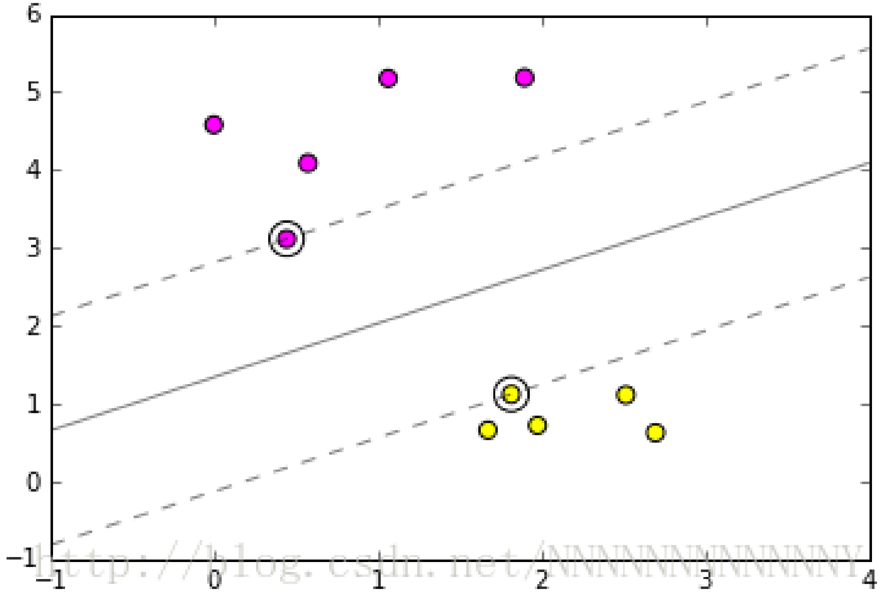
sklearn的SVM里面会有一个属性support_vectors_，标示“支持向量”，也就是样本点里离超平面最近的点，组成的。

SVM分类实战

SVM scikit-learn的python实现

【线性 SVM 分类器】

```
from IPython.html.widgets import interact
def plot_svm(N=100):
    X, y = make_blobs(n_samples=200, centers=2,
              random_state=0, cluster_std=0.60)
    X = X[:N]
    y = y[:N]
    clf = SVC(kernel='linear')
    clf.fit(X, y)
    plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='spring')
    plt.xlim(-1, 4)
    plt.ylim(-1, 6)
    plot_svc_decision_function(clf, plt.gca())
    plt.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1],
                s=200, facecolors='none')
interact(plot_svm, N=[10, 200], kernel='linear');
```



可以用Python的 interact 函数来看看样本点的分布，会怎么样影响超平面

SVM分类实战

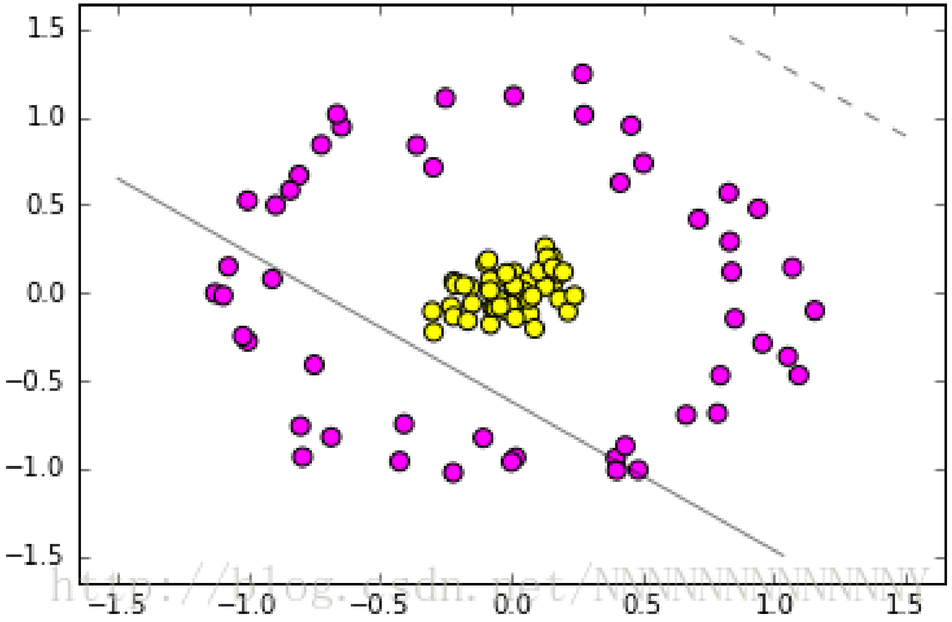
SVM scikit-learn的python实现

【SVM 与 核函数】

```
from sklearn.datasets.samples_generator import make_circles
X, y = make_circles(100, factor=.1, noise=.1)

clf = SVC(kernel='linear').fit(X, y)

plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='spring')
plot_svc_decision_function(clf);
```



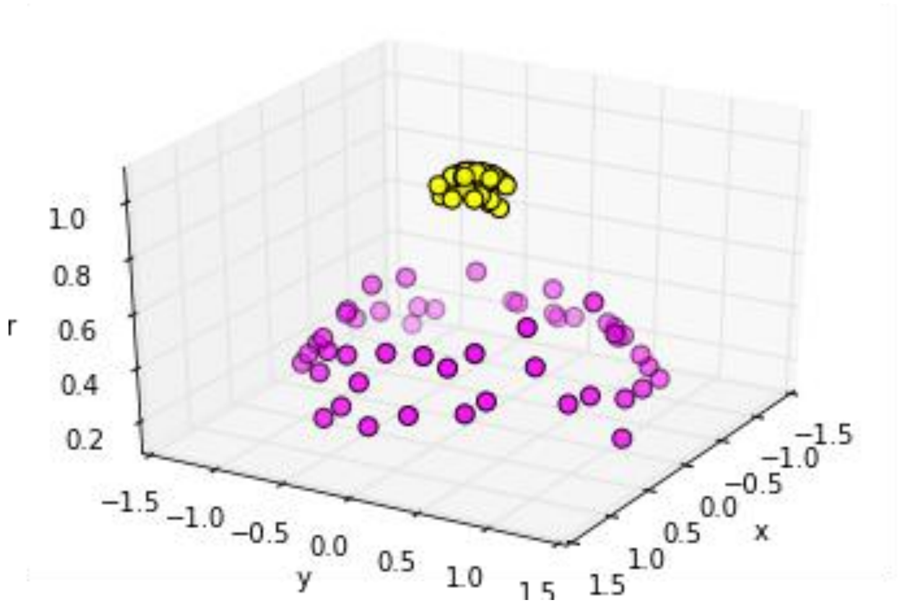
线性的kernel(线性的SVM)对于这种非线性可切分的数据集，是无能为力的。

SVM分类实战

SVM scikit-learn的python实现

【带高斯核的SVM】

```
r = np.exp(-(X[:, 0] ** 2 + X[:, 1] ** 2))
from mpl_toolkits import mplot3d
def plot_3D(elev=30, azim=30):
    ax = plt.subplot(projection='3d')
    ax.scatter3D(X[:, 0], X[:, 1], r, c=y, s=50, cmap='spring')
    ax.view_init(elev=elev, azim=azim)
    ax.set_xlabel('x')
    ax.set_ylabel('y')
    ax.set_zlabel('r')
interact(plot_3D, elev=[-90, 90], azip=(-180, 180));
```



原本在2维空间无法切分的2类点，映射到3维空间以后，可以由一个平面轻松地切开了。

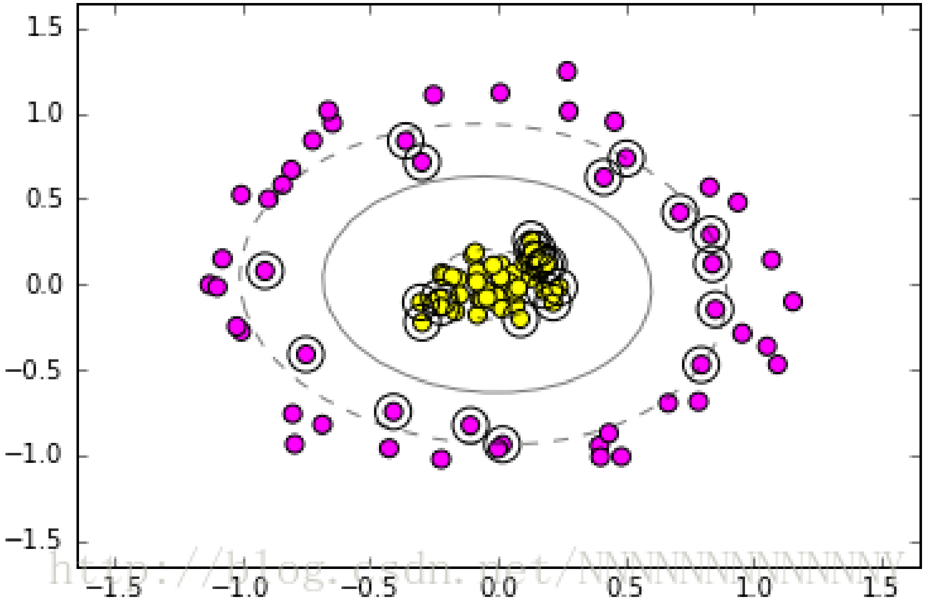
SVM分类实战

SVM scikit-learn的python实现

【带高斯核的SVM】

```
clf = SVC(kernel='rbf')
clf.fit(X, y)

plt.scatter(X[:, 0], X[:, 1], c=y, s=50, cmap='spring')
plot_svc_decision_function(clf)
plt.scatter(clf.support_vectors_[:, 0], clf.support_vectors_[:, 1],
            s=200, facecolors='none');
```



Thank You

AI300学院