

法律声明

□ 本课件包括演示文稿、示例、代码、题库、视频和声音等内容，小象学院和主讲老师拥有完全知识产权的权利；只限于善意学习者在本课程使用，不得在课程范围外向任何第三方散播。任何其他人或机构不得盗版、复制、仿造其中的创意及内容，我们保留一切通过法律手段追究违反者的权利。

□ 课程详情请咨询

■ 微信公众号：小象

■ 新浪微博：ChinaHadoop



支持向量机



小象学院
ChinaHadoop.cn

邹博

主要内容和目标

- 理解支持向量机SVM的原理和目标
- 掌握支持向量机的计算过程和算法步骤
- 理解软间隔最大化的含义
 - 对线性不可分的数据给出(略有错误)的分割面
 - 线性可分的数据需要使用“软间隔”目标函数吗?
- 了解核函数的思想
- 了解SMO算法的过程

各种概念

□ 线性可分支持向量机

- 硬间隔最大化hard margin maximization
- 硬间隔支持向量机

□ 线性支持向量机

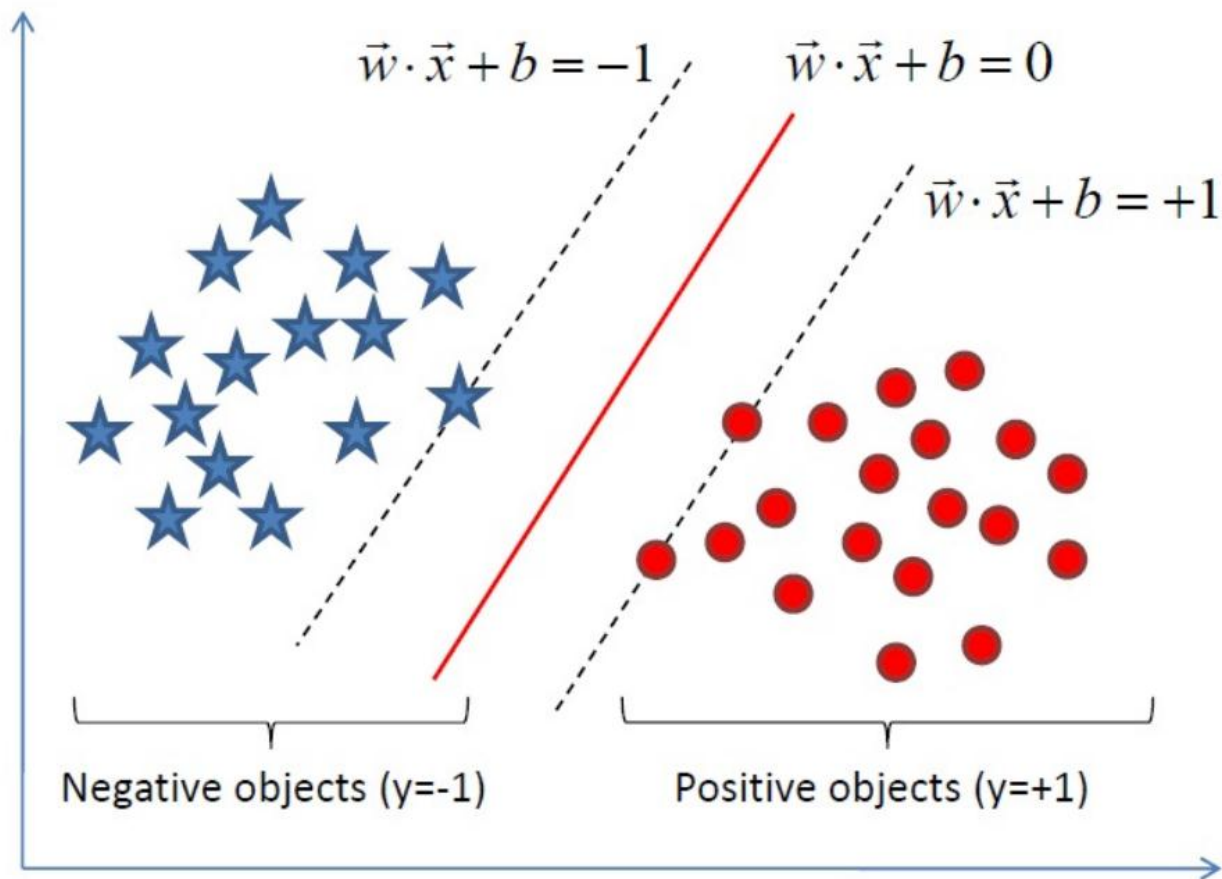
- 软间隔最大化soft margin maximization
- 软间隔支持向量机

□ 非线性支持向量机

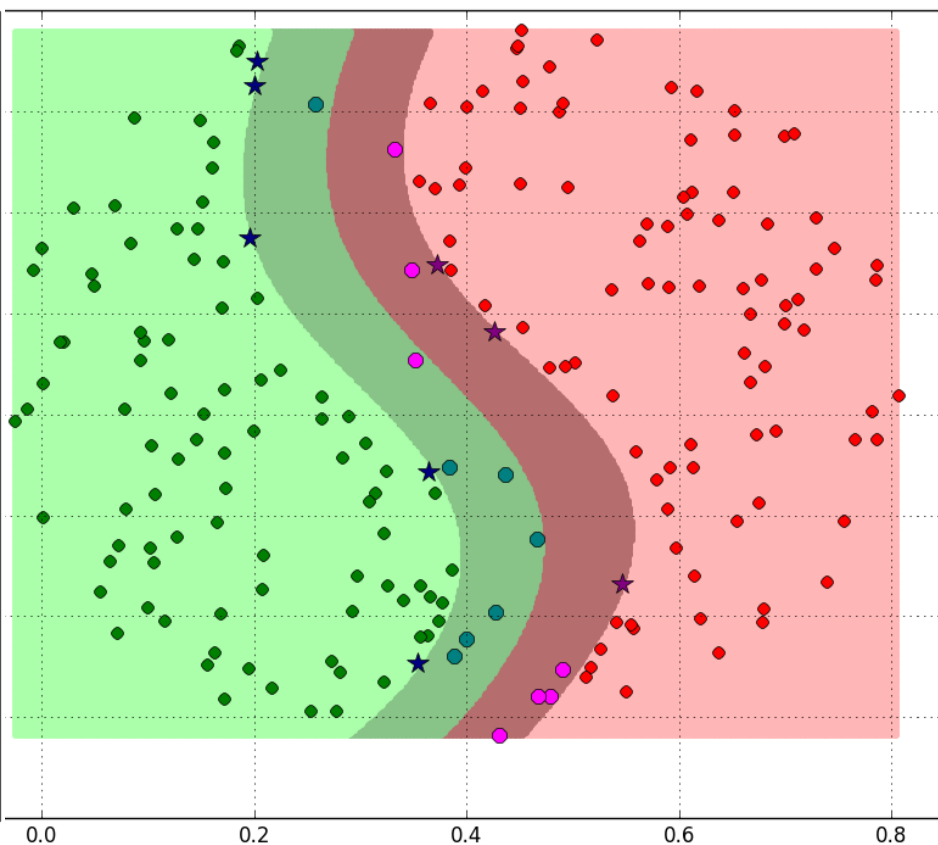
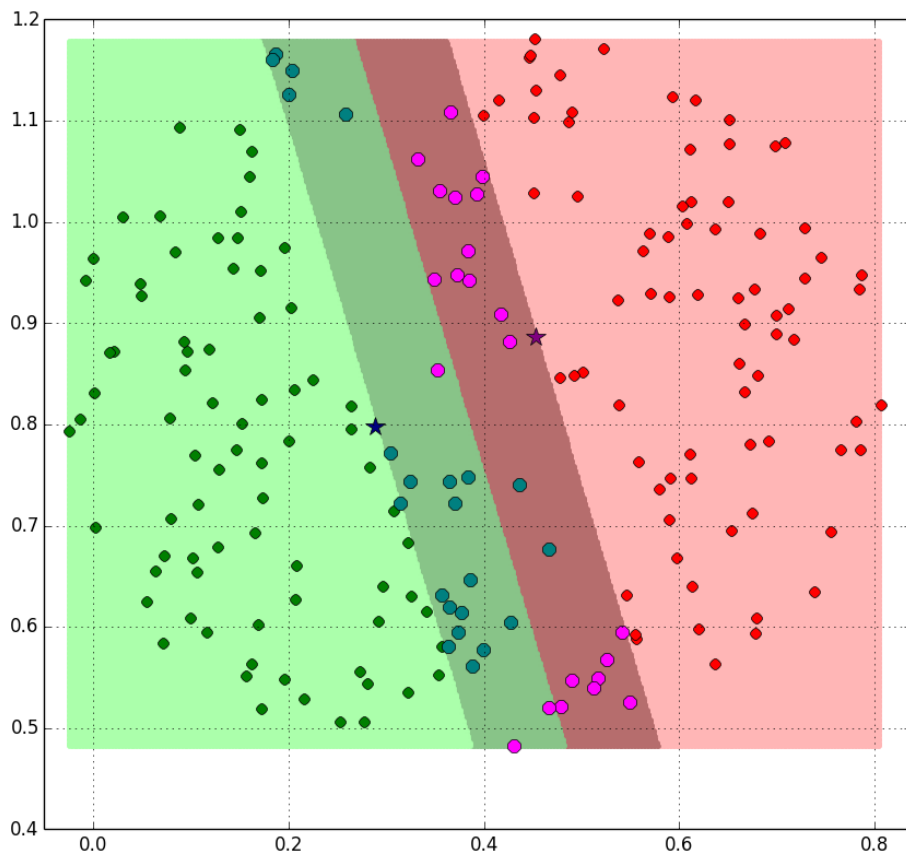
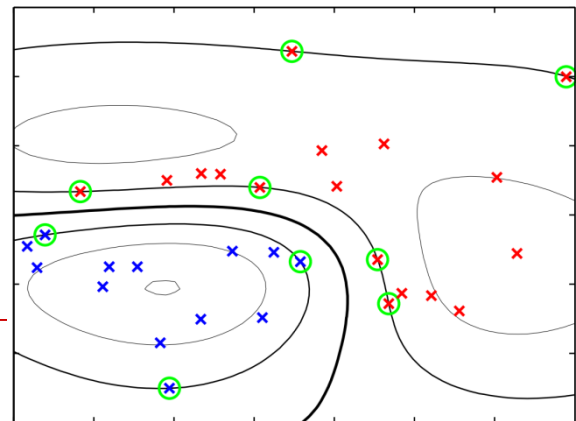
- 核函数kernel function

- 注：以上概念的提法，各个文献并不十分统一。

线性可分支持向量机

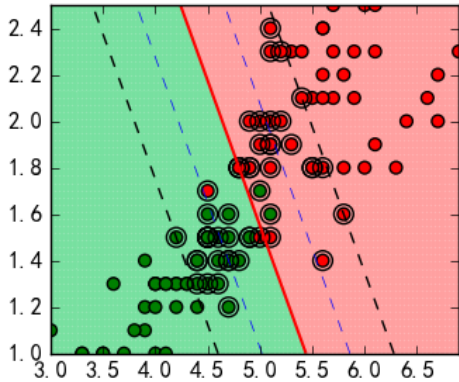


使用核解决线性不可分

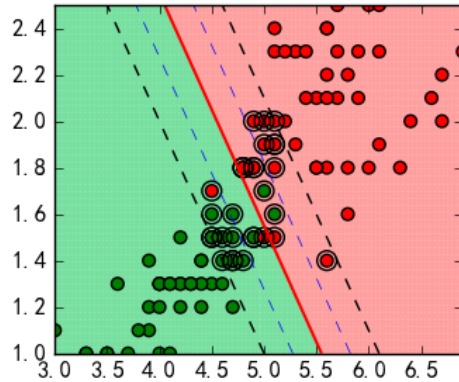


SVM不同参数的分类

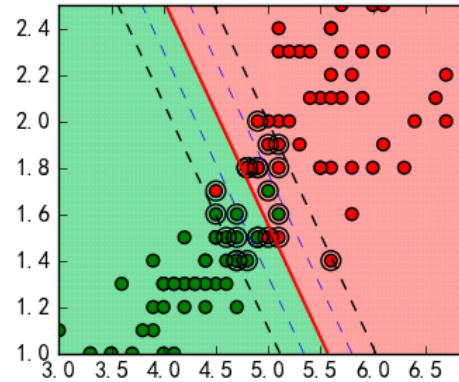
线性核, $C=0.1$



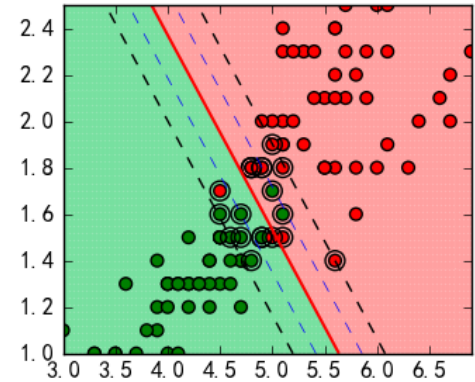
线性核, $C=0.5$



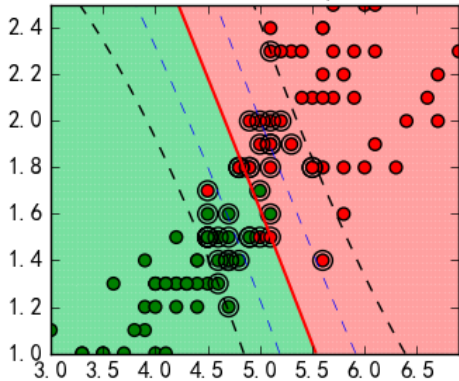
线性核, $C=1.0$



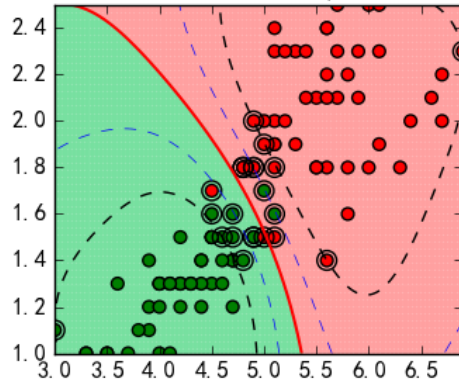
线性核, $C=2.0$



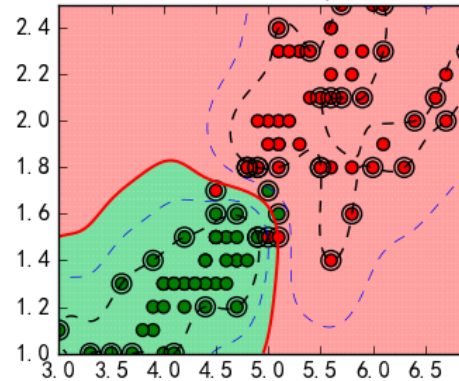
高斯核, $C=1.0$, $\gamma=0.1$



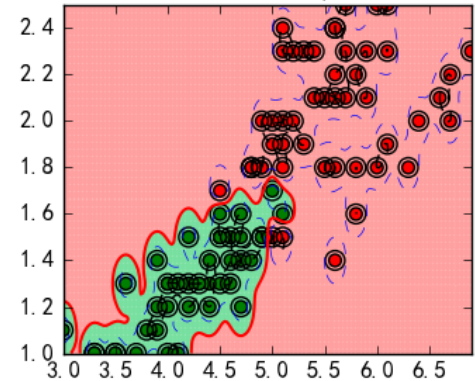
高斯核, $C=1.0$, $\gamma=1.0$



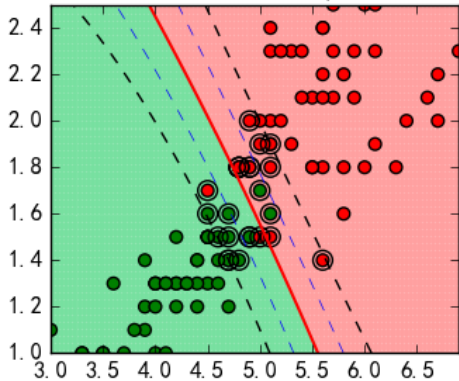
高斯核, $C=1.0$, $\gamma=10.0$



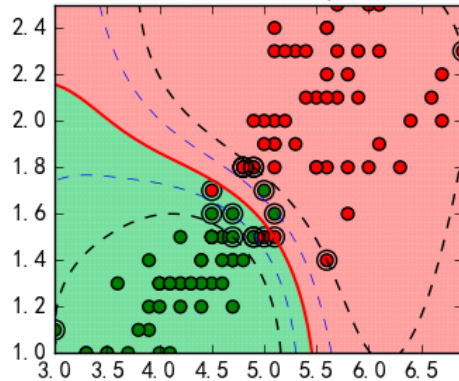
高斯核, $C=1.0$, $\gamma=100.0$



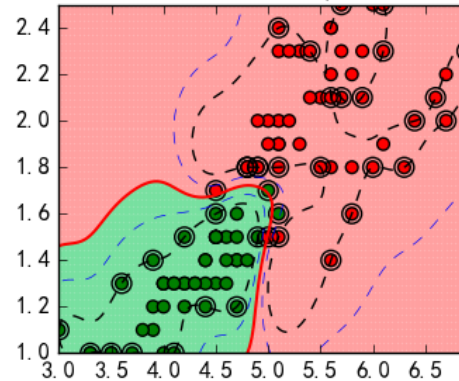
高斯核, $C=5.0$, $\gamma=0.1$



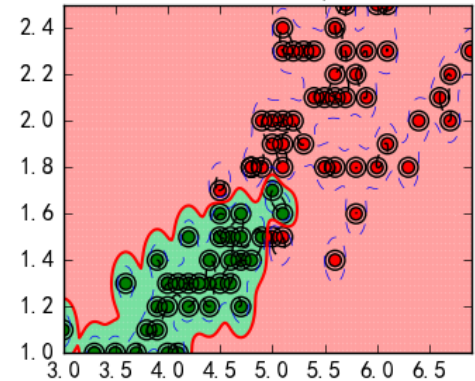
高斯核, $C=5.0$, $\gamma=1.0$



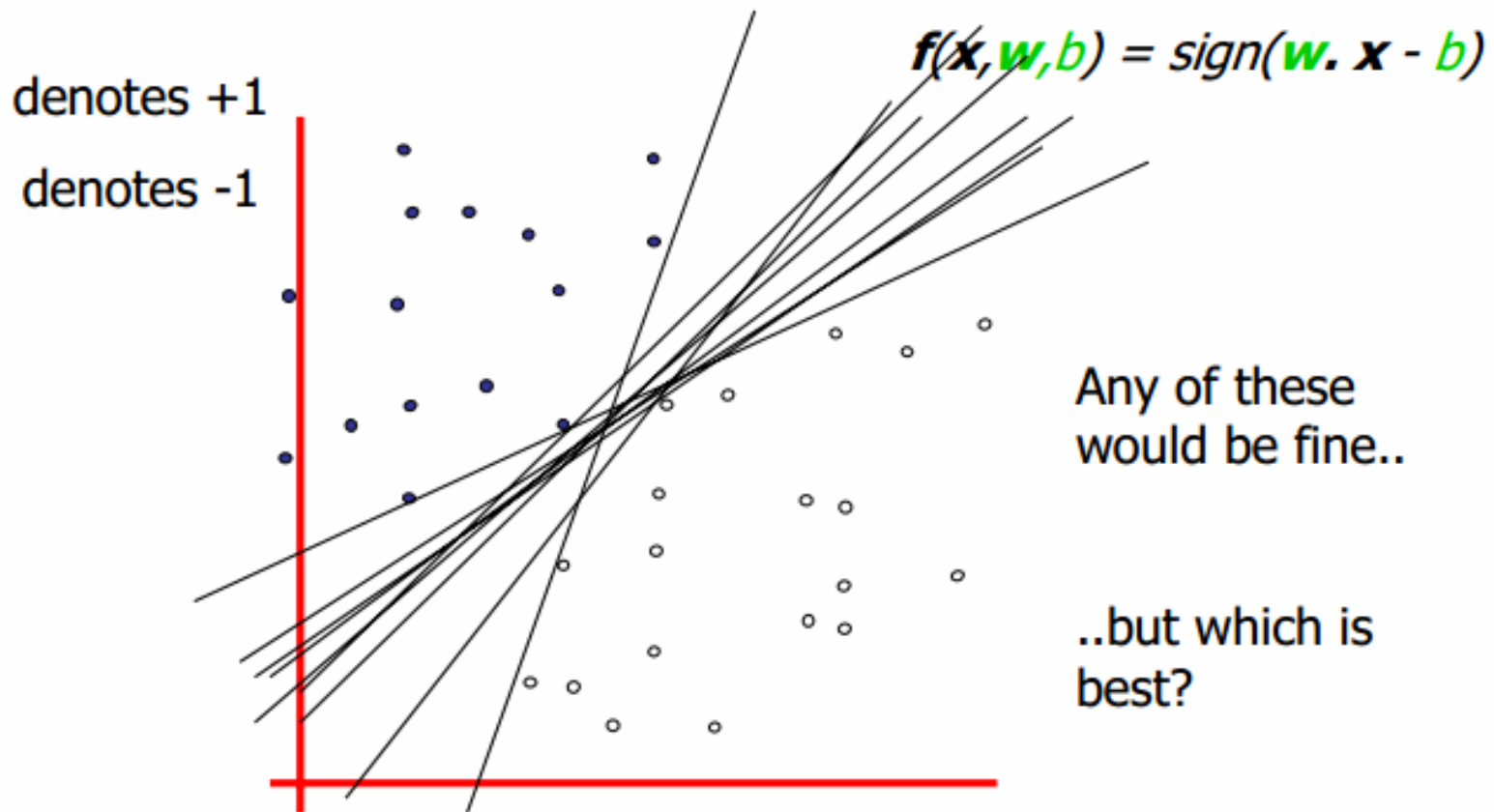
高斯核, $C=5.0$, $\gamma=10.0$



高斯核, $C=5.0$, $\gamma=100.0$



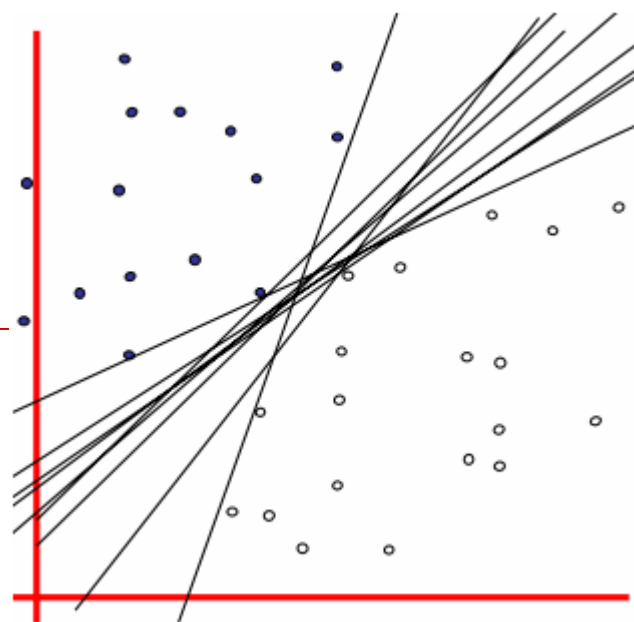
线性分类问题



输入数据

- 假设给定一个特征空间上的训练数据集
 $T = \{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \dots, (\mathbf{x}_N, y_N)\}$
 - 其中, $\mathbf{x}_i \in \mathbb{R}^n$, $y_i \in \{+1, -1\}$, $i=1, 2, \dots, N$ 。
- \mathbf{x}_i 为第*i*个实例(若 $n>1$, \mathbf{x}_i 为向量);
- y_i 为 \mathbf{x}_i 的类标记;
 - 当 $y_i=+1$ 时, 称 \mathbf{x}_i 为正例;
 - 当 $y_i=-1$ 时, 称 \mathbf{x}_i 为负例;
- (\mathbf{x}_i, y_i) 称为样本点。

线性可分支持向量机



- 给定线性可分训练数据集，通过
间隔最大化得到的分离超平面为

$$w^T \cdot \Phi(x) + b = 0$$

相应的分类决策函数 $f(x) = \text{sign}(w^T \Phi(x) + b)$

该决策函数称为线性可分支持向量机。

- $\Phi(x)$ 是某个确定的特征空间转换函数，它的作用是将 x 映射到(更高的)维度。
 - 最简单直接的： $\Phi(x) = x$
- 稍后会看到，求解分离超平面问题可以等价求解相应的**凸二次规划问题**。

整理符号

- 分割平面: $w^T \Phi(x) + b = 0$
- 训练集: x_1, x_2, \dots, x_n
- 目标值: $y_1, y_2, \dots, y_n, y_i \in \{-1, 1\}$
- 新数据的分类: $y(x) = w^T \Phi(x) + b$
 $\text{sign}(y(x))$

优化问题

□ 一般优化问题的Lagrange乘子法

$$\text{minimize } f_0(x), \quad x \in \mathbf{R}^n$$

$$\text{subject to } f_i(x) \leq 0, \quad i = 1, \dots, m$$

$$h_j(x) = 0, \quad j = 1, \dots, p$$

□ Lagrange函数

$$L(x, \lambda, \nu) = f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{j=1}^p \nu_j h_j(x)$$

□ Lagrange对偶函数

$$g(\lambda, \nu) = \inf_{x \in D} L(x, \lambda, \nu) = \inf_{x \in D} (f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{j=1}^p \nu_j h_j(x))$$

Lagrange对偶函数(dual function)

□ Lagrange对偶函数

$$g(\lambda, \nu) = \inf_{x \in D} L(x, \lambda, \nu) = \inf_{x \in D} (f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) + \sum_{i=1}^p \nu_i h_i(x))$$

□ 若没有下确界，定义：

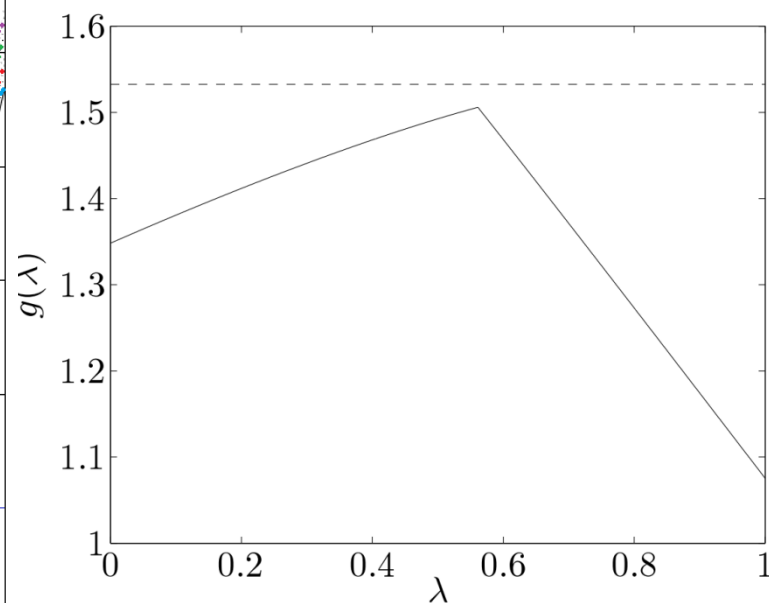
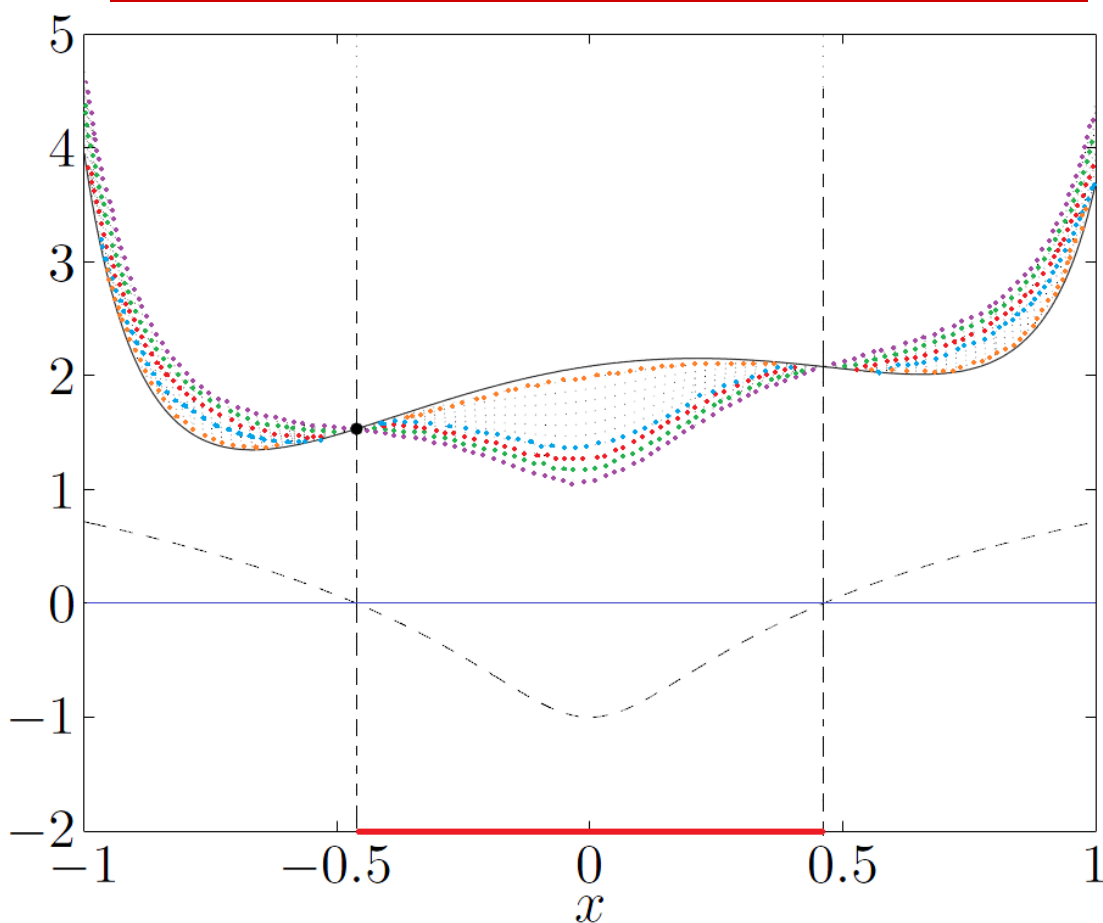
$$g(\lambda, \nu) = -\infty$$

□ 根据定义，显然有：对 $\forall \lambda > 0$ ， $\forall \nu$ ，若原优化问题有最优值 p^* ，则

$$g(\lambda, \nu) \leq p^*$$

□ 进一步：Lagrange对偶函数为凹函数。

左侧为原函数，右侧为对偶函数



推导目标函数

□ 根据题设 $y(x) = w^T \Phi(x) + b$

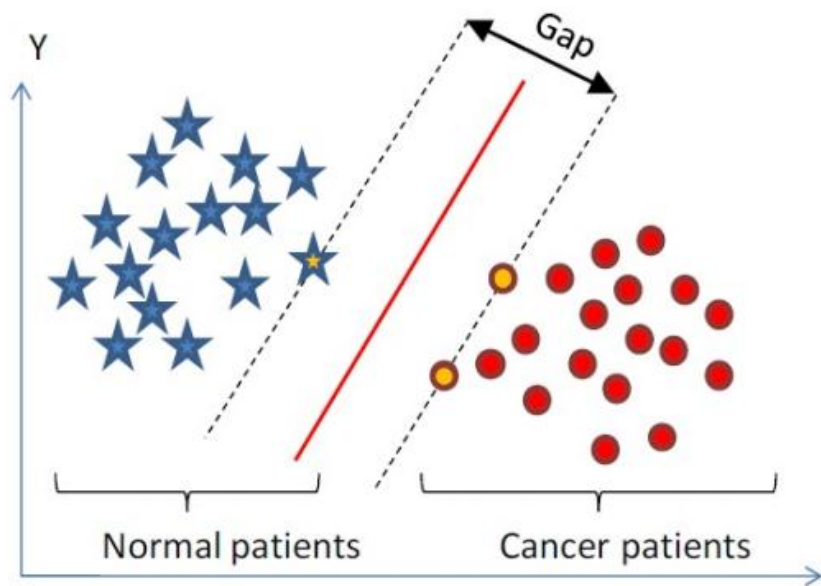
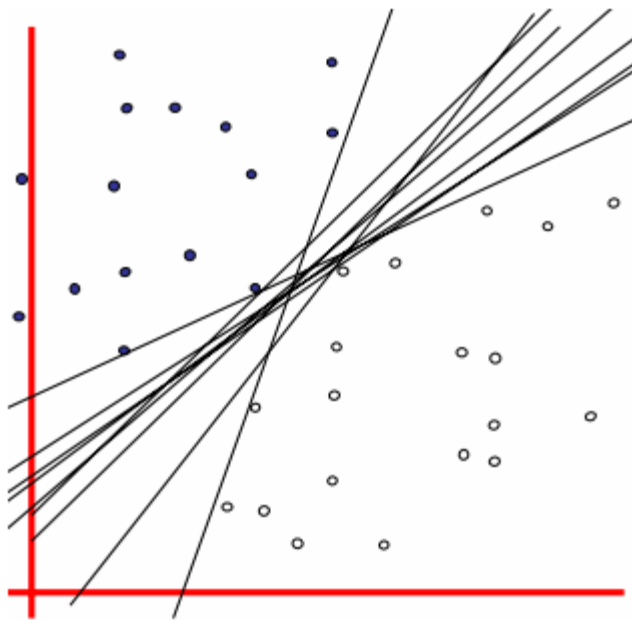
□ 有：
$$\begin{cases} y(x_i) > 0 \Leftrightarrow y_i = +1 \\ y(x_i) < 0 \Leftrightarrow y_i = -1 \end{cases} \Rightarrow y_i \cdot y(x_i) > 0$$

□ w, b 等比例缩放，则 t^*y 的值同样缩放，从而：

$$\frac{y_i \cdot y(x_i)}{\|w\|} = \frac{y_i \cdot (w^T \cdot \Phi(x_i) + b)}{\|w\|}$$

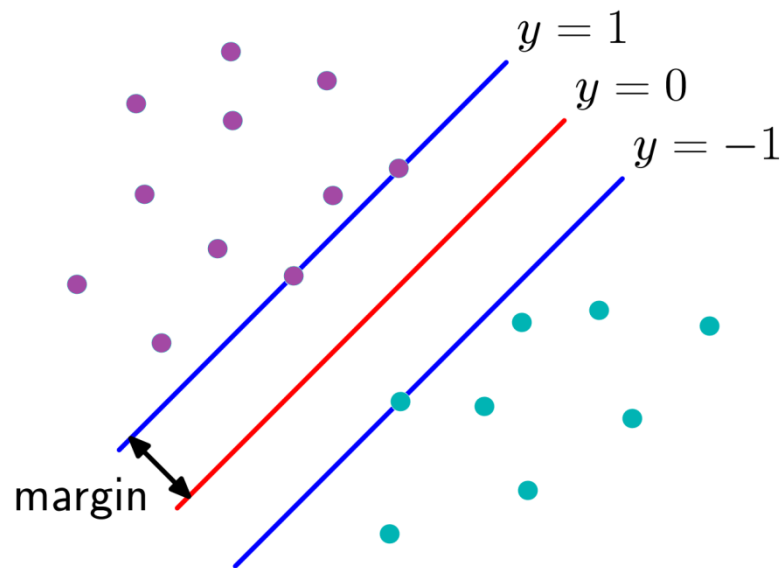
最大间隔分离超平面 $\frac{y_i \cdot y(x_i)}{\|w\|} = \frac{y_i \cdot (w^T \cdot \Phi(x_i) + b)}{\|w\|}$

□ 目标函数: $\arg \max_{w,b} \left\{ \frac{1}{\|w\|} \min_i [y_i \cdot (w^T \cdot \Phi(x_i) + b)] \right\}$

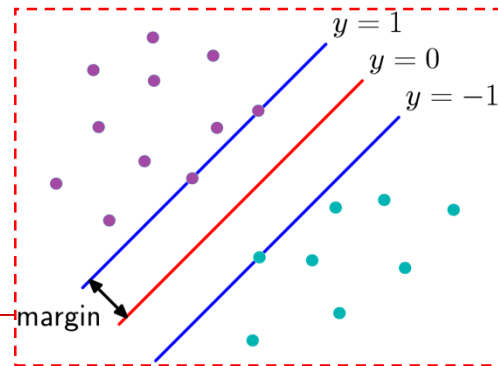


函数间隔和几何间隔 $\frac{w^T \cdot \Phi(x_i) + b}{\|w\|}$

- 分割平面: $y = w^T \cdot \Phi(x) + b$
- 总可以通过等比例缩放 w 的方法, 使得两类点的函数值都满足 $|y| \geq 1$



建立目标函数



□ 总可以通过等比例缩放 w 的方法，使得两类的函数值都满足 $|y| \geq 1$

□ 约束条件： $y_i \cdot (w^T \cdot \Phi(x_i) + b) \geq 1$

□ 原目标函数：
$$\arg \max_{w,b} \left\{ \frac{1}{\|w\|} \min_i [y_i \cdot (w^T \cdot \Phi(x_i) + b)] \right\}$$

□ 新目标函数：
$$\arg \max_{w,b} \frac{1}{\|w\|}$$

建立目标函数

$$\max_{w,b} \frac{1}{\|w\|}$$
$$s.t. \quad y_i(w^T \cdot \Phi(x_i) + b) \geq 1, \quad i = 1, 2, \dots, n$$

$$\min_{w,b} \frac{1}{2} \|w\|^2$$
$$s.t. \quad y_i(w^T \cdot \Phi(x_i) + b) \geq 1, \quad i = 1, 2, \dots, n$$

拉格朗日乘子法 $\min_{w,b} \frac{1}{2} \|w\|^2, s.t. y_i(w^T \cdot \Phi(x_i) + b) \geq 1, i = 1, 2, \dots, N$

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

□ 原问题是极小极大问题

$$\min_{w,b} \max_{\alpha} L(w, b, \alpha)$$

□ 原始问题的对偶问题，是极大极小问题

$$\max_{\alpha} \min_{w,b} L(w, b, \alpha)$$

拉格朗日函数

□ 将拉格朗日函数 $L(w, b, a)$ 分别对 w , b 求偏导并令其为0:

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^n \alpha_i y_i \Phi(x_i)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow 0 = \sum_{i=1}^n \alpha_i y_i$$

计算拉格朗日函数的对偶函数

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

$$= \frac{1}{2} w^T w - w^T \sum_{i=1}^n \alpha_i y_i \Phi(x_i) - b \sum_{i=1}^n \alpha_i y_i + \sum_{i=1}^n \alpha_i$$

$$= \frac{1}{2} w^T \sum_{i=1}^n \alpha_i y_i \Phi(x_i) - w^T \sum_{i=1}^n \alpha_i y_i \Phi(x_i) - b \cdot 0 + \sum_{i=1}^n \alpha_i$$

$$= \sum_{i=1}^n \alpha_i - \frac{1}{2} \left(\sum_{i=1}^n \alpha_i y_i \Phi(x_i) \right)^T \sum_{i=1}^n \alpha_i y_i \Phi(x_i)$$

$$= \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \Phi^T(x_i) \Phi(x_j)$$

$$a^* = \arg \max_{\alpha} \left(\sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i,j=1}^n \alpha_i \alpha_j y_i y_j \Phi^T(x_i) \Phi(x_j) \right)$$

$$\begin{aligned} w &= \sum_{i=1}^n \alpha_i y_i \Phi(x_i) \\ 0 &= \sum_{i=1}^n \alpha_i y_i \end{aligned}$$

继续求 $\min_{w,b} L(w,b,\alpha)$ 对 α 的极大

$$\max_{\alpha} \sum_{i=1}^n \alpha_i - \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (\Phi(x_i) \cdot \Phi(x_j))$$

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

$$\alpha_i \geq 0, \quad i = 1, 2, \dots, n$$

整理目标函数：添加负号

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

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

$$\alpha_i \geq 0, \quad i = 1, 2, \dots, n$$

线性可分支持向量机器学习算法

□ 构造并求解约束最优化问题

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

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

$$\alpha_i \geq 0, \quad i = 1, 2, \dots, n$$

□ 求得最优解 α^*

线性可分支持向量机器学习算法

□ 计算

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

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

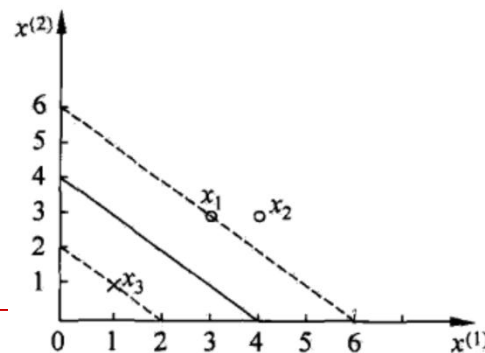
□ 求得分离超平面

$$w^* \Phi(x) + b^* = 0$$

□ 分类决策函数

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

举例



□ 给定3个数据点：正例点 $x_1=(3,3)^T$, $x_2=(4,3)^T$, 负例点 $x_3=(1,1)^T$, 求线性可分支持向量机。

□ 目标函数：

$$\begin{aligned} \min_{\alpha} \quad & \frac{1}{2} \sum_{i=1}^n \sum_{j=1}^n \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^n \alpha_i \\ & = \frac{1}{2} (18\alpha_1^2 + 25\alpha_2^2 + 2\alpha_3^2 + 42\alpha_1\alpha_2 - 12\alpha_1\alpha_3 - 14\alpha_2\alpha_3) - \alpha_1 - \alpha_2 - \alpha_3 \\ \text{s.t.} \quad & \alpha_1 + \alpha_2 - \alpha_3 = 0 \\ & \alpha_i \geq 0, \quad i = 1, 2, 3 \end{aligned}$$

将约束带入目标函数，化简计算

- 将 $\alpha_1 + \alpha_2 = \alpha_3$
- 带入目标函数，得到关于 α_1, α_2 的函数：

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

- 对 α_1, α_2 求偏导并令其为0，易知 $s(\alpha_1, \alpha_2)$ 在点 $(1.5, -1)$ 处取极值。而该点不满足条件 $\alpha_2 \geq 0$ ，所以，最小值在边界上达到。
- 当 $\alpha_1 = 0$ 时，最小值 $s(0, 2/13) = -2/13 = -0.1538$
- 当 $\alpha_2 = 0$ 时，最小值 $s(1/4, 0) = -1/4 = -0.25$
- 于是， $s(\alpha_1, \alpha_2)$ 在 $\alpha_1 = 1/4, \alpha_2 = 0$ 时达到最小，此时， $\alpha_3 = \alpha_1 + \alpha_2 = 1/4$

分离超平面

□ $\alpha_1 = \alpha_3 = 1/4$ 对应的点 x_1, x_3 是支持向量。

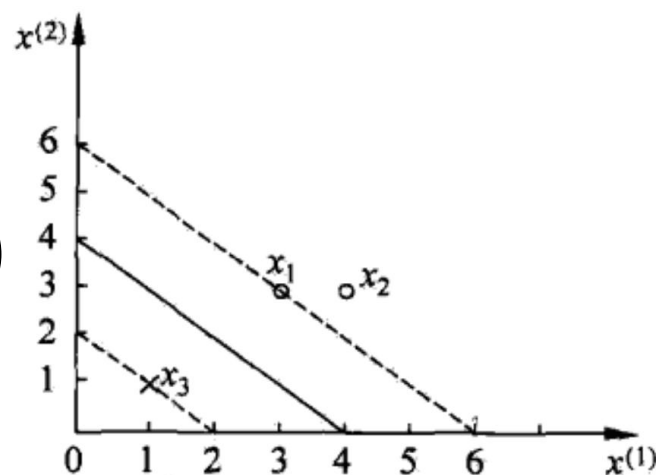
□ 带入公式: $w^* = \sum_{i=1}^N \alpha_i^* y_i \Phi(x_i)$

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

□ 得到 $w_1 = w_2 = 0.5$, $b = -2$

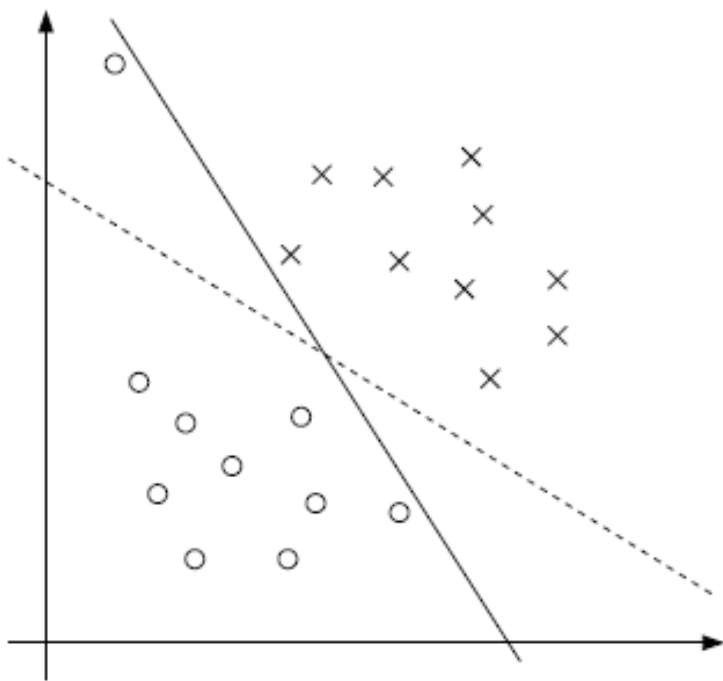
□ 因此, 分离超平面为 $\frac{1}{2}x_1 + \frac{1}{2}x_2 - 2 = 0$

□ 分离决策函数为 $f(x) = \text{sign}\left(\frac{1}{2}x_1 + \frac{1}{2}x_2 - 2\right)$



线性支持向量机

- 不一定分类完全正确的超平面就是最好的
- 样本数据本身线性不可分



线性支持向量机

- 若数据线性不可分，则增加松弛因子 $\xi_i \geq 0$ ，使函数间隔加上松弛变量大于等于1。这样，约束条件变成

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

- 目标函数：
$$\min_{w,b} \frac{1}{2} \|w\|^2 + C \sum_{i=1}^N \xi_i$$

线性SVM的目标函数

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

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

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

带松弛因子的SVM拉格朗日函数

□ 拉格朗日函数

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

□ 对 w, b, ξ 求偏导

$$\frac{\partial L}{\partial w} = 0 \Rightarrow w = \sum_{i=1}^n \alpha_i y_i \phi(x_i)$$

$$\frac{\partial L}{\partial b} = 0 \Rightarrow 0 = \sum_{i=1}^n \alpha_i y_i$$

$$\frac{\partial L}{\partial \xi_i} = 0 \Rightarrow C - \alpha_i - \mu_i = 0$$

带入目标函数

□ 将三式带入L中，得到

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

□ 对上式求关于 α 的极大，得到：

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

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

$$C - \alpha_i - \mu_i = 0$$

$$\alpha_i \geq 0$$

$$\mu_i \geq 0, \quad i = 1, 2, \dots, n$$

$$0 \leq \alpha_i \leq C$$

最终的目标函数

□ 整理，得到对偶问题：

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

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

$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, n$$

线性支持向量机学习算法

□ 构造并求解约束最优化问题

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

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

$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, n$$

□ 求得最优解 α^*

线性支持向量机学习算法

□ 计算

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

$$b^* = \frac{\max_{i: y_i=-1} w^* \cdot x_i + \min_{i: y_i=1} w^* \cdot x_i}{2}$$

■ 注意：计算 b^* 时，需要使用满足条件 $0 < \alpha_j < C$ 的向量

■ 实践中往往取支持向量的所有值取平均，作为 b^*

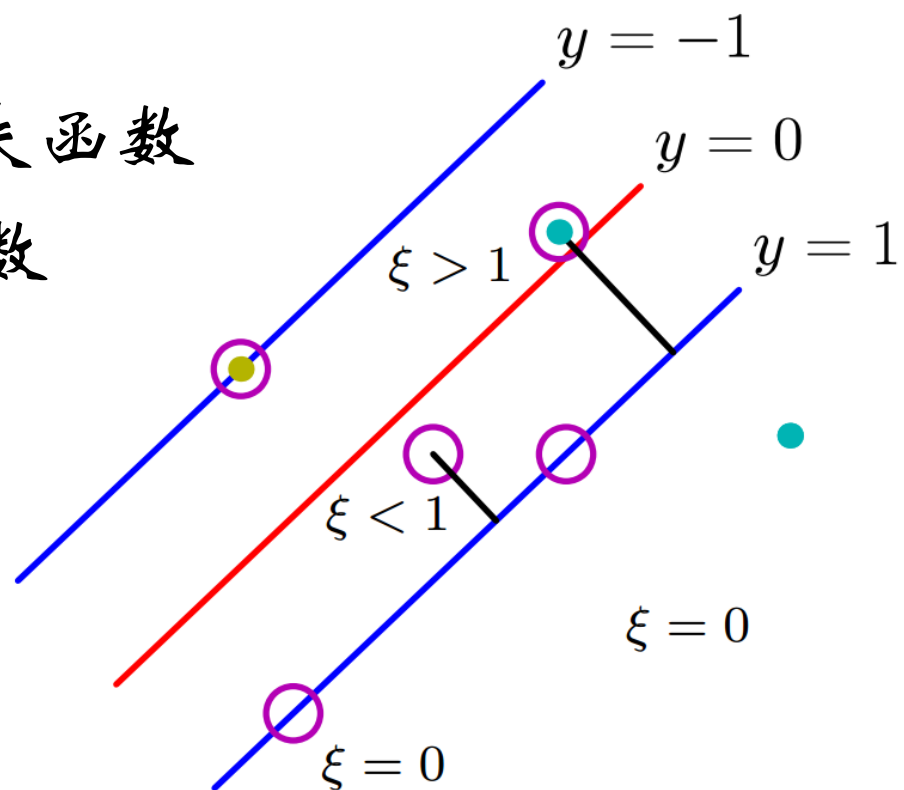
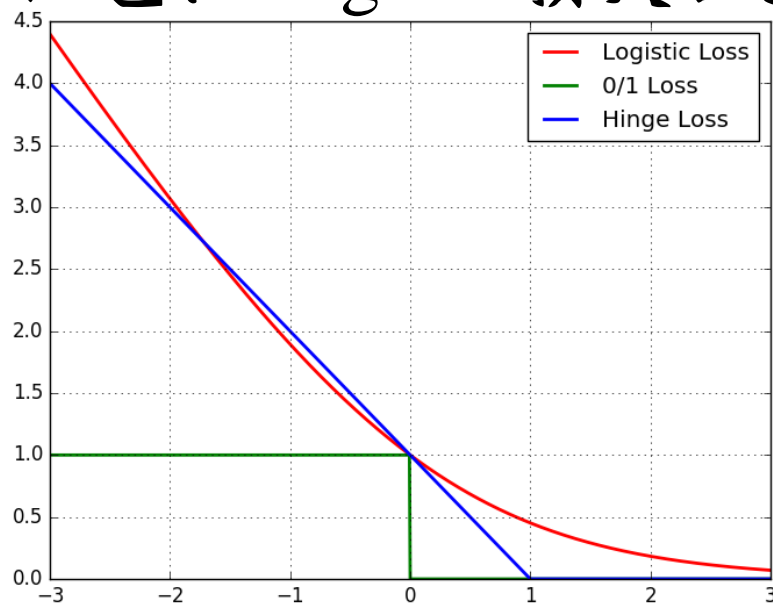
□ 求得分离超平面 $w^* x + b^* = 0$

□ 分类决策函数

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

损失函数分析

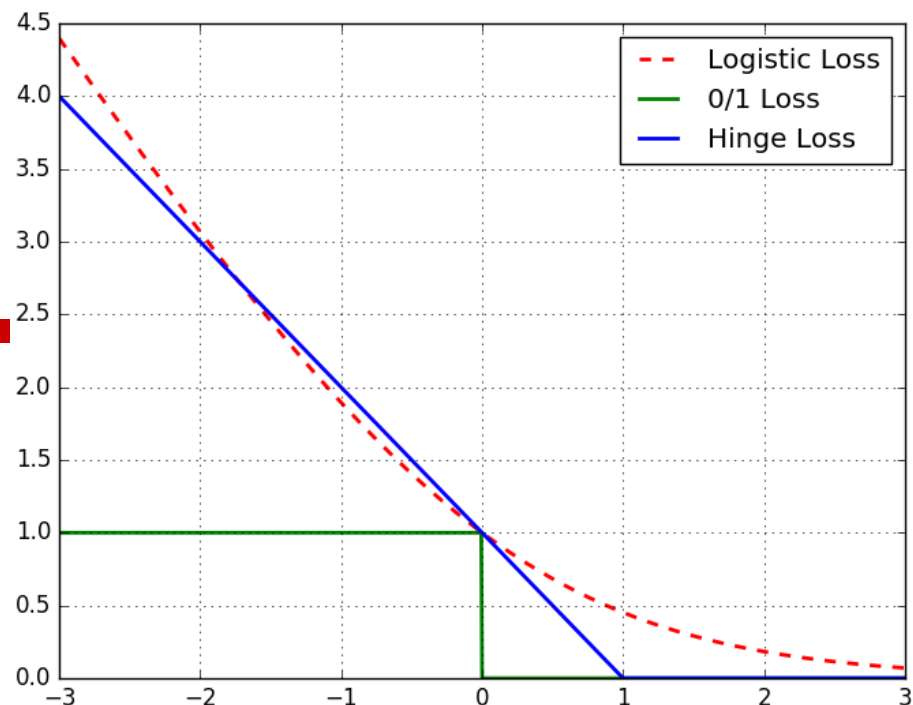
- 绿色：0/1 损失
- 蓝色：SVM Hinge 损失函数
- 红色：Logistic 损失函数



Code

```
import math
import numpy as np
import matplotlib.pyplot as plt

if __name__ == "__main__":
    x = np.array(np.linspace(start=-3, stop=3, num=1001, dtype=np.float))
    y_logit = np.log(1 + np.exp(-x)) / math.log(2)
    y_01 = x < 0
    y_hinge = 1.0 - x
    y_hinge[y_hinge < 0] = 0
    plt.plot(x, y_logit, 'r--', label='Logistic Loss', linewidth=2)
    plt.plot(x, y_01, 'g-', label='0/1 Loss', linewidth=2)
    plt.plot(x, y_hinge, 'b-', label='Hinge Loss', linewidth=2)
    plt.grid()
    plt.legend(loc='upper right')
    plt.savefig('1.png')
    plt.show()
```



核函数

- 可以使用核函数，将原始输入空间映射到新的特征空间，从而，使得原本线性不可分的样本可能在核空间可分。
 - 多项式核函数： $\kappa(x_1, x_2) = (x_1 \cdot x_2 + c)^d$
 - 高斯核RBF函数： $\kappa(x_1, x_2) = \exp(-\gamma \cdot \|x_1 - x_2\|^2)$
 - Sigmoid核函数： $\kappa(x_1, x_2) = \tanh(x_1 \cdot x_2 + c)$
- 在实际应用中，往往依赖先验领域知识/交叉验证等方案才能选择有效的核函数。
 - 没有更多先验信息，则使用高斯核函数

多项式核函数 $\kappa(x, y) = (x \cdot y)^2$

$$\kappa(x, y) = (x \cdot y)^2$$

$$\Rightarrow \left(\sum_{i=1}^n x_i y_i \right)^2$$

$$= \sum_{i=1}^n \sum_{j=1}^n x_i x_j y_i y_j$$

$$= \sum_{i=1}^n \sum_{j=1}^n (x_i x_j) (y_i y_j)$$

$$\Rightarrow \Phi(x) = \text{vec}(x_i x_j) \Big|_{i,j=1}^n$$

特殊的，若 $n=3$ ，即： $\Phi(x) =$

$$\begin{pmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \end{pmatrix}$$

多项式核 $\kappa(x, y) = (x \cdot y + c)^2$

$$\kappa(x, y) = (x \cdot y + c)^2$$

$$\Rightarrow (x \cdot y)^2 + 2cx \cdot y + c^2$$

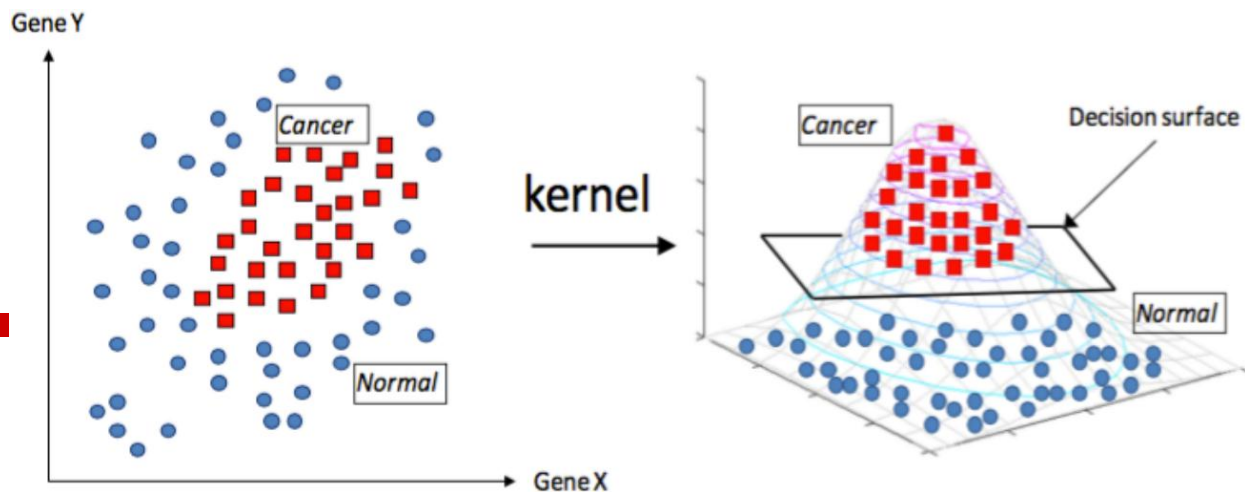
$$= \sum_{i=1}^n \sum_{j=1}^n (x_i x_j)(y_i y_j) + \sum_{i=1}^n (\sqrt{2c} x_i \cdot \sqrt{2c} x_i) + c^2$$

$$\Rightarrow \Phi(x) = \left(\text{vec}(x_i x_j) \Big|_{i,j=1}^n, \text{vec}(\sqrt{2c} x_i) \Big|_{i=1}^n, c \right)$$

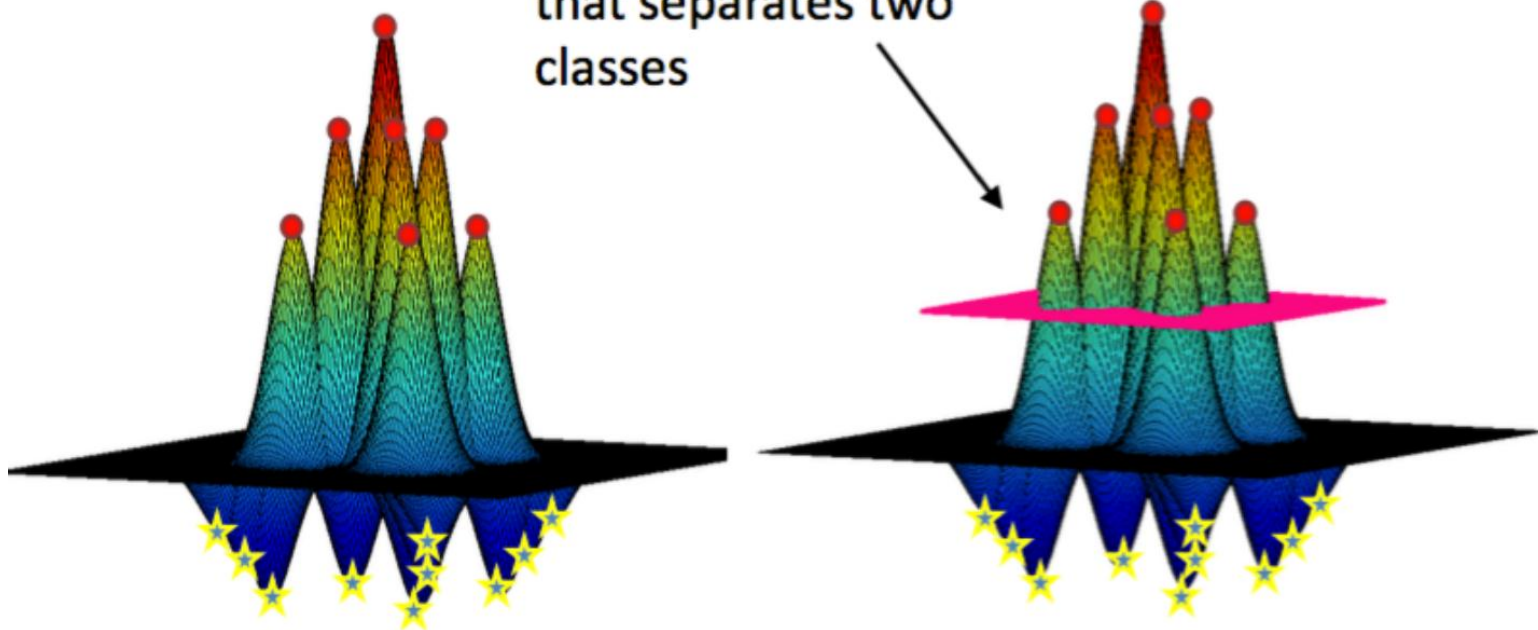
特殊的，若 $n=3$ ，即： $\Phi(x) =$

$$\begin{pmatrix} x_1 x_1 \\ x_1 x_2 \\ x_1 x_3 \\ x_2 x_1 \\ x_2 x_2 \\ x_2 x_3 \\ x_3 x_1 \\ x_3 x_2 \\ x_3 x_3 \\ \sqrt{2c} x_1 \\ \sqrt{2c} x_2 \\ \sqrt{2c} x_3 \\ c \end{pmatrix}$$

核函数映射

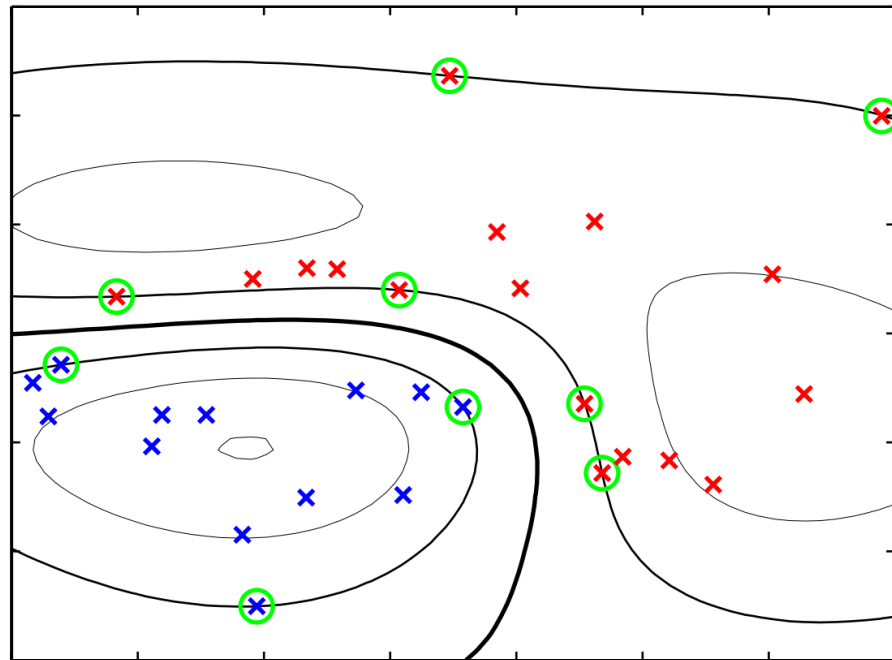


Linear hyperplane
that separates two
classes



高斯核

- 粗线是分割超“平面”
- 其他线是 $y(x)$ 的等高线
- 绿色圈点是支持向量点



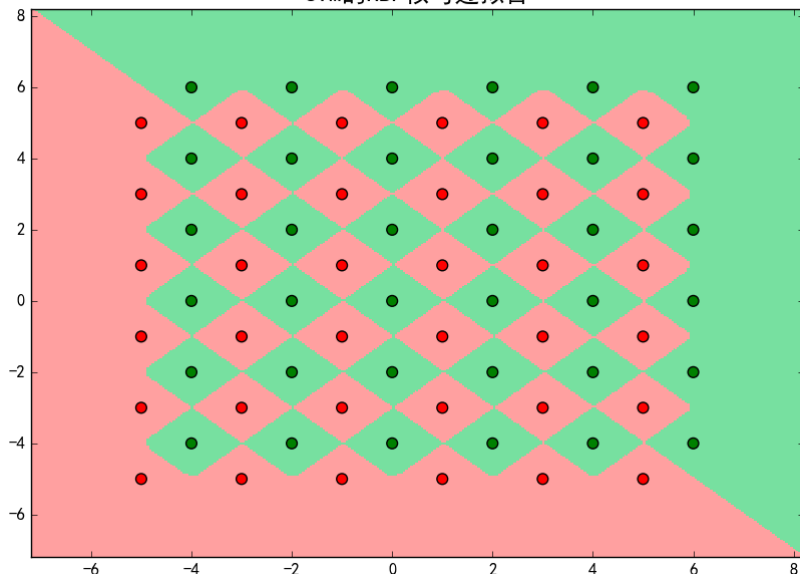
高斯核是无穷维的 $e^x = 1 + x + \frac{x^2}{2!} + \frac{x^3}{3!} + \dots + \frac{x^n}{n!} + R_n$

$$\begin{aligned}
 \kappa(x_1, x_2) &= e^{-\frac{\|x_1 - x_2\|^2}{2\sigma^2}} = e^{-\frac{(x_1 - x_2)^2}{2\sigma^2}} = e^{-\frac{x_1^2 + x_2^2 - 2x_1x_2}{2\sigma^2}} = e^{-\frac{x_1^2 + x_2^2}{2\sigma^2}} \cdot e^{\frac{x_1x_2}{\sigma^2}} \\
 &= e^{-\frac{x_1^2 + x_2^2}{2\sigma^2}} \cdot \left(1 + \frac{1}{\sigma^2} \cdot \frac{x_1x_2}{1!} + \left(\frac{1}{\sigma^2}\right)^2 \cdot \frac{(x_1x_2)^2}{2!} + \left(\frac{1}{\sigma^2}\right)^3 \cdot \frac{(x_1x_2)^3}{3!} + \dots + \left(\frac{1}{\sigma^2}\right)^n \cdot \frac{(x_1x_2)^n}{n!} + \dots \right) \\
 &= e^{-\frac{x_1^2 + x_2^2}{2\sigma^2}} \cdot \left(1 \cdot 1 + \frac{1}{1!} \frac{x_1}{\sigma} \cdot \frac{x_2}{\sigma} + \frac{1}{2!} \cdot \frac{x_1^2}{\sigma^2} \cdot \frac{x_2^2}{\sigma^2} + \frac{1}{3!} \cdot \frac{x_1^3}{\sigma^3} \cdot \frac{x_2^3}{\sigma^3} + \dots + \frac{1}{n!} \cdot \frac{x_1^n}{\sigma^n} \cdot \frac{x_2^n}{\sigma^n} + \dots \right) \\
 &= \Phi(x_1)^T \cdot \Phi(x_2)
 \end{aligned}$$

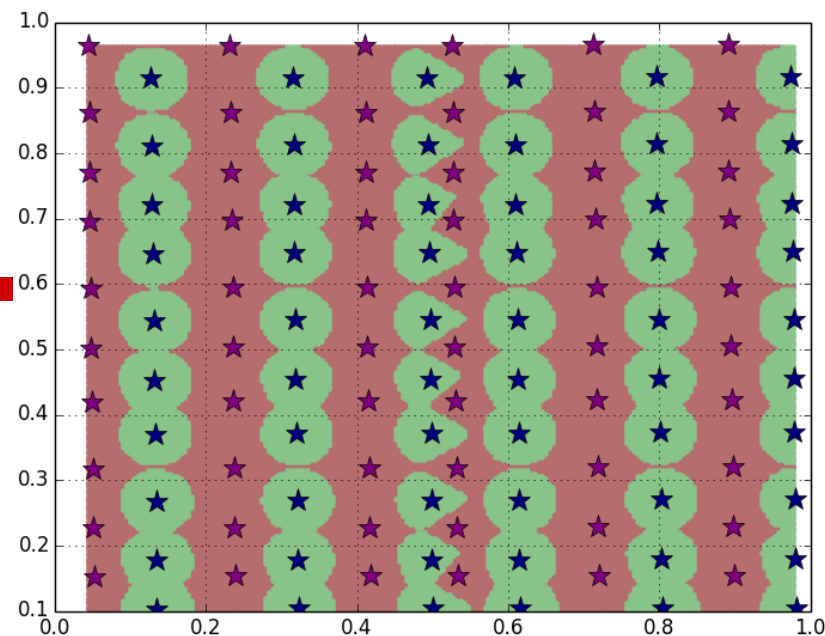
□ 其中 $\Phi(x) = e^{-\frac{x^2}{2\sigma^2}} \left(1, \sqrt{\frac{1}{1!}} \frac{x}{\sigma}, \sqrt{\frac{1}{2!}} \frac{x^2}{\sigma^2}, \sqrt{\frac{1}{3!}} \frac{x^3}{\sigma^3}, \dots, \sqrt{\frac{1}{n!}} \frac{x^n}{\sigma^n}, \dots \right)$

高斯核的分类

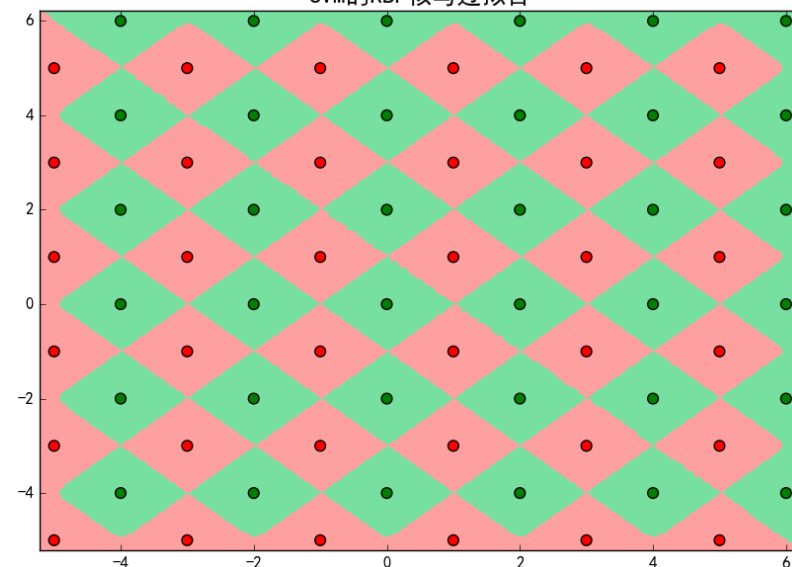
SVM的RBF核与过拟合



```
def kernel(x1, x2):  
    n = len(x2) - 1  
    s = 0  
    if kn == 0: # 线性核  
        for i in range(n):  
            s += x1[i] * x2[i]  
        return s  
    for i in range(n):  
        s += (x1[i] - x2[i]) ** 2  
    k = math.exp(-s / (2 * sigma**2))  
    return k
```



SVM的RBF核与过拟合



SVM中系数的求解：SMO

□ 序列最小最优优化

■ Sequential Minimal Optimization

□ 有多个拉格朗日乘子

□ 每次只选择其中两个乘子做优化，其他因子认为是常数。

■ 将N个解问题，转换成两个变量的求解问题：并且目标函数是凸的。

SMO: 序列最小最优化

□ 考察目标函数，假设 α_1 和 α_2 是变量，其他是定值：

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

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

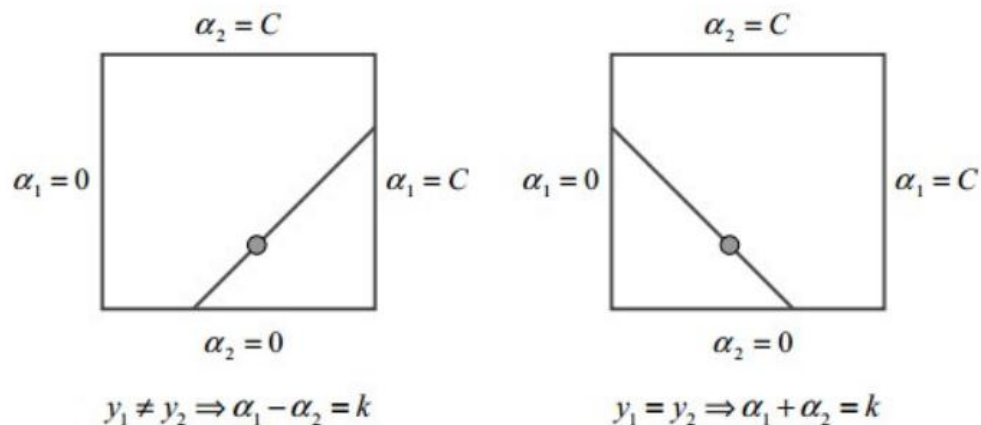
$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, N$$

$$\min_{\alpha_1, \alpha_2} W(\alpha_1, \alpha_2)$$

$$= \frac{1}{2} \kappa_{11} \alpha_1^2 + \frac{1}{2} \kappa_{22} \alpha_2^2 + y_1 y_2 \alpha_1 \alpha_2 \kappa_{12} - (\alpha_1 + \alpha_2) \quad s.t. \quad \alpha_1 y_1 + \alpha_2 y_2 = - \sum_{i=3}^N y_i \alpha_i = \zeta$$

$$+ y_1 \alpha_1 \sum_{i=3}^N y_i \alpha_i \kappa_{i1} + y_2 \alpha_2 \sum_{i=3}^N y_i \alpha_i \kappa_{i2} \quad 0 \leq \alpha_i \leq C$$

二变量优化问题



$$\begin{cases} L = \max \{0, \alpha_1 - \alpha_2\} \\ H = \min \{C, C + \alpha_1 - \alpha_2\} \end{cases}, \quad y_1 \neq y_2$$
$$\begin{cases} L = \max \{0, \alpha_1 + \alpha_2 - C\} \\ H = \min \{C, \alpha_1 + \alpha_2\} \end{cases}, \quad y_1 = y_2$$

SMO的迭代公式

□ 迭代公式:

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

$$\eta = \kappa(x_1, x_1) + \kappa(x_2, x_2) - 2\kappa(x_1, x_2) = \|\Phi(x_1) - \Phi(x_2)\|^2$$

$$E_i = g(x_i) - y_i = \left(\sum_{j=1}^N y_j \alpha_j \kappa(x_j, x_i) + b \right) - y_i, \quad i = 1, 2$$
$$\alpha_j^{new} = \alpha_j^{old} + \frac{y_j (E_i - E_j)}{\eta}$$

退出条件

$$\sum_{i=1}^N \alpha_i y_i = 0$$

$$0 \leq \alpha_i \leq C, \quad i = 1, 2, \dots, n$$

$$y_i \cdot g(x_i) = \begin{cases} \geq 1, & \{x_i | \alpha_i = 0\} // \text{落在边界外} \\ = 1, & \{x_i | 0 < \alpha_i < C\} // \text{落在边界上} \\ \leq 1, & \{x_i | \alpha_i = C\} // \text{落在边界内} \end{cases}$$

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

Code

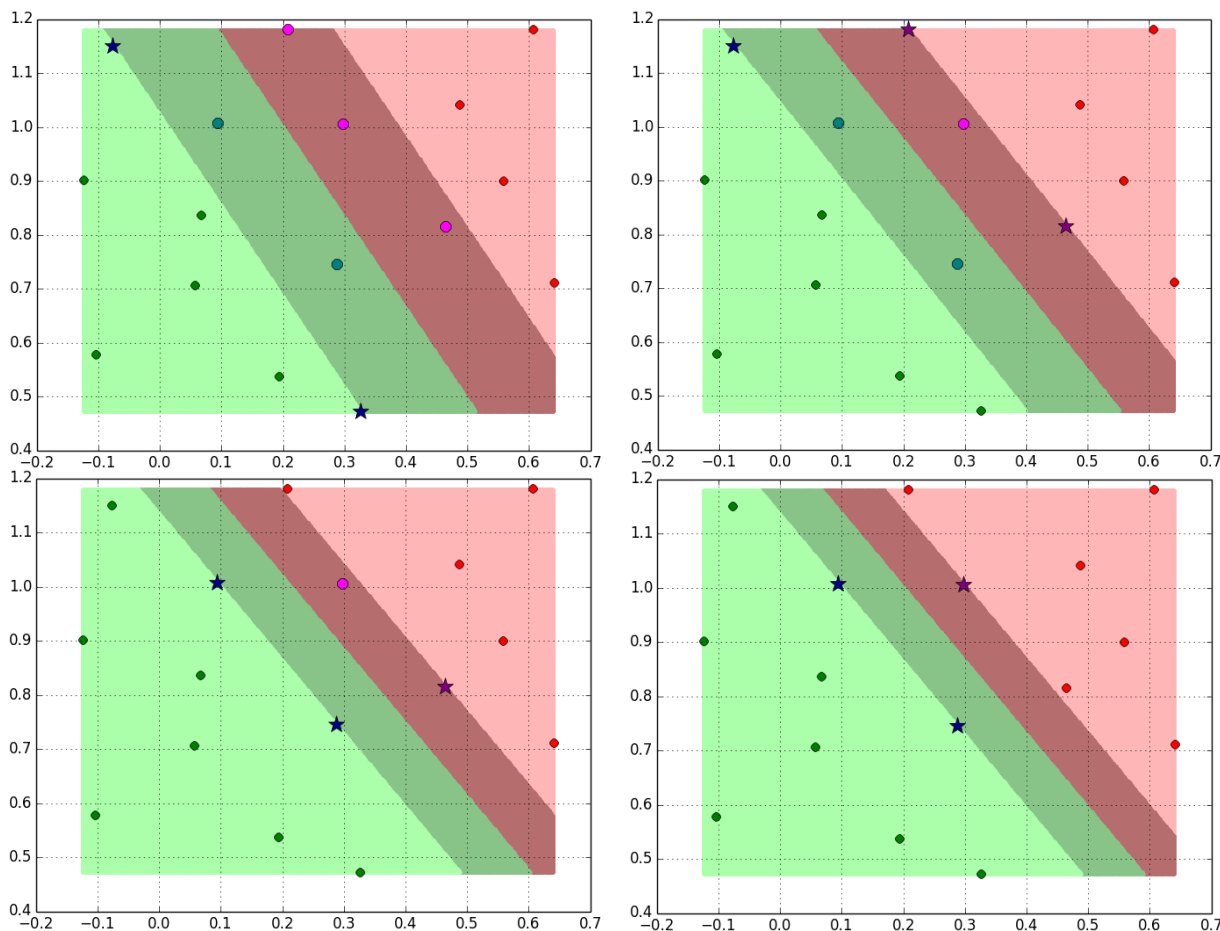
```
def update(i, j, data):
    low = 0
    high = C
    if data[i][-1] == data[j][-1]:
        low = max(0, alpha[i]+alpha[j]-C)
        high = min(C, alpha[i]+alpha[j])
    else:
        low = max(0, alpha[j]-alpha[i])
        high = min(C, alpha[j]-alpha[i]+C)
    if low == high:
        return False
    eta = kernel(data[i], data[i]) + kernel(data[j], data[j])\
        - 2*kernel(data[i], data[j])
    if is_same(eta, 0):
        return False
    ei = predict(data[i], data) - data[i][-1]
    ej = predict(data[j], data) - data[j][-1]
    alpha_j = alpha[j] + data[j][-1] * (ei - ej) / eta
    if alpha_j == alpha[j]:
        return False
    if alpha_j > high:
        alpha_j = high
    elif alpha_j < low:
        alpha_j = low
    alpha[i] += (alpha[j] - alpha_j) * data[i][-1] * data[j][-1]
    alpha[j] = alpha_j
    return True
```

Code

```
def update_b(i, j, data):
    global b
    bi = b + data[i][-1] - predict(data[i], data)
    bj = b + data[j][-1] - predict(data[j], data)
    if C > alpha[i] > 0:
        return bi
    elif C > alpha[j] > 0:
        return bj
    return (bi + bj) / 2

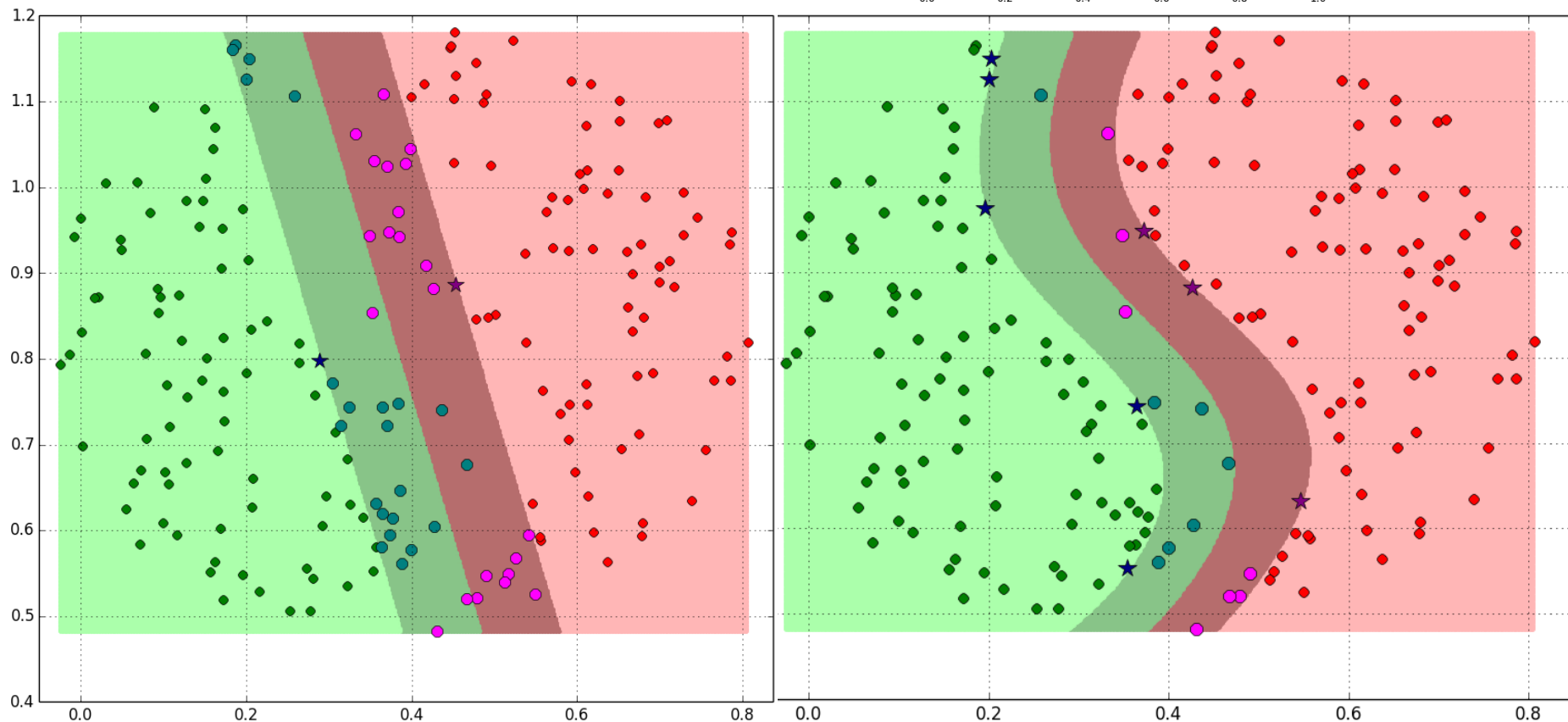
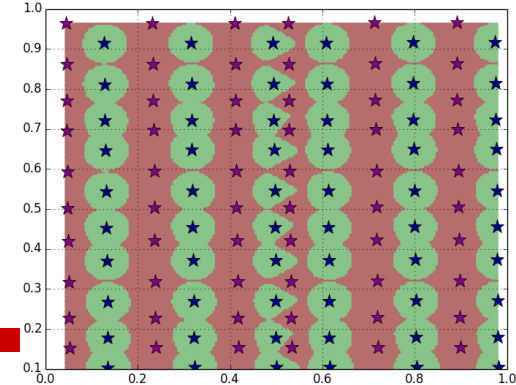
def smo(data):
    m = len(data)
    global b
    for time in range(5000):
        no_change = 0
        i = select_first(data)
        if i == -1:
            break
        j = select_second(i, m)
        if not update(i, j, data):
            no_change += 1
            continue
        b = update_b(i, j, data)
        print time, b
        if no_change > 100:
            break
```

惩罚因子的影响



10	20
50	100

高斯核函数的影响



总结与思考

□ SVM可以用来划分多类别吗？

- 直接多分类
- 1 vs rest / 1 vs 1

□ SVM和Logistic回归的比较

- 经典的SVM，直接输出类别，不给出后验概率；
- Logistic回归，会给出属于哪个类别的后验概率。
- **重点：**二者目标函数的异同

□ SVM框架下引入Logistic函数：输出条件后验概率

□ SVM用于回归问题：SVR；

□ 体会SVM的目标函数的建立过程

- 原始目标函数和Lagrange函数有什么联系？

附：优化问题的鞍点解释和KKT条件

- 为表述方便，假设没有等式约束，只考虑不等式约束，结论可方便的扩展到等式约束。
- 假设 x_0 不可行，即存在某些 i ，使得 $f_i(x) > 0$ 。则选择 $\lambda_i \rightarrow \infty$ ，对于其他乘子， $\lambda_j = 0, j \neq i$
- 假设 x_0 可行，则有 $f_i(x) \leq 0, (i=1,2,\dots,m)$ ，选择

$$\lambda_i = 0, i = 1, 2, \dots, m$$

- 有：

$$\sup_{\lambda \geq 0} L(x, \lambda) = \sup_{\lambda \geq 0} \left(f_0(x) + \sum_{i=1}^m \lambda_i f_i(x) \right) = \begin{cases} f_0(x) & f_i(x) \leq 0, i = 1, 2, \dots, m \\ \infty & \text{otherwise} \end{cases}$$

鞍点：最优点

□ 而原问题是： $\inf_x f_0(x)$

□ 从而，原问题的本质为： $\inf_x \sup_{\lambda \geq 0} L(x, \lambda)$

□ 而对偶问题，是求对偶函数的最大值，即：

$$\sup_{\lambda \geq 0} \inf_x L(x, \lambda)$$

□ 而：

$$\sup_{\lambda \geq 0} \inf_x L(x, \lambda) \leq \inf_x \sup_{\lambda \geq 0} L(x, \lambda)$$

$$\text{证明: } \max_x \min_y f(x, y) \leq \min_y \max_x f(x, y)$$

□ 对于任意的 $(x, y) \in \text{dom} f$

$$f(x, y) \leq \max_x f(x, y)$$

$$\Rightarrow \min_y f(x, y) \leq \min_y \max_x f(x, y)$$

$$\Rightarrow \max_x \min_y f(x, y) \leq \min_y \max_x f(x, y)$$

强对偶条件

□ 若要对偶函数的最大值即为原问题的最小值，考察需要满足的条件：

$$\begin{aligned} f_0(x^*) &= g(\lambda^*, \nu^*) \\ &= \inf_x \left(f_0(x) + \sum_{i=1}^m \lambda_i^* f_i(x) + \sum_{i=1}^p \nu_i^* h_i(x) \right) \\ &\leq f_0(x^*) + \sum_{i=1}^m \lambda_i^* f_i(x^*) + \sum_{i=1}^p \nu_i^* h_i(x^*) \\ &\leq f_0(x^*). \end{aligned}$$

Karush-Kuhn-Tucker (KKT)条件

$$\begin{aligned}f_0(x^*) &= g(\lambda^*, \nu^*) \\&= \inf_x \left(f_0(x) + \sum_{i=1}^m \lambda_i^* f_i(x) + \sum_{i=1}^p \nu_i^* h_i(x) \right) \\&\leq f_0(x^*) + \sum_{i=1}^m \lambda_i^* f_i(x^*) + \sum_{i=1}^p \nu_i^* h_i(x^*) \\&\leq f_0(x^*).\end{aligned}$$

$$\begin{aligned}f_i(x^*) &\leq 0, \quad i = 1, \dots, m \\h_i(x^*) &= 0, \quad i = 1, \dots, p \\\lambda_i^* &\geq 0, \quad i = 1, \dots, m \\\lambda_i^* f_i(x^*) &= 0, \quad i = 1, \dots, m\end{aligned}$$

$$\nabla f_0(x^*) + \sum_{i=1}^m \lambda_i^* \nabla f_i(x^*) + \sum_{i=1}^p \nu_i^* \nabla h_i(x^*) = 0$$

参考文献

- Corinana Cortes, Vladimir Vapnik. *Support-Vector Networks*. Machine Learning, 20, 273-297, 1995
- Christopher M. Bishop, *Pattern Recognition and Machine Learning*, Springer Press, 2006
- 李航, 统计学习方法, 清华大学出版社, 2012
- Stephen Boyd, Lieven Vandenberghe. *Convex Optimization*, Cambridge University Press. 2004
 - 中译本: 王书宁, 许鋆, 黄晓霖, 凸优化, 清华大学出版社, 2013
- Charlie Frogner. *Support Vector Machines*. 2011
- John C. Platt. *Sequential Minimal Optimization: A Fast Algorithm for Training Support Vector Machines*. 1998
- Andrew W. Moore. *Support Vector Machines*, 2001

作业

- 核函数是什么？高斯核映射到无穷维是怎么回事？
- 怎么理解SVM的损失函数？
- 使用高斯核函数，请描述SVM的参数 C 和 σ 对分类器的影响。

我们在这里

□ <http://wenda.ChinaHadoop.cn>

■ 视频/课程/社区

□ 微博

■ @ChinaHadoop

■ @邹博_机器学习

□ 微信公众号

■ 小象学院

■ 大数据分析挖掘

互联网新技术在线教育领航者

小象问答 搜索标题、用户 全站内容搜索 提问 首页 动态 发现 话题 通知

全部 招聘求职 机器学习 大数据平台技术 DCon 大数据行业应用 NoSQL数据库 数据科学 江湖救急

发现 最新 推荐 热门 等待回复

graphviz has no attribute 'write' 贡献
邹博 回复了问题 • 2 人关注 • 1 个回复 • 3 次浏览 • 2017-04-09 15:48

sklearn中如何理解Pipeline机制 贡献
数据分析与数据挖掘 邹博 回复了问题 • 2 人关注 • 1 个回复 • 28 次浏览 • 2017-04-09 15:39

关于9.Logistic回归的ppt中第9页的对数线性函数 贡献
机器学习 邹博 回复了问题 • 3 人关注 • 3 个回复 • 39 次浏览 • 2017-04-09 15:35

关于“贝叶斯估计中，最大后验概率估计就是结构化风险最小化的例子：当模型是条件概率分布，损失函数为对数损失函数，模型的复杂度由模型的先验概率表示，结构化风险最小化就等价于最大后验概率估计” 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 26 次浏览 • 2017-04-09 15:27

关于连续值的预测 贡献
咨询 邹博 回复了问题 • 2 人关注 • 1 个回复 • 31 次浏览 • 2017-04-09 15:24

拉格朗日对偶函数为什么一定是凸函数 贡献
数据科学 邹博 回复了问题 • 2 人关注 • 2 个回复 • 26 次浏览 • 2017-04-09 15:20

梯度下降公式中的斯堪J 是 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 29 次浏览 • 2017-04-09 15:17

深度学习适合做预测吗？ 贡献
深度学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 27 次浏览 • 2017-04-09 15:15

关于6.4PCA_FeatureSelection.py中plt.legend的参数疑问 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 28 次浏览 • 2017-04-09 15:04

@邹博 有哪些可以下载数据源的网站？ 贡献
数据分析与数据挖掘 邹博 回复了问题 • 4 人关注 • 1 个回复 • 31 次浏览 • 2017-04-09 14:53

LDA主题模型 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 29 次浏览 • 2017-04-09 14:45

代码10.6bagging_ridged老师提到了采样率设为0.2能够使峰值部分的数据被体现出来。这是为什么呢？ 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 22 次浏览 • 2017-04-09 14:26

GraphViz's executables not found 贡献
机器学习 邹博 回复了问题 • 3 人关注 • 2 个回复 • 23 次浏览 • 2017-04-09 13:47

决策树中关于feature_importances代码的问题 贡献
机器学习 邹博 回复了问题 • 2 人关注 • 1 个回复 • 6 次浏览 • 2017-04-09 13:11

专题
招聘求职
大数据行业应用
数据科学
系统与编程
云计算技术

热门话题 更多 >
机器学习 907 个问题, 230 人关注
spark 387 个问题, 172 人关注
hadoop 1059 个问题, 155 人关注
python数据分析 171 个问题, 28 人关注
数据分析与数据挖掘 54 个问题, 111 人关注

热门用户 更多 >
小心巴 14 个问题, 0 次赞同
又又V 45 个问题, 22 次赞同
铁甲无声 10 个问题, 0 次赞同
带刀锦衣卫 13 个问题, 0 次赞同

感谢大家！

恳请大家批评指正！