

CDA大数据分析师就业班 之 Python **犯器学习**

Beautiful is better than ugly.
Explicit is better than implicit. Simple is better than complex. Complex is better than complicated. Flat is better than nested. Sparse is better than dense.
Readability counts. Special cases aren't special enough to break the rules.

racticality beats purity. Frost should never

Although practicality beats purity. Errors should never pass silently. Unless explicitly silenced. In the face of ambiguity, refuse the temptation to guess. There should be one—and preferably only one—obvious way to do it. Although that way may not be obvious at first unless you're Dutch. Now is better than never. Although never is often better than right now. If the implementation is hard to explain, it's a bad

dea. If the implementation is easy to explain, it may be a good idea.

Namespaces are one honking great leu idea — let's do more of those!

idea, It fire implementation idea, It fire implementation it seasy to easy to good idea.

A management in a ma

Although practicality beats purity, Errors should never pass silently. Unless explicitly silenced. In the face of ambiguity, refuse the temptation to guess. There should be one ambiguity, refuse the temptation to guess. There should be one — and preferably only one, one — obvious way to do it. Although never is often beatter than right beatter than never, though never is often beatter than right now. If the implementation is hard to explain, it's a bad now. If the implementation is hard to explain, it's a bad

Explicit is better than ugly, simple stample stample stample stample stamples are than complex. Complex is better than the complex is better than the complex are is better than the complex is petter than the complex is petter in the complex is petter in the complex in the complex in the complex is petter in the complex in the complex

python

Z

覃秉丰

支持向量机 SVM(Support Vector Machines)

SVM



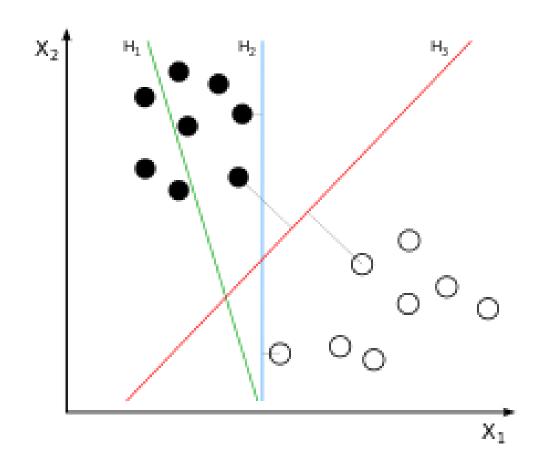
最早是由 Vladimir N. Vapnik 和 Alexey Ya. Chervonenkis 在1963年提出

目前的版本(soft margin)是由Corinna Cortes 和 Vapnik在1993年提出,并在1995年发表

深度学习(2012)出现之前,SVM被认为机器学习中近十几年来最成功,表现最好的算法

SVM

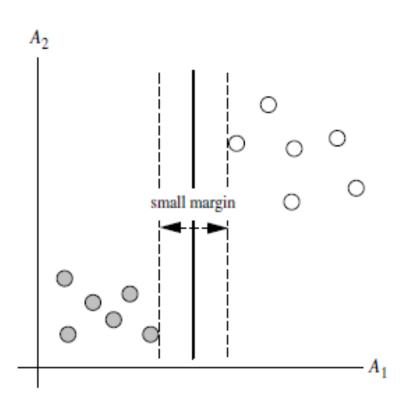


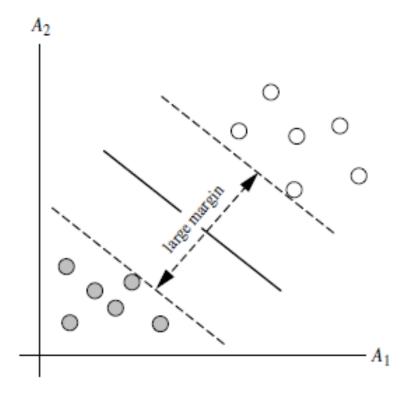


SVM



SVM寻找区分两类的超平面 (hyper plane), 使边际(margin)最大





向量内积



$$x = \begin{cases} x_1 \\ x_2 \\ \dots \\ x_n \end{cases} \qquad y = \begin{cases} y_1 \\ y_2 \\ \dots \\ y_n \end{cases}$$

向量内积: $x \cdot y = x_1 y_1 + x_2 y_2 + \dots + x_n y_n$

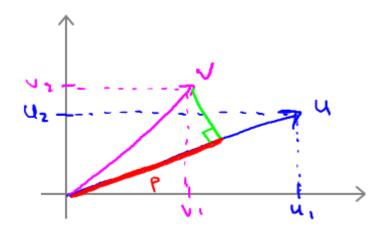
向量内积: $x \cdot y = ||x||||y||\cos(\theta)$

范数:
$$||x|| = \sqrt{x \cdot x} = \sqrt{x_1^2 + x_2^2 + \dots + x_n^2}$$

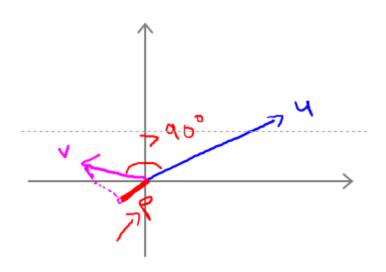
当 $||x|| \neq 0$, $||y|| \neq 0$ 时,可以求余弦相似度: $cos\theta = \frac{x \cdot y}{||x||||y||}$

向量内积





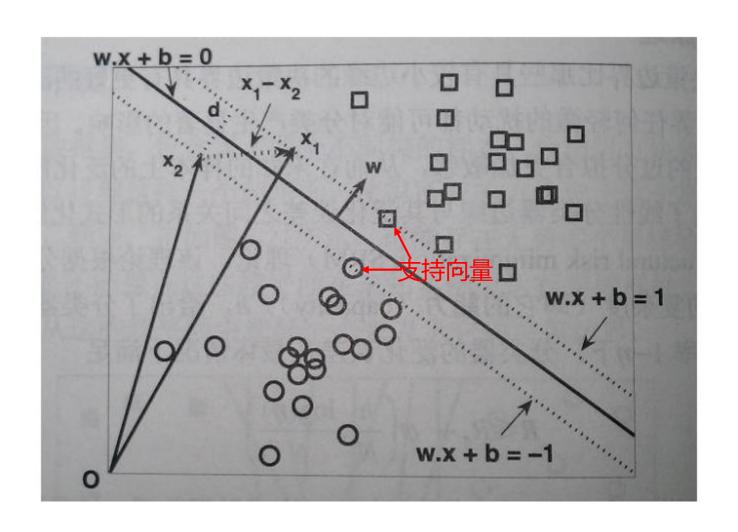
向量内积: $v \cdot u > 0$



向量内积: $v \cdot u < 0$

SVM分类





一些计算



$$w \cdot x + b = 1$$
$$w \cdot x + b = -1$$

$$w \cdot x + b = 2$$
$$w \cdot x + b = -3$$

$$w \cdot x_1 + b = 1$$
$$w \cdot x_2 + b = -1$$

$$w \cdot (x_1 - x_2) = 2$$

$$||w|| ||(x_1 - x_2)|| \cos(\theta) = 2$$

$$||w|| *d=2$$

$$d = \frac{2}{||w||}$$

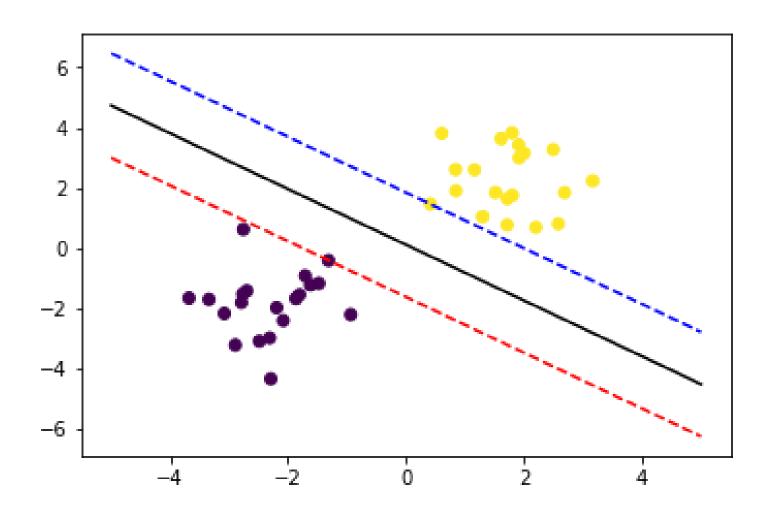
SVM简单例子



Talk is cheap Show me the

SVM-线性分类





SVM-非线性分类(练习)



Talk is cheap Show me the

转化为凸优化问题



$$w \cdot x + b \ge 1$$
,则分类y=1 $y(w \cdot x + b) \ge 1$ $w \cdot x + b \le -1$,则分类y=-1

求
$$d = \frac{2}{\|w\|}$$
最大值,

也就是求
$$min \frac{||w||^2}{2}$$

凸优化问题



1. 无约束优化问题:

 $\min f(x)$

-费马定理

2.带等式约束的优化问题:

 $\min f(x)$

-拉格朗日乘子法:

$$s.t. h_i(x) = 0, i = 1, 2, \dots n$$

$$\mathcal{L}(\mathbf{x}, \lambda) = f(\mathbf{x}) + \sum_{i=1}^{n} \lambda_i h_i(\mathbf{x})$$

3.带不等式约束的优化问题:

 $\min f(x)$

-KKT条件

$$s.t. h_i(\mathbf{x}) = 0, \qquad i = 1, 2, \dots, n$$

$$i=1,2,\cdots,n$$

$$g_i(\mathbf{x}) \leq 0, \qquad i = 1, 2, \cdots, k$$

$$i=1,2,\cdots,k$$

$$\mathcal{L}(x,\lambda,v) = f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{i=1}^{n} v_i h_i(x)$$

广义拉格朗日乘子法



目标函数 α拉格朗日乘子 约束条件

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

$$\frac{\partial L}{\partial w} = 0 \to w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

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

跟岭回归和LASSO类似



岭回归代价函数:

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)})^{2} + \lambda \sum_{j=1}^{n} \theta_{j}^{2} \right]$$

 λ 的值可以用于限制 $\sum_{j=1}^{n} \theta_j^2 \leq t$

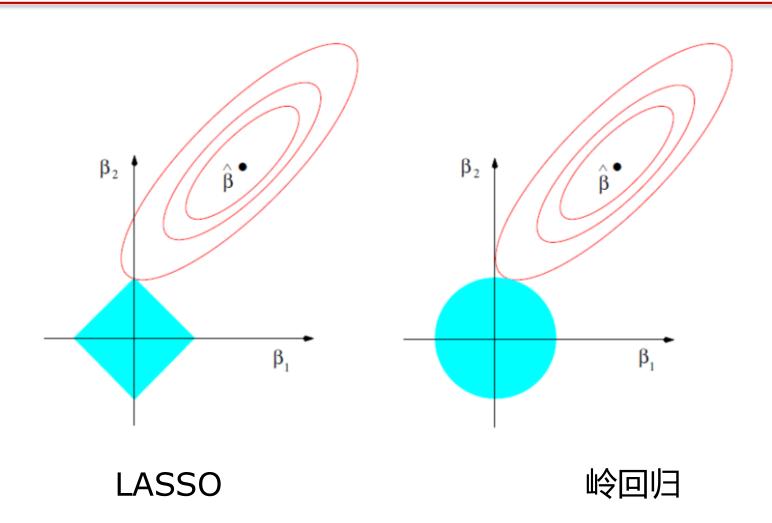
LASSO代价函数:

$$J(\theta) = \frac{1}{2m} \left[\sum_{i=1}^{m} (h_{\theta}(x^{(i)}) - y^{(i)})^{2} + \lambda \sum_{j=1}^{n} |\theta_{j}| \right]$$

 λ 的值可以用于限制 $\sum_{j=1}^{n} |\theta_j| \leq t$

跟岭回归和LASSO类似





Karush-Kuhn-Tucker最优化条件(KKT条件)



拉格朗日乘子法的一种推广,可以处理有不等号的约束条件。

$$\min f(\mathbf{x})$$
s.t. $h_i(\mathbf{x}) = 0$, $i = 1, 2, \dots, n$

$$g_i(\mathbf{x}) \le 0$$
, $i = 1, 2, \dots, k$

$$\mathcal{L}(x,\lambda,v) = f(x) + \sum_{i=1}^{k} \lambda_i g_i(x) + \sum_{i=1}^{n} v_i h_i(x)$$



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

上述问题可以改写成:

$$\min_{\boldsymbol{w},b} \max_{\alpha_i \geq 0} \mathcal{L}(\boldsymbol{w},b,\boldsymbol{\alpha}) = p^*$$

可以等价为下列对偶问题:

$$\max_{\alpha_i \ge 0} \min_{\boldsymbol{w}, b} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = d^*$$



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

$$\frac{\partial L}{\partial w} = 0 \to w = \sum_{i=1}^{n} \alpha_i y_i x_i$$

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

把w和b消除了
$$L(w,b,a) = \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j$$



$$\max_{\alpha_i \ge 0} \min_{\boldsymbol{w}, b} \mathcal{L}(\boldsymbol{w}, b, \boldsymbol{\alpha}) = \max_{\boldsymbol{\alpha}} \left[\sum_{i=1}^k \alpha_i - \frac{1}{2} \sum_{i,j=1}^k \alpha_i \alpha_j y_i y_j (x_i)^T x_j \right]$$

s. t.
$$\sum_{i=1}^{k} \alpha_i y_i = 0$$
, $\alpha_i \ge 0$, $i = 1, 2, \dots, n$

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

$$s.t.\sum_{i=1}^{k} \alpha_i y_i = 0, \qquad \alpha_i \ge 0, i = 1, 2, \dots, n$$



由此可以求出最优解 α^* ,求出该值后将其带入可以得到:

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

$$b^* = y_i - (w^*)^T x_i$$

SMO算法



Microsoft Research的John C. Platt在1998年提出针对线性SVM和数据稀疏时性能更优

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

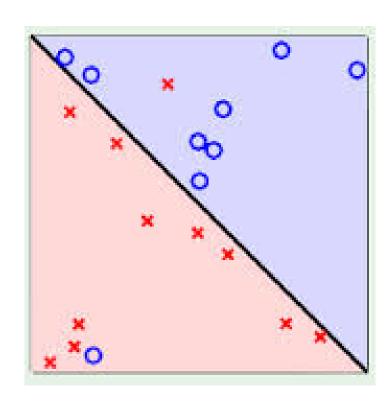
$$s.t.\sum_{i=1}^{k} \alpha_i y_i = 0, \qquad \alpha_i \ge 0, i = 1, 2, \cdots, n$$

s.t.,
$$C \ge \alpha_i \ge 0$$
, $i = 1,...,n$

基本思路是先根据约束条件随机给 α 赋值。然后每次 选取两个 α ,调节这两个 α 使得目标函数最小。然后再 选取两个 α ,调节 α 使得目标函数最小。以此类推

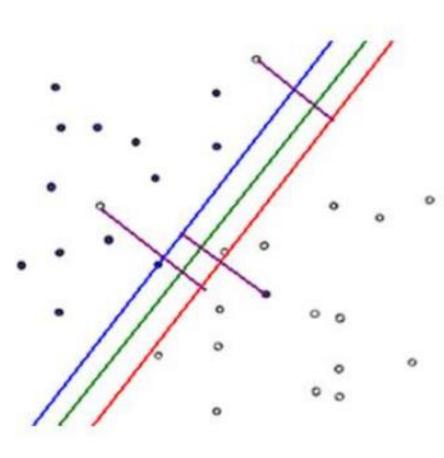
线性不可分的情况





松弛变量与惩罚函数





$$y_i(w_i \cdot x_i + b) \ge 1 - \varepsilon_i, \varepsilon_i \ge 0$$

约束条件没有体现错误分类 的点要尽量接近分类边界

$$min\frac{\|w\|^2}{2} + C\sum_{i=1}^n \varepsilon_i$$

使得分错的点越少越好,距 离分类边界越近越好

线性不可分情形下的对偶问题

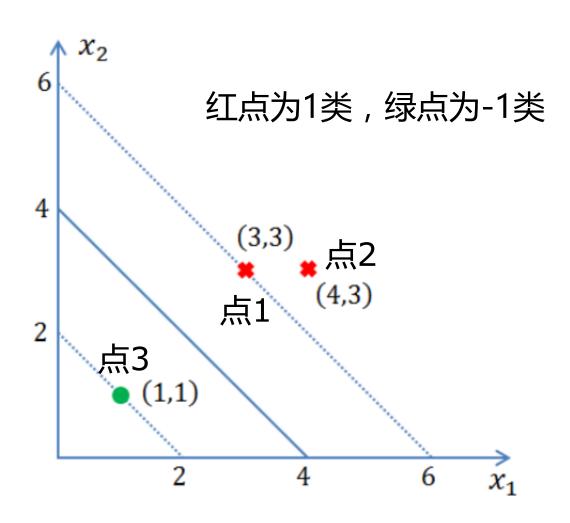


$$\min_{\alpha} \left[\frac{1}{2} \sum_{i,j=1}^k \alpha_i \alpha_j y_i y_j (x_i)^T x_j - \sum_{i=1}^k \alpha_i \right] = \min_{\alpha} \left[\frac{1}{2} \sum_{i,j=1}^k \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^k \alpha_i \right]$$

$$s.t.\sum_{i=1}^{k} \alpha_i y_i = 0, \qquad \alpha_i \ge 0, i = 1, 2, \dots, n$$

s.t.,
$$C \ge \alpha_i \ge 0, i = 1,...,n$$







可知目标函数为

$$\min_{\alpha} f(\alpha), \qquad s.t. \, \alpha_1 + \alpha_2 - \alpha_3 = 0, \qquad \alpha_i \ge 0, i = 1,2,3$$

其中

$$f(\boldsymbol{\alpha}) = \frac{1}{2} \sum_{i,j=1}^{3} \alpha_i \alpha_j y_i y_j (x_i \cdot x_j) - \sum_{i=1}^{3} \alpha_i$$

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

然后,将 $\alpha_3 = \alpha_1 + \alpha_2$ 带入目标函数,得到一个关于 α_1 和 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 和 α_2 求偏导数并令其为0,易知 $s(\alpha_1,\alpha_2)$ 在点(1.5,-1)处取极值。而该点不满足 $a_i \geq 0$ 的约束条件,于是可以推断最小值在边界上达到。经计算当 $\alpha_1 = 0$ 时, $s(\alpha_1 = 0,\alpha_2 = 2/13) = -0.1538$;当 $\alpha_2 = 0$ 时, $s(\alpha_1 = 1/4,\alpha_2 = 0) = -0.25$ 。于是 $s(\alpha_1,\alpha_2)$ 在 $\alpha_1 = 1/4$, $\alpha_2 = 0$ 时取得最小值,此时亦可算出 $\alpha_3 = \alpha_1 + \alpha_2 = 1/4$ 。因为 α_1 和 α_3 不等于0,所以对应的点 α_1 和 α_2 就应该是支持向量。



进而可以求得

$$\mathbf{w}^* = \sum_{i=1}^3 \alpha_i^* y_i x_i = \frac{1}{4} \times (3,3) - \frac{1}{4} \times (1,1) = \left(\frac{1}{2}, \frac{1}{2}\right)$$

即 $w_1 = w_2 = 0.5$ 。进而有

$$b^* = 1 - (w_1, w_2) \cdot (3,3) = -2$$

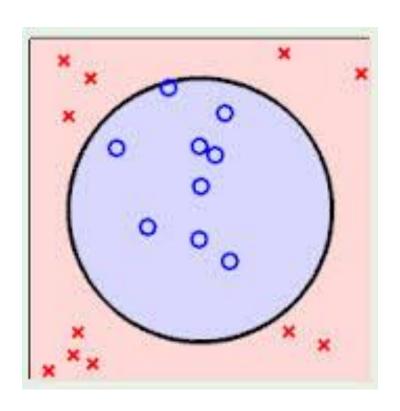
因此最大间隔分类超平面为

$$\frac{1}{2}x_1 + \frac{1}{2}x_2 - 2 = 0$$

分类决策函数为

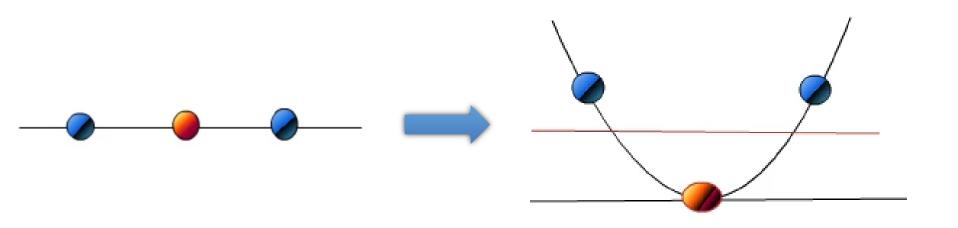
$$f(\mathbf{x}) = sign\left(\frac{1}{2}x_1 + \frac{1}{2}x_2 - 2\right)$$



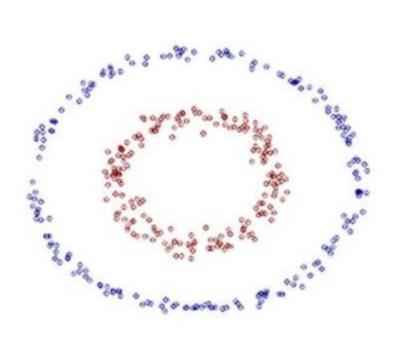


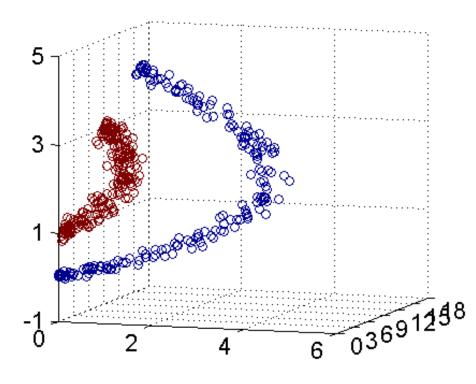


把低维空间的非线性问题映射到高维空间,变成求解线性问题

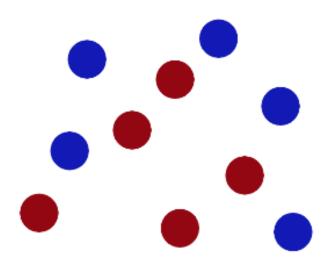


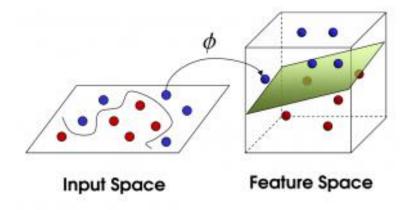






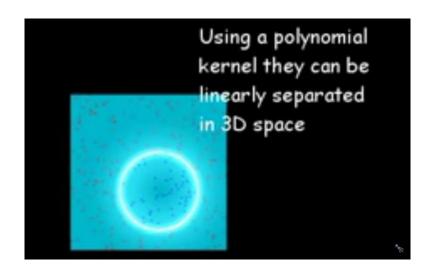


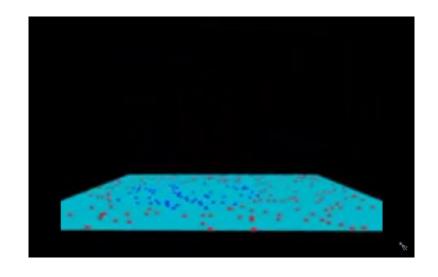




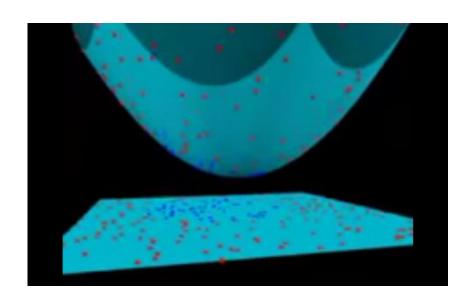


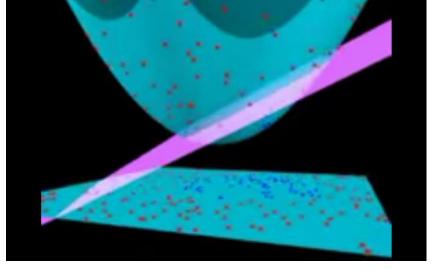
https://v.qq.com/x/page/k05170ntgzc.html



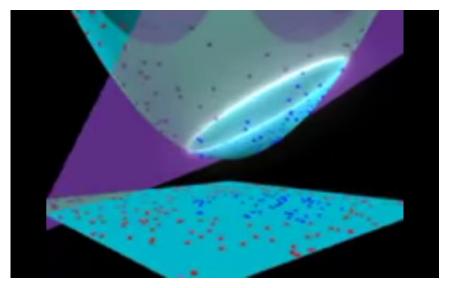


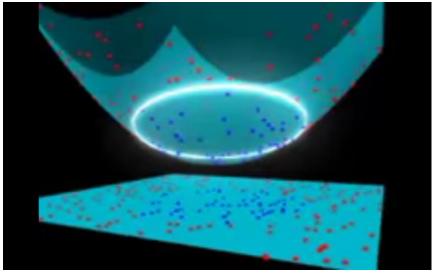




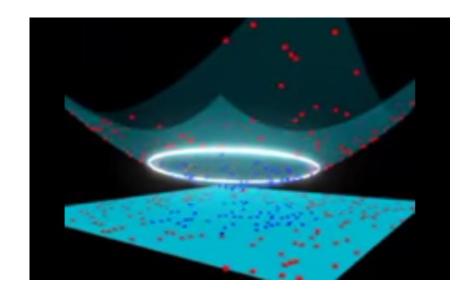


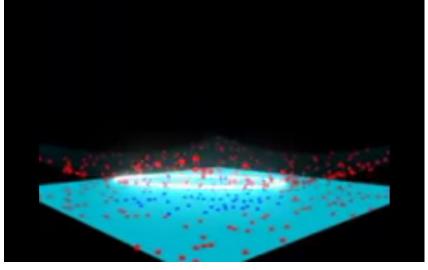




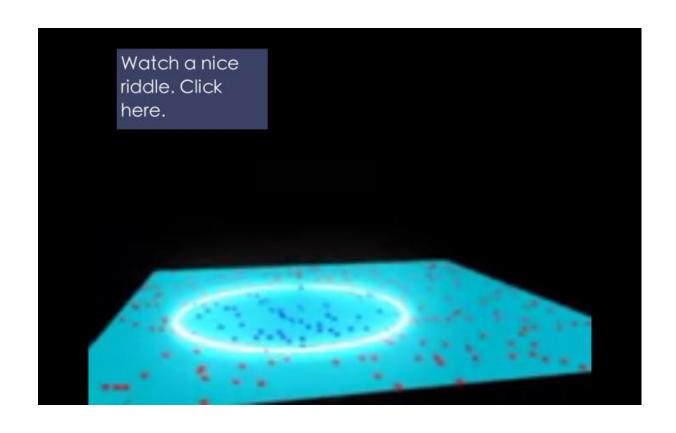












SVM-低维映射高维



Talk is cheap Show me the

映射举例



3维输入向量: $X = (x_1, x_2, x_3)$

转化到6维空间 Z 中去:

$$\phi_1(X) = x_1, \ \phi_2(X) = x_2, \ \phi_3(X) = x_3, \ \phi_4(X) = (x_1)^2, \ \phi_5(X) = x_1x_2, \ \text{and} \ \phi_6(X) = x_1x_3.$$

新的决策超平面 :d(Z) = WZ + b, 其中W和Z是向量,这个超平面是线性的,解出W和b之后,并且带回原方程:

$$d(\mathbf{Z}) = w_1 x_1 + w_2 x_2 + w_3 x_3 + w_4 (x_1)^2 + w_5 x_1 x_2 + w_6 x_1 x_3 + b$$

= $w_1 z_1 + w_2 z_2 + w_3 z_3 + w_4 z_4 + w_5 z_5 + w_6 z_6 + b$

存在的问题



$$min_{\alpha} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \right], \sum_{i=1}^{k} \alpha_i y_i = 0, C \ge \alpha_i \ge 0$$

$$\min_{\alpha} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \emptyset(x_i)^T \emptyset(x_j) \right], \sum_{i=1}^{k} \alpha_i y_i = 0, C \ge \alpha_i \ge 0$$

1.维度灾难

红色的地方要使用映射后的样本向量做内积 假如最初的特征是n维的,我们把它映射到n²维,然后 再计算。这样需要的时间从原来的的O(n),变成了O(n²)

2.如何选择合理的非线性转换?

引入核函数



我们可以构造核函数使得运算结果等同于非线性映射,同时运算量要远远小于非线性映射。

$$K(X_i, X_j) = \phi(X_i) \cdot \phi(X_j)$$

$$h$$
 次多项式核函数 : $K(X_i, X_j) = (X_i, X_j + 1)^n$ 高斯径向基函数核函数 : $K(X_i, X_j) = e^{-\|X_i - X_j\|^2/2\sigma^2}$ S 型核函数 : $K(X_i, X_j) = \tanh(\kappa X_i \cdot X_j - \delta)$

核函数举例



假设定义两个向量: x = (x1, x2, x3); y = (y1, y2, y3) 定义高维映射方程: f(x) = (x1x1, x1x2, x1x3, x2x1, x2x2, x2x3, x3x1, x3x2, x3x3) 假设x = (1, 2, 3), y = (4, 5, 6). f(x) = (1, 2, 3, 2, 4, 6, 3, 6, 9) f(y) = (16, 20, 24, 20, 25, 36, 24, 30, 36) 求内积<f(x), f(y)> = 16 + 40 + 72 + 40 + 100+ 180 + 72 + 180 + 324 = 1024

定义核函数: $K(x,y) = (\langle f(x), f(y) \rangle)^2$ $K(x,y) = (4 + 10 + 18)^2 = 1024$ 同样的结果,使用核方法计算容易得多。

SVM优点



- 训练好的模型的算法复杂度是由支持向量的个数决定的, 而不是由数据的维度决定的。所以SVM不太容易产生 overfitting
- SVM训练出来的模型完全依赖于支持向量(Support Vectors),即使训练集里面所有非支持向量的点都被去 除,重复训练过程,结果仍然会得到完全一样的模型。
- 一个SVM如果训练得出的支持向量个数比较小,SVM 训练出的模型比较容易被泛化。

LFW人脸数据集



http://vis-www.cs.umass.edu/lfw/

Labeled Faces in the Wild



Menu

- LFW Home
 - o Mailing
 - o Explore
 - o Download
 - o Train/Test
 - o Results
 - o Information
 - o Errata
 - o Reference
 - o Resources
 - o Contact
 - o Support
 - o Changes
- Part Labels
- UMass Vision

Labeled Faces in the Wild Home



NEW SURVEY PAPER:

Erik Learned-Miller, Gary B. Huang, Aruni RoyChowdhury, Haoxiang Li, and Gang Hua.

Labeled Faces in the Wild: A Survey.

In *Advances in Face Detection and Facial Image Analysis*, edited by Michal Kawulok, M. Emre Celebi, and Bogdan Smolka, Springer, pages 189-248, 2016.

[Springer Page] [Draft pdf]

SVM-人脸识别



Talk is cheap Show me the