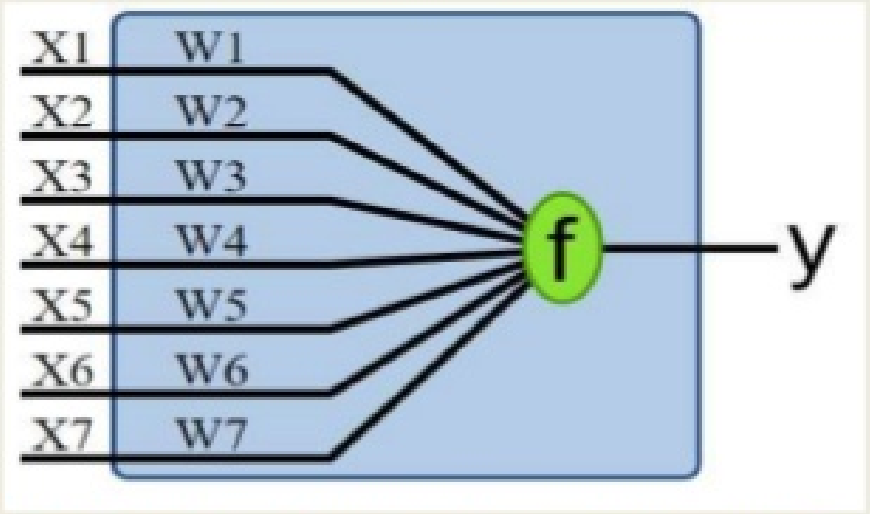
《数据挖掘技术》

⭐ CH02 感知机

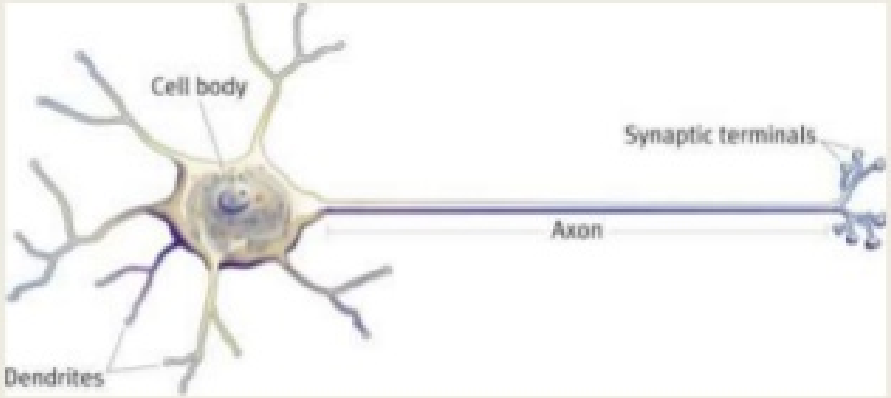
➡️ Created by *Wang JingHui*

➡️ Version *4.0*

1957: Rosenblatt's "Perceptron"



https://upload.wikimedia.org/wikipedia/commons/3/31/Perceptron.svg



http://bio3520.nicerweb.com/Locked/chap/ch03/3_11-neuron.jpg

"The embryo of an electronic computer that [the Navy] expects will be able to walk, talk, see, write, reproduce itself and be conscious of its existence."

https://en.wikipedia.org/wiki/Perceptron

http://www.rutherfordjournal.org/article040101.html

历史

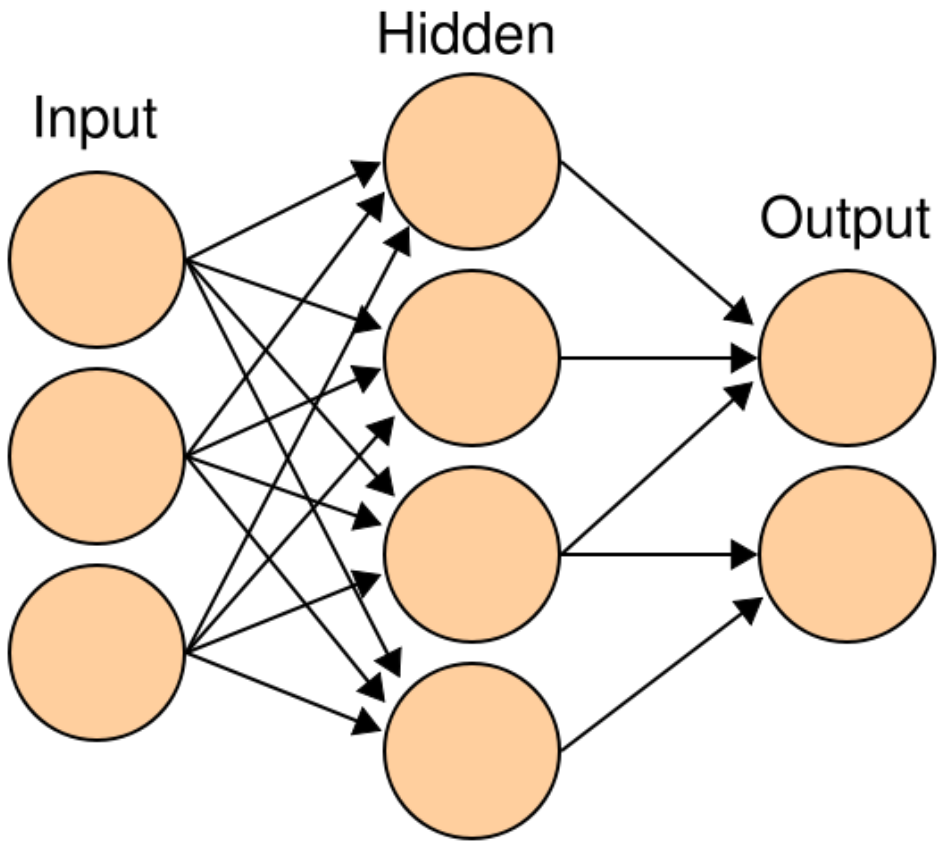
- 1943年，心理学家沃伦.马卡洛克(Warren McCulloch，1898-1969)和皮茨发表《神经活动中内在思想的逻辑演算》，讨论了理想化、极简化的人工神经网络，首次提到人工神经元网络；
- 1949年，心理学家唐纳德.赫布（Donald Hell，1904-1985）在《行为组织学》中提出了基于神经元构建学习模型的法则；
- 1957年，心理学家弗兰克.罗森布拉特（Frank Rosenblatt,1928-1971)第一个将Hebb学习理论用于模拟人类感知，提出“感知器”的概念模型；1962年《神经动力学原理：感知器和大脑机制的理论》；
- 1969年，马文.明斯基和西摩尔.派普特发表《感知器：计算几何学导论》(Perceptrons: An Introduction to Computational Geometry, MIT Press, 1969)；

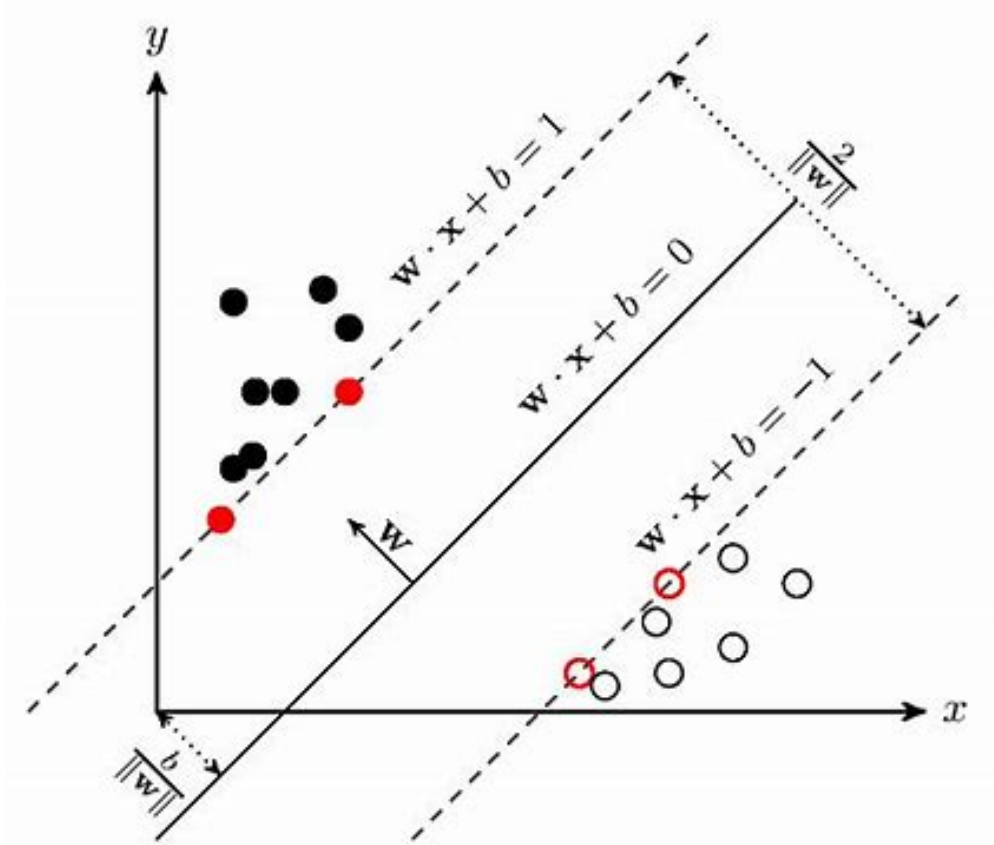
主要内容

1. 感知机模型
2. 感知机学习策略
    i. 数据集的线性可分性
    ii. 感知机学习策略
    iii. 感知机学习算法
3. 感知机学习算法
    i. 感知机学习算法的原始形式
    ii. 算法的收敛性
    iii. 感知机学习算法的对偶形式

摘要：

- 1957年由Rosenblatt提出，是**神经网络**与**支持向量机**的基础。
- 感知机是二类分类的**线性分类模型**。
- 损失函数$L(w,b)$的经验风险最小化。
- 感知机和SVM的更多联系源自margin的思想。


Input
Hidden
Output


y
x
w · x + b = 1
w · x + b = 0
w · x + b = −1
w
2/‖w‖
b/‖w‖

感知器模型

感知器定义

假设输入空间$\mathcal{X} \subseteq R^n$,
输出空间$\mathcal{Y} = \{+1, -1\}$。

- 输入 $x \in \mathcal{X}$ 表示实例的特征向量，对应于输入空间的点；
- 输出 $y \in \mathcal{Y}$ 表示实例的类别。

由输入空间到输出空间的函数

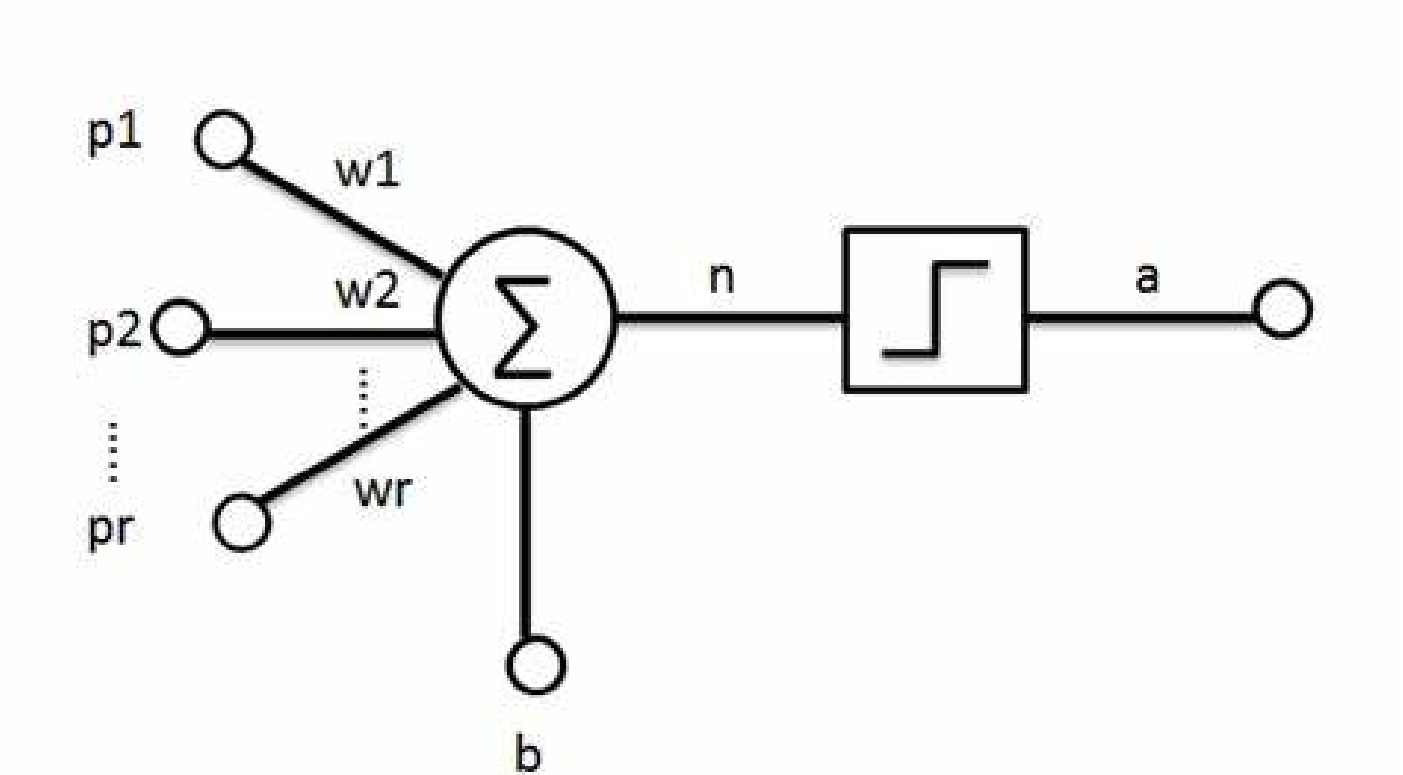
$$f(x) = sign(w \cdot x + b)$$

称为感知机。

其中，w和b为感知机模型参数，$w \in R^n$叫做权值或权值向量，$b \in R$叫偏置，$w \cdot x$表示w和x的内积。$sign$是符号函数，即

$$sign\left(x\right) = \begin{cases} +1, x \geq 0 \\ -1, x < 0 \end{cases}$$

感知器结构



线性方程

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

对应于特征空间R^n中的一个超平面S，其中w是超平面的法向量，b是超平面的截距。超平面S将特征空间划分为两部分，位于其中的点被分为正、负两类，超平面S称为分离超平面。

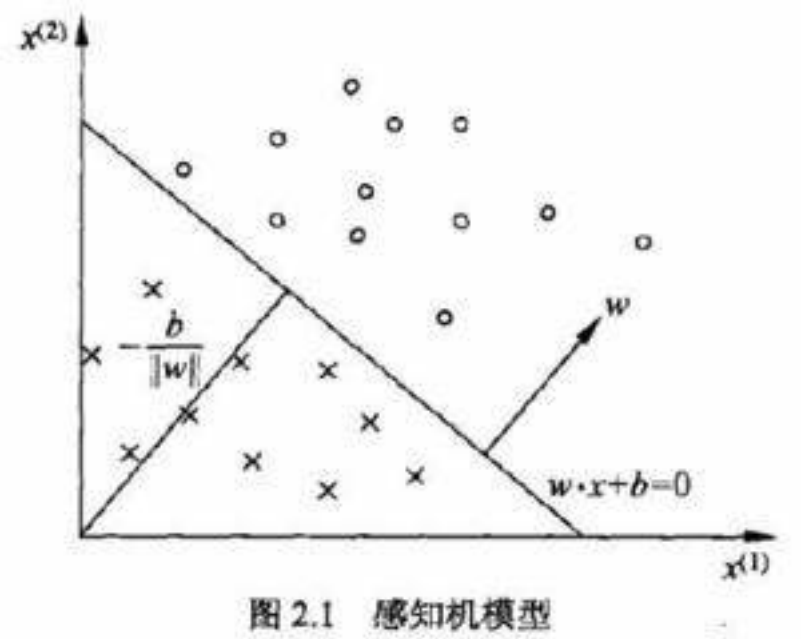


图 2.1 感知机模型

给定数据集

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

其中，$x_i\in\mathcal{X}=R^n, y_i\in\mathcal{Y}=\{+1,-1\}, i=1,2,\cdots,N,$

如果存在某个超平面S

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

能够将数据集的正实例和负实例完全正确地划分到超平面的两侧，即对所有$y_i=+1$的实例x_i，有$w\cdot x_i+b>0$，对所有$y_i=-1$的实例x_i，有$w\cdot x_i+b<0$，则称数据集T为线性可分数据集；否则，称数据集T线性不可分。

感知器学习策略

- 确定学习策略就是定义**经验**损失函数并将损失函数最小化。
- 注意这里提到了**经验**，所以学习是在**训练数据集**上的操作。

损失函数选择

> 损失函数的一个自然选择是误分类点的总数，但是，这样的损失函数**不是参数**w, b**的连续可导函数，不易优化**。
>
> 损失函数的另一个选择是误分类点到超平面S的总距离，这是感知机所采用的

输入空间R^n中的任一点x_0到超平面S的距离：

$$\frac{1}{\|w\|}|w \cdot x_0 + b|$$

其中$\|w\|$是w的L_2范数。

对于误分类数据(x_i, y_i)，当$w \cdot x + b > 0$时，$y_i = -1$，当$w \cdot x + b < 0$时，$y_i = +1$，有

$$-y_i (w \cdot x_i + b) > 0$$

误分类点x_i到分离超平面的距离：

$$-\frac{1}{\|w\|} y_i (w \cdot x_i + b)$$

假设超平面S的误分类点集合为M，则所有误分类点到超平面S的总距离：

$$-\frac{1}{\|w\|}\sum_{x_i\in M} y_i\left(w\cdot x_i+b\right)$$

感知机学习的经验风险函数(损失函数)

$$L(w,b)=-\sum_{x_i\in M} y_i(w\cdot x_i+b)$$

其中M是误分类点的集合

给定训练数据集T，损失函数$L(w,b)$是w和b的连续可导函数。

感知器学习算法

- 感知器学习算法是误分类驱动的，具体采用**随机梯度下降法(stochastic gradient descent)**

梯度下降法

梯度微积分含义

- 在微积分里面，对多元函数的参数求偏导数，把求得的各个参数的偏导数以向量的形式写出来，就是梯度。
- 梯度的定义：

$$gradf(x_0, x_2, ..., x_n) = (\frac{\partial f}{x_0}, ..., \frac{\partial f}{\partial x_j}, ..., \frac{\partial f}{\partial x_n})$$

- 函数在某一点的梯度是这样一个向量，它的方向与取得最大方向导数的方向一致，而它的模为方向导数的最大值。

  注意：

  - 梯度是一个向量，即有方向有大小；
  - 梯度的方向是最大方向导数的方向；
  - 梯度的值是最大方向导数的值。

梯度下降法可以描述为：

Repeat

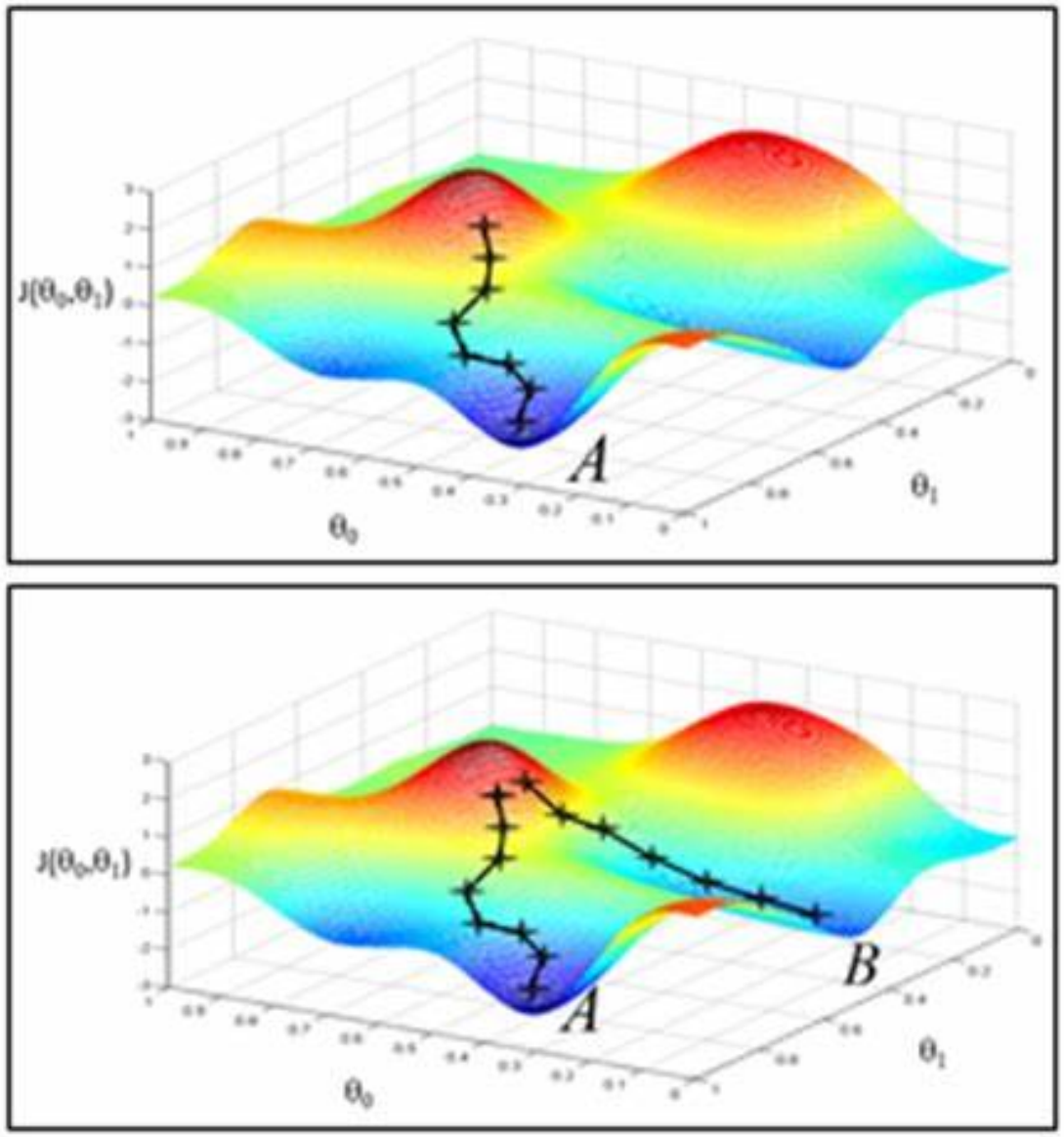
{

$$x_0 = x_0 - \alpha \frac{\partial f}{\partial x_0}$$

$$x_j = x_j - \alpha \frac{\partial f}{\partial x_j}$$

$$x_n = x_n - \alpha \frac{\partial f}{\partial x_n}$$

}



梯度的几何意义：

- 本意是一个向量（矢量），表示某一函数在该点处的方向导数沿着该方向取得最大值，几何意义上就是函数在该点处沿着该方向（此梯度的方向）变化最快，变化率最大（为该梯度的模）。

感知器学习算法的原始形式

任意选取一个超平面w_0, b_0，用梯度下降法不断极小化目标函数。

损失函数$L(w, b)$的梯度为：

$$\nabla_w L(w, b) = -\sum_{x_i \in M} y_i x_i$$

$$\nabla_b L(w, b) = -\sum_{x_i \in M} y_i$$

算法2.1(感知器学习原始算法)

输入

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

$$x_i \in \mathcal{X}=\mathbf{R^n}, \mathbf{y_i} \in \mathcal{Y}=\{-1,+1\}, i=1,2,\ldots,N;\quad 0<\eta \leqslant 1$$

输出 $w,b;f(x)=sign(w\cdot x+b)$

1. 选取初值w_0,b_0
2. 训练集中选取数据(x_i,y_i)
3. 如果$y_i(w\cdot x_i+b)\leqslant 0$

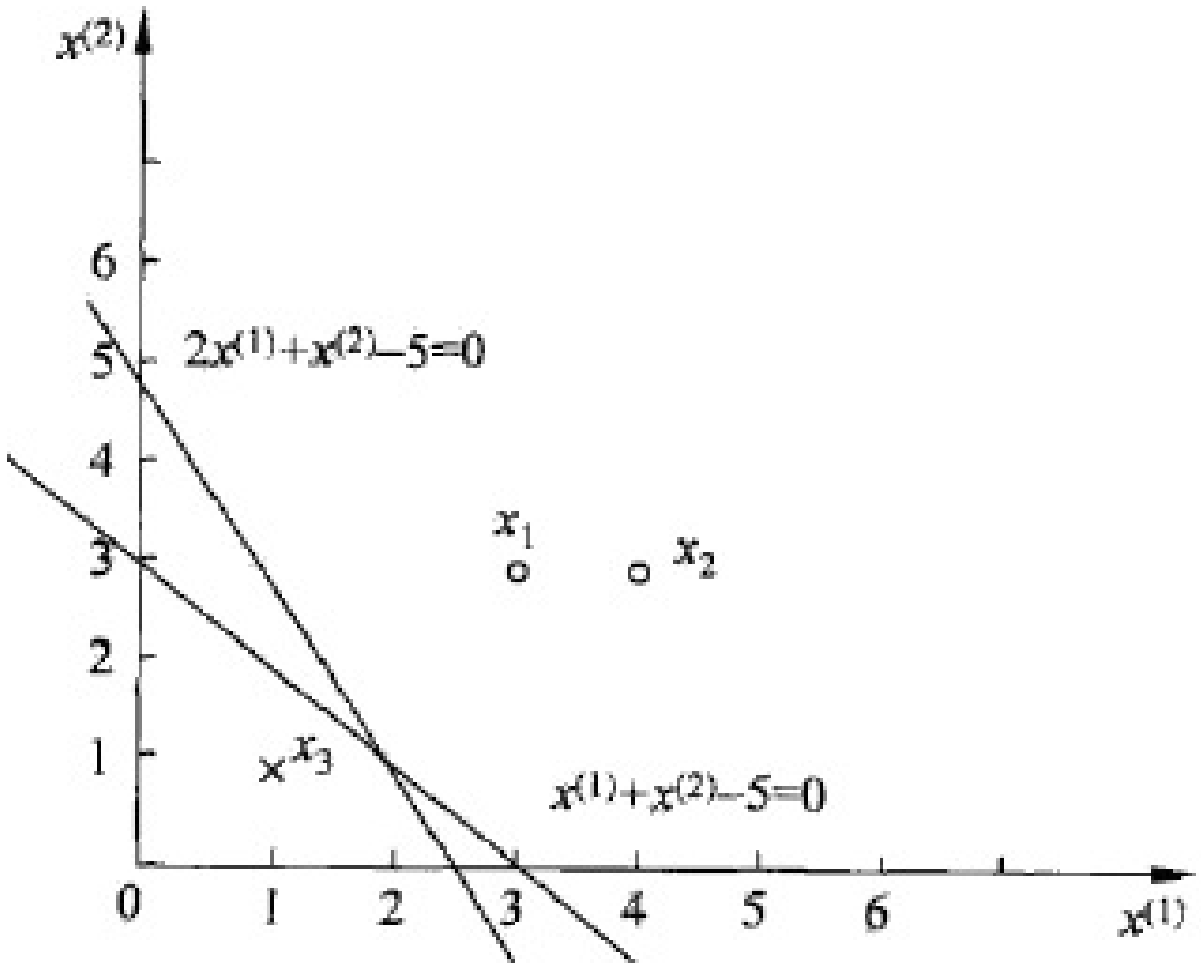
$$w \leftarrow w+\eta y_i x_i \quad b \leftarrow b+\eta y_i$$

4. 转至(2)，直至训练集中没有误分类点

例：

正例：$x_1 = (3,3)^T, x_2 = (4,3)^T,$

负例：$x_3 = (1,1)^T$



解： 1.构建优化问题：$\min_{w,b} L(w,b) = -\sum_{x_i \in M} y_i(w \cdot x + b)$
2.求解$w, b, \eta = 1$

- (1)取初值$w_0 = 0, b_0 = 0$
- (2)对$x_1 = (3,3)^T, y_1 = (w_0 \cdot x_1 + b_0)$未能被正确分类，更新$w, b$;
  - $w_1 = w_0 + y_1 x_1 = (3,3)^T, b_1 = b_0 + y_1 = 1$
  - 得到线形模型：$w_1 \cdot x + b_1 = 3x^{(1)} + 3x^{(2)} + 1$
- (3)x_2,显然,$y_i(w_i \cdot x_i + b_i) > 0$,被正确分类，对$x_3 = (1,1)^T, y_3(w_1 \cdot x_3 + b_1) < 0$,被误分类，$w_2 = w_1 + y_3 x_3 = (2,2)^T, b_2 = b_1 + y_3 = 0$
- 得到线形模型$w_2 \cdot x + b_2 = 2x^{(1)} + 2x^{(2)}$
- (4)如此继续下去：
- (5)直到得到超平面$x^{(1)} + x^{(2)} - 3 = 0$
- (6)得到感知器模型$f(x) = sign(x^{(1)} + x^{(2)} - 3)$

感知器学习算法收敛性

定理2.1（Novikoff)

算法的收敛性：证明经过有限次迭代可以得到一个将训练数据集完全正确划分的分离超平面及感知机模型。

定理表明：

- 误分类的次数k是有上界的，当训练数据集线性可分时，感知机学习算法原始形式迭代是收敛的。
- 感知机算法存在许多解，既依赖于初值，也依赖迭代过程中误分类点的选择顺序。
- 为得到唯一分离超平面，需要增加约束，如SVM。
- 线性不可分数据集，迭代震荡。

对偶形式

对偶形式的基本思想是将w和b表示为实例x_i和标记y_i的线性组合的形式，通过求解其系数而求得w和b。

可以假设初始值w_0, b_0均为0,对误分类点(x_i, y_i)通过

$$w \leftarrow w + \eta y_i x_i$$
$$b \leftarrow b + \eta y_i$$

逐步修改w, b，设修改n次，则w, b关于(x_i, y_i)的增量分别是$\alpha_i y_i x_i$和$\alpha_i y_i$，这里$\alpha_i = n_i \eta$，最后得到的w, b可以分别表示为

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

感知器学习算法的对偶形式

> 输入：$T=\{(x_1,y_1),(x_2,y_2),\cdots,(x_N,y_N)\}$
> $x_i \in \mathcal{X}=\mathbf{R^n},\mathbf{y_i} \in \mathcal{Y}=\{\mathbf{-1},\mathbf{+1}\},\mathbf{i=1,2,\ldots,N};\mathbf{0}<\eta \leqslant \mathbf{1}$
>
> 输出：

$$\alpha,b;f(x)=sign\left(\sum_{j=1}^{N}\alpha_j y_j x_j \cdot x+b\right)$$

$$\alpha=(\alpha_1,\alpha_2,\cdots,\alpha_N)^T$$

算法

> 1. $\alpha \leftarrow 0, b \leftarrow 0$
> 2. 训练集中选取数据(x_i, y_i)
> 3. 如果$y_i \left(\sum_{j=1}^{N} \alpha_j y_j x_j \cdot x_i + b \right) \leqslant 0$
>
> $$\alpha_i \leftarrow \alpha_i + \eta$$
> $$b \leftarrow b + \eta y_i$$
>
> 4. 转至(2)，直至训练集中没有误分类点

Gram matrix

对偶形式中，训练实例仅以内积的形式出现。
为了方便可预先将训练集中的实例间的内积计算出来并以矩阵的形式存储，这个矩阵就是所谓的Gram矩阵

$$G = [x_i \cdot x_j]_{N \times N}$$

例： 正例：$x_1 = (3,3)^T, x_2 = (4,3)^T$,负例：$x_3 = (1,1)^T$

解：

- (1) 取$\alpha_i = 0, i = 1,2,3, b = 0, \eta = 1$
- (2) 计算Gram矩阵

$$\begin{bmatrix} 18 & 21 & 6 \\ 21 & 25 & 7 \\ 6 & 7 & 2 \end{bmatrix}$$

- (3) 误分条件

$$y_i(\sum_{j=1}^{N}\alpha_j y_j x_j \cdot x_i + b) \leq 0$$

参数更新

$$\alpha_i \leftarrow \alpha_i + 1, b \leftarrow b + y_j$$

- (4) 迭代
- (5) $w = 2x_1 + 0x_2 - 5x_3 = (1,1)^T, b = -3$
- (6) 得到感知器模型$f(x) = sign(x^{(1)} + x^{(2)} - 3)$

参考

1.XOR [^1]: XOR

Enjoy your machine learning!

https://github.com/wjssx/

E-mail: csr_dsp@sina.com