神经网络
Neural Network

深度学习三次热潮

第一次热潮(20世纪50-60年代)

- 1950年图灵提出了图灵测试。
- 1958年计算机科学家Rosenblatt提出了一种具有三层网络特性的神经网络结构，成为“感知器”。
- 1969年，人工智能的先驱Minsky出版了一本名为《感知器》的书，书中指出简单的神经网络只能运用于线性问题的求解，能够求解非线性问题的网络应具有隐层，而从理论上还不能证明将感知器扩展到多层网络是有意义的。

第二次热潮(20世纪80-90年代)

- 语音识别是当时最具代表性的突破成果之一，语音识别领域最具代表性的人物就是李开复了。
- 1986年，Rumelhart，Hinton，Williams发展了BP算法。（多层感知器的误差反向传播算法）。
- 1987年6月，首届国际神经网络学术会议在美国加州圣地亚哥召开，到会代表有1600余人。之后国际神经网络学会和国际电气工程师与电子工程师学会（IEEE）联合召开每年一次的国际学术会议。

第三次热潮(2006年至今)

- 2006年Hinton在《Science》杂志上的发表了一篇深度学习的论文。
- 2009年李飞飞创立ImageNet。
- 2016年AlphaGo战胜人类顶级围棋选手。

深度学习爆发三要素

ImageNet Dataset

IMAGENET

Russakovsky, O., Deng, J., Su, H., Krause, J., Satheesh, S., Ma, S., ... & Fei-Fei, L. (2015). Imagenet large scale visual recognition challenge. *arXiv preprint arXiv:1409.0575*. [web]

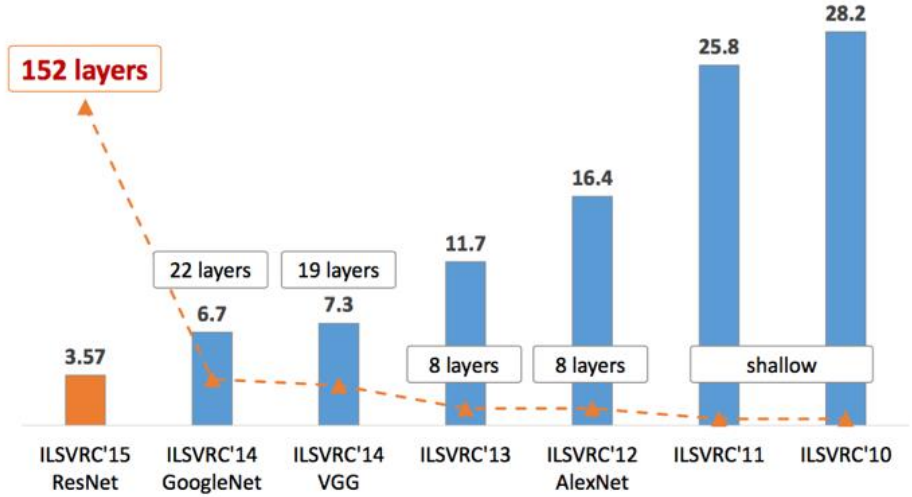
ILSVRC

ImageNet Challenge

IMAGENET

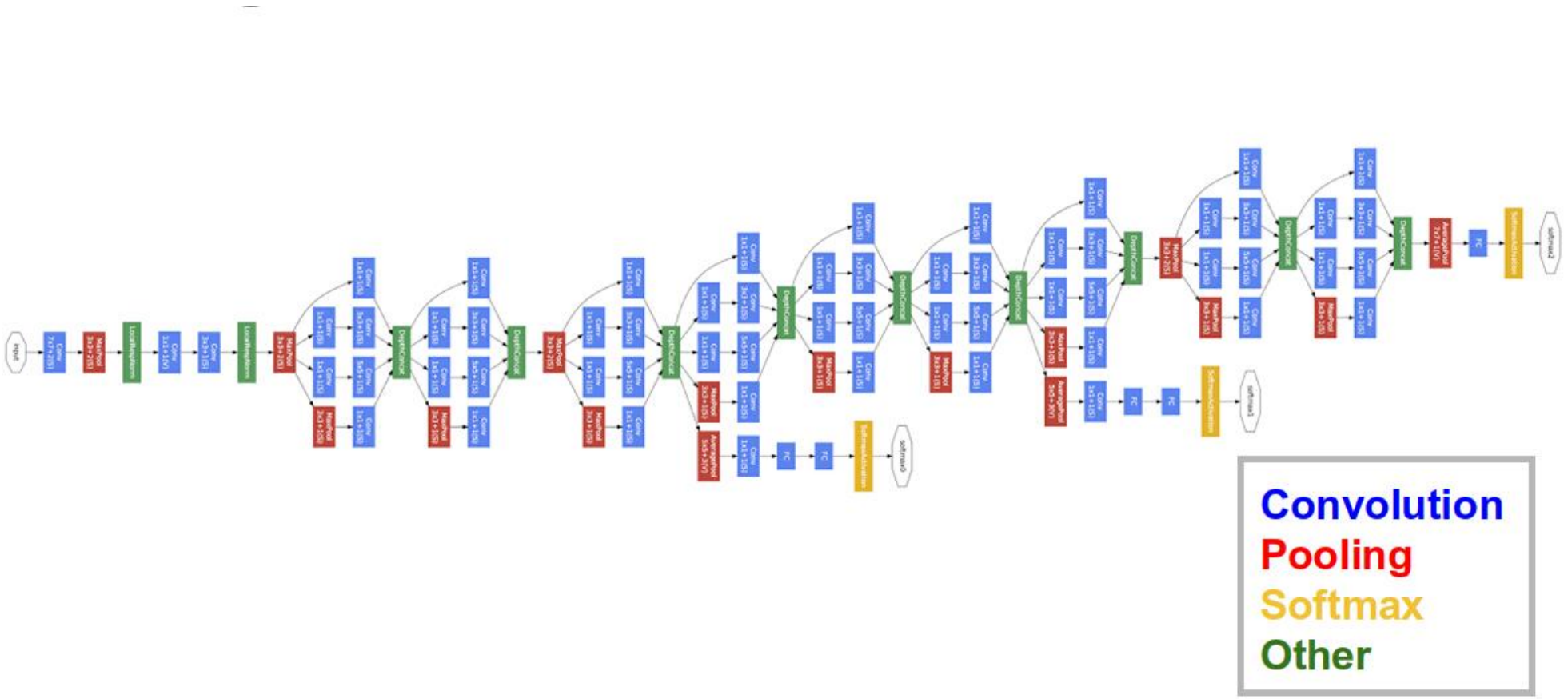
- 1,000 object classes (categories).
- Images:
  - 1.2 M train
  - 100k test.

4



计算能力

算法



深度学习三巨头

Geoffrey Hinton

- 英国出生的计算机学家和心理学家，以其在神经网络方面的贡献闻名。Hinton是反向传播算法的发明人之一，也是深度学习的积极推动者。目前担任多伦多大学计算机科学系教授。
- 2013年3月加入Google，领导Google Brain项目。

Yann LeCun

- 计算机科学家，他最著名的工作是光学字符识别和计算机视觉上使用卷积神经网络（CNN），他也被称为卷积网络之父。
- 多伦多大学跟随Hinton做博士后。1988年，加入贝尔实验室，之后研发出了卷积神经网络，曾广泛用于手写数字识别。
- 2003年去了纽约大学任教。2013年12月加入了Facebook，成为Facebook人工智能实验室的第一任主任。

Yoshua Bengio

- 在MIT和贝尔实验室做过博士后研究员，自1993年之后就在蒙特利尔大学任教。在预训练和自动编码器等方面作出过重大贡献。

- “我留在学术圈为全人类作贡献，而不是为某一个公司赚钱”

Andrew Wu（吴恩达）

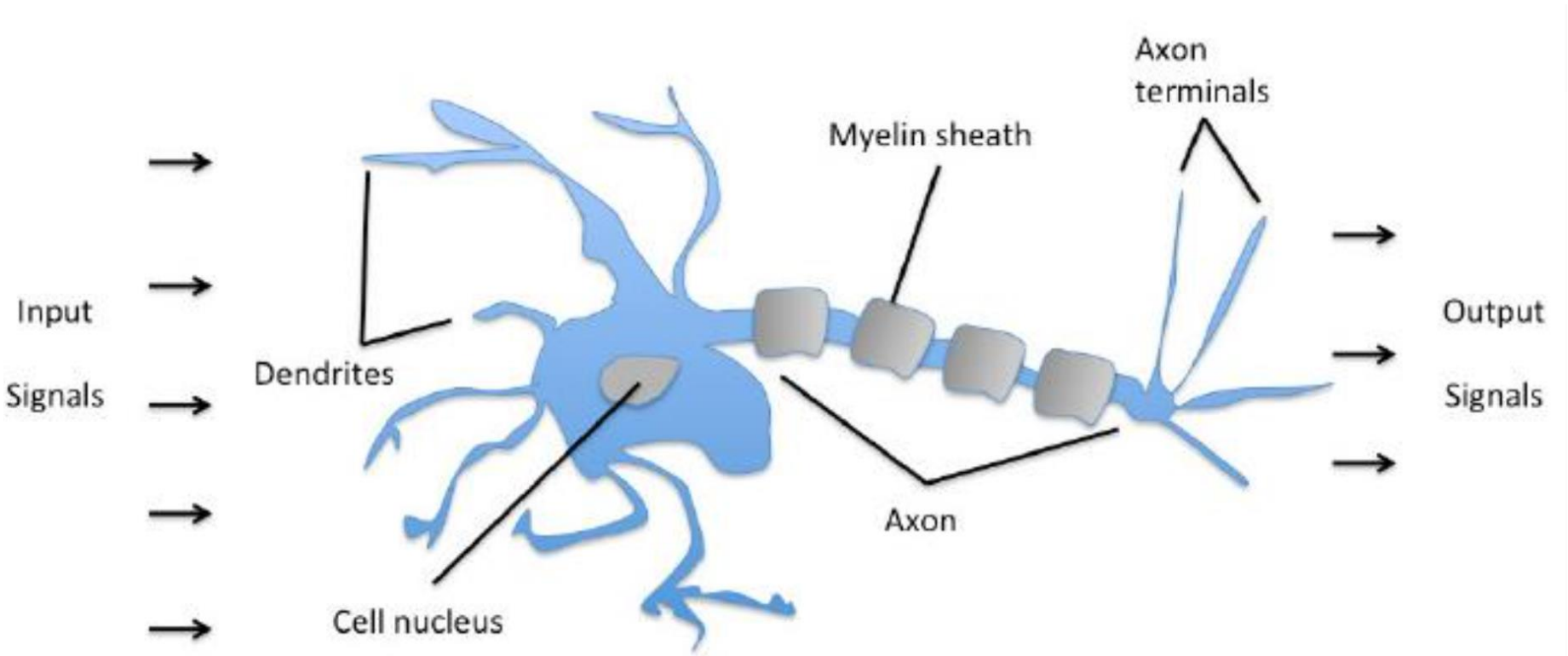
- 曾经是斯坦福大学计算机科学系和电气工程系的副教授，斯坦福人工智能实验室主任。他创建了在线教育平台Coursera。
- 2011年，吴恩达在Google创建了Google Brain项目，通过分布式集群计算机开发超大规模的人工神经网络。
- 2014年5月，吴恩达加入百度，负责百度大脑计划，并担任百度公司首席科学家。
- 2017年3月，吴恩达从百度离职。

关系图

LeCun Hinton Bengio Andrew Ng

单层感知器

人体神经网络



Schematic of a biological neuron.

单层感知器

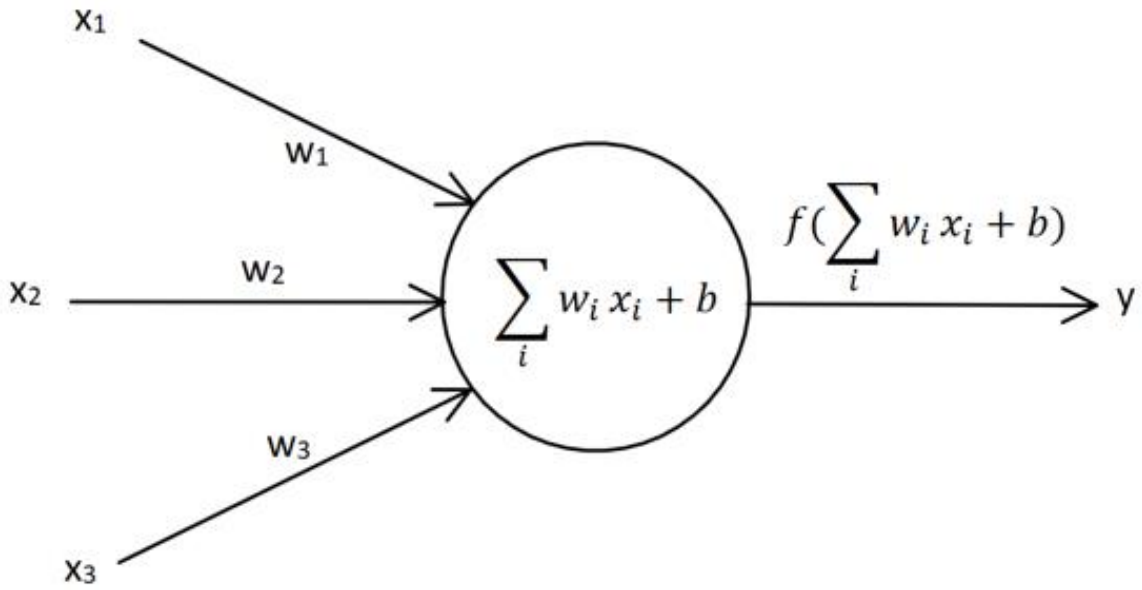
输入节点：x_1,x_2,x_3

输出节点：y

权向量：w_1,w_2,w_3

偏置因子：b

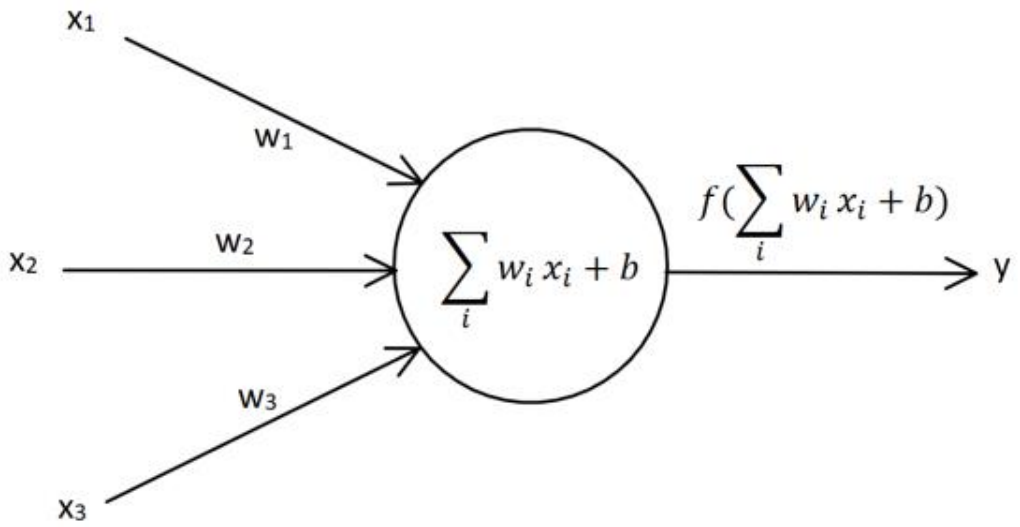
激活函数：$sign(x)=\begin{matrix}1 & X>=0\\ -1 & X<0\end{matrix}$



单层感知器举例

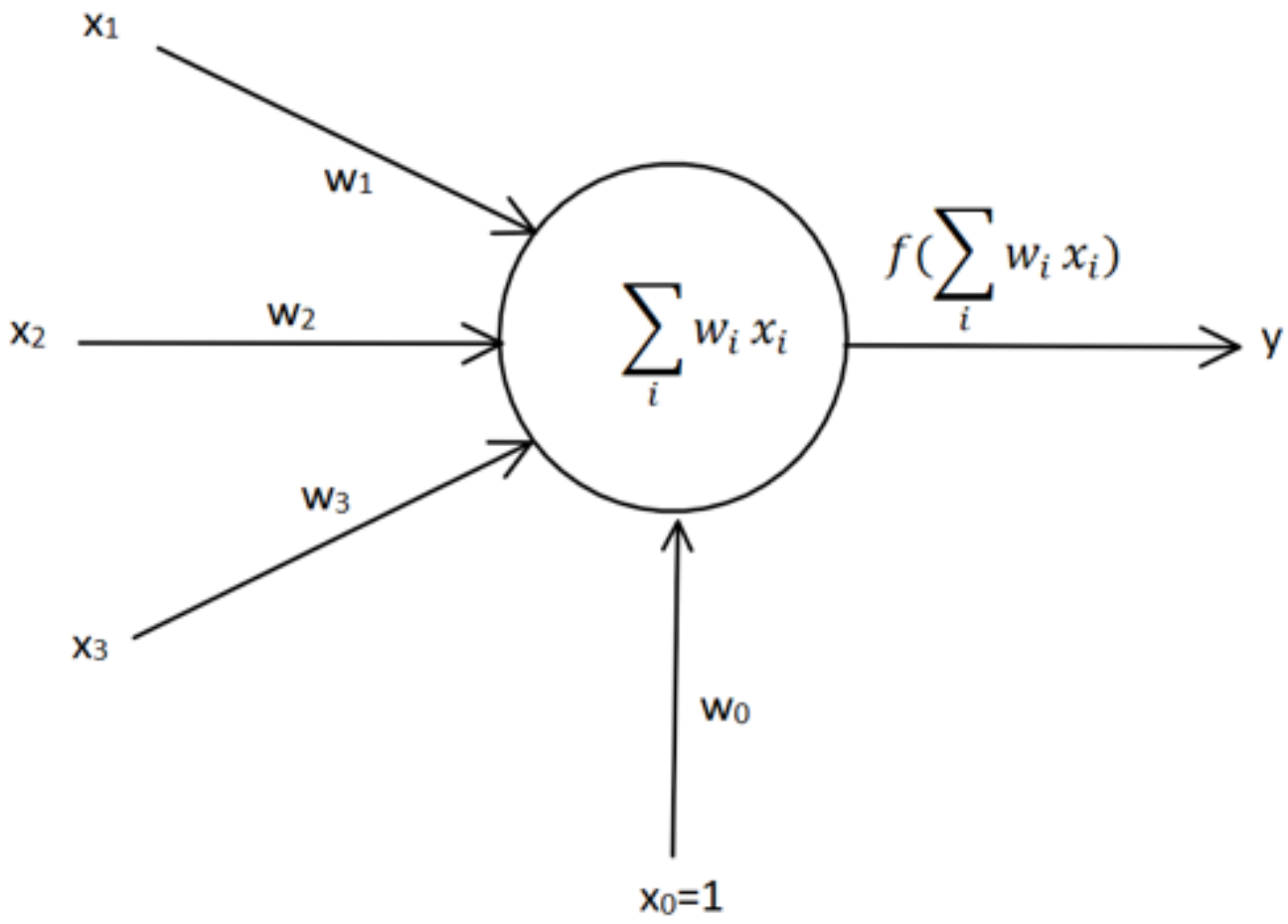
b=-0.6

x_1	x_2	x_3	Y
0	0	0	-1
0	0	1	-1
0	1	0	-1
0	1	1	1
1	0	0	-1
1	0	1	1
1	1	0	1
1	1	1	1



$$y = \begin{cases} 1 & (0.5x_1+0.5x_2+0.5x_3-0.6>=0) \\ -1 & (0.5x_1+0.5x_2+0.5x_3-0.6<0) \end{cases}$$

单层感知器



感知器学习规则

$$y = f\left(\sum_{i=1}^{n} x_i \cdot w_i\right)$$

i = 0,1,2...
y是网络输出
f是sign函数

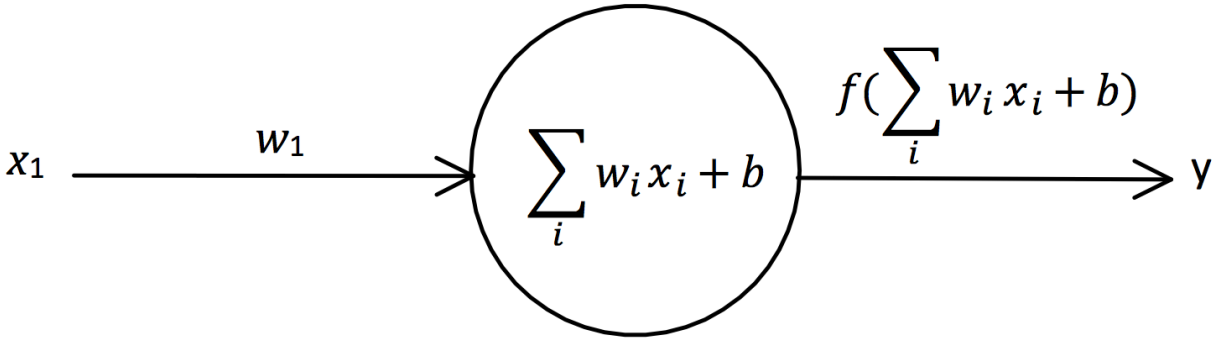
$$\Delta w_i = \eta(t - y)x_i$$

η表示学习率
t表示正确的标签
t和y的取值为±1

$$\Delta w_i = \pm 2\eta x_i$$

$$w_i = w_i + \Delta w_i$$

感知器学习规则



$$\Delta w_i = \eta(t - y)x_i$$

假设：t=1，η=1，x1=1，w1=-5，b=0:

Step1:
y = sign(1*(-5)) = -1
Δw = 1*(1-(-1))*1 = 2
w1 = w1 + Δw = -3

Step2:
y = sign(1*(-3)) = -1
Δw = 1*(1-(-1))*1 = 2
w1 = w1 + Δw = -1

Step3:
y = sign(1*(-1)) = -1
Δw = 1*(1-(-1))*1 = 2
w1 = w1 + Δw = 1
y = sign(1*1) = 1 = t

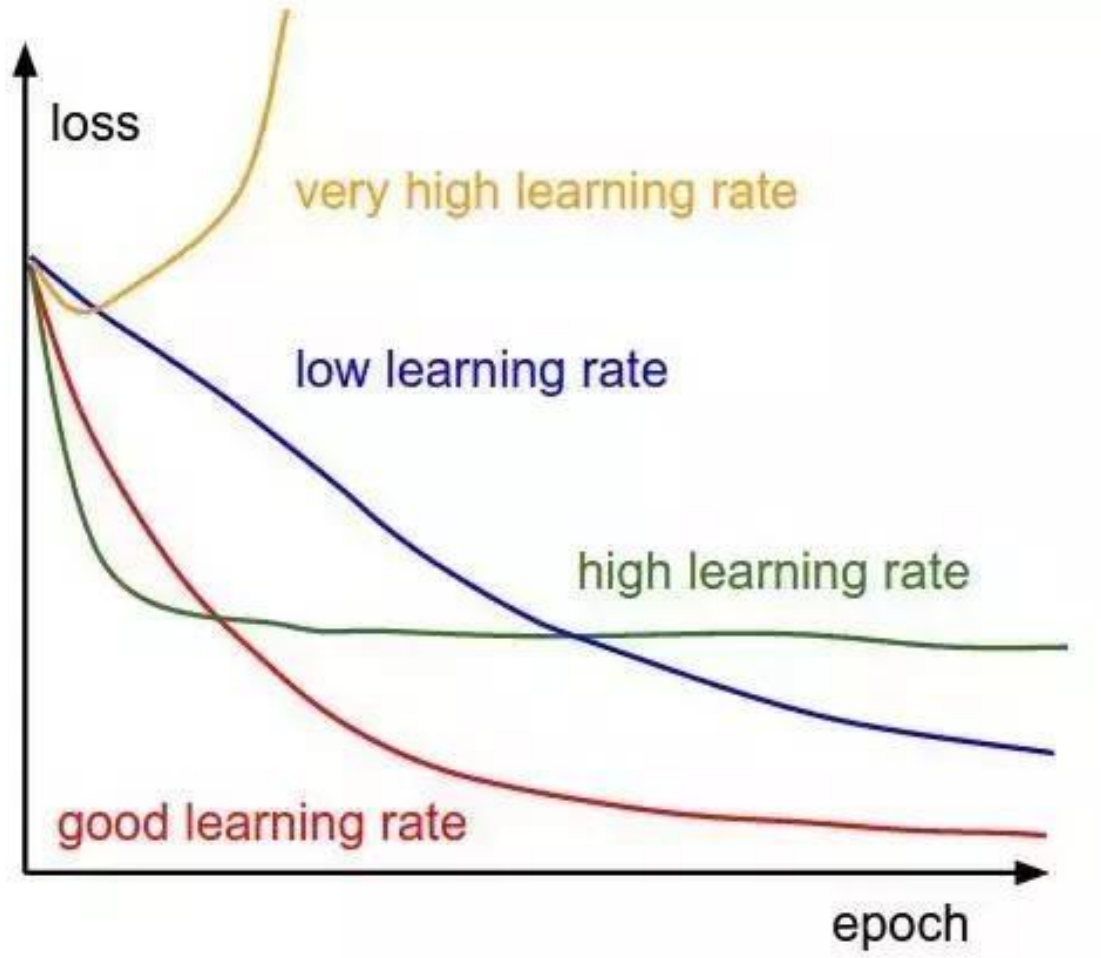
学习率

η取值一般取0-1之间

学习率太大容易造成权值调整不稳定

学习率太小，权值调整太慢，迭代次数太多

不同学习率


loss
very high learning rate
low learning rate
high learning rate
good learning rate
epoch

模型收敛条件

误差小于某个预先设定的较小的值

两次迭代之间的权值变化已经很小

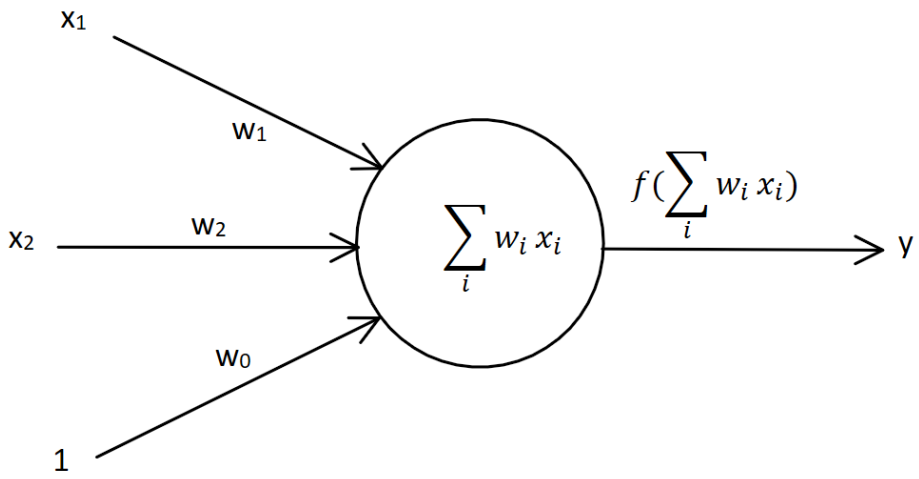
设定最大迭代次数，当迭代超过最大次数就停止

单层感知器程序

题目：假设平面坐标系上有四个点，(3,3),(4,3)这两个点的标签为1，(1,1),(0,2)这两个点的标签为-1。构建神经网络来分类。

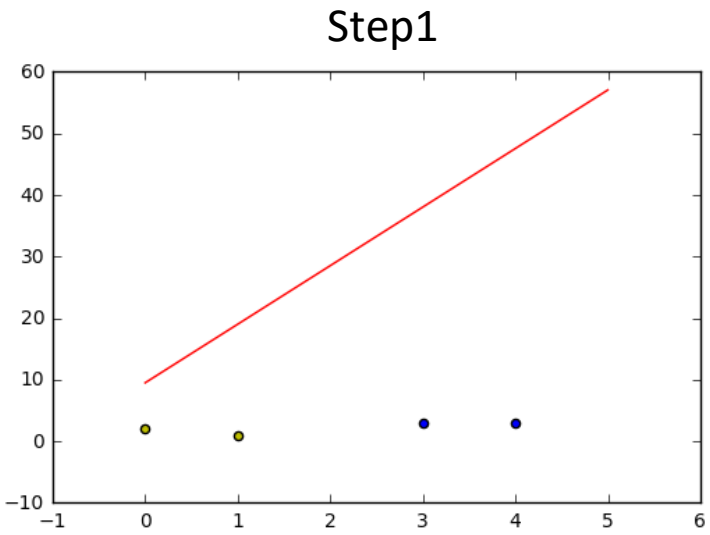
思路：我们要分类的数据是2维数据，所以只需要2个输入节点，我们可以把神经元的偏置值也设置成一个节点，这样我们需要3个输入节点。

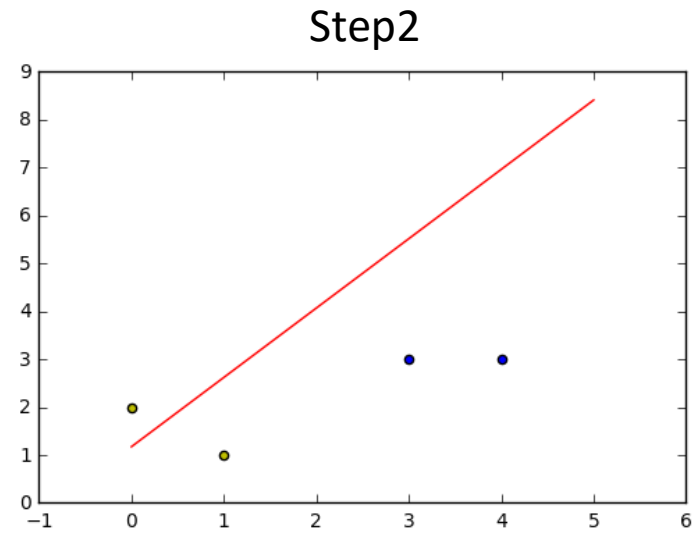
输入数据有4个(1,3,3),(1,4,3),
(1,1,1),(1,0,2)
数据对应的标签为(1,1,-1,-1)
初始化权值w_0,w_1,w_2取-1到1的随机数
学习率(learning rate)设置为0.11
激活函数为sign函数

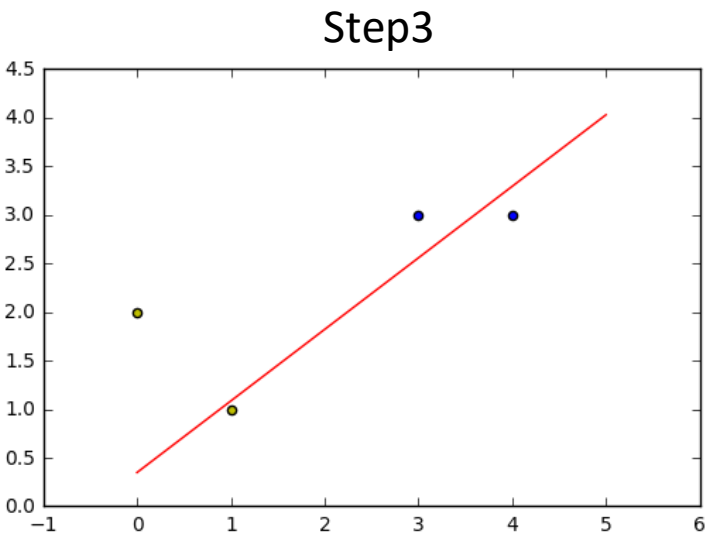


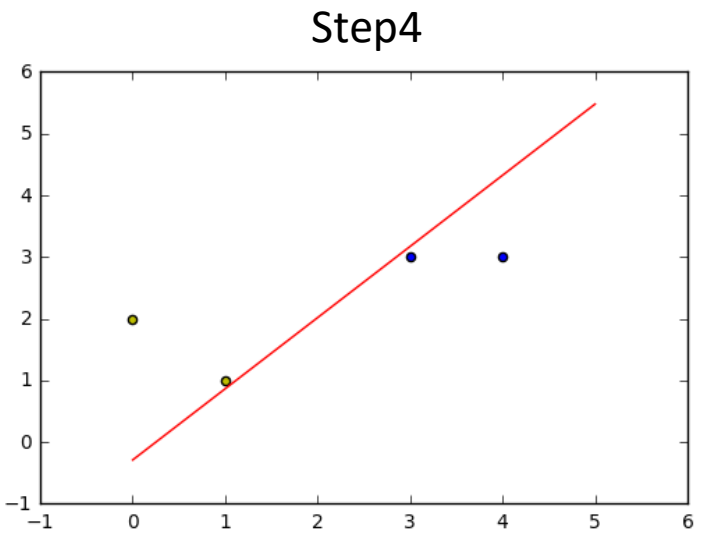
单层感知器

单层感知器分类


Step1
60
50
40
30
20
10
0
−10
−1
0
1
2
3
4
5
6


Step2
9
8
7
6
5
4
3
2
1
0
−1
0
1
2
3
4
5
6


Step3
4.5
4.0
3.5
3.0
2.5
2.0
1.5
1.0
0.5
0.0
−1
0
1
2
3
4
5
6


Step4
6
5
4
3
2
1
0
−1
−1
0
1
2
3
4
5
6

单层感知器异或问题

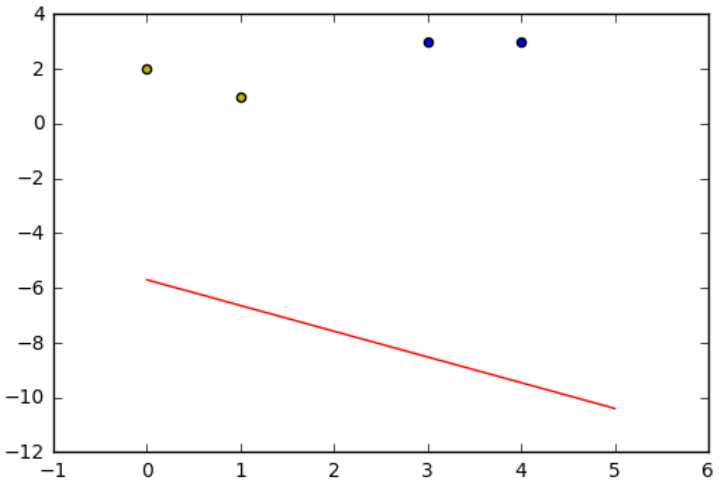
线性神经网络

线性神经网络在结构上与感知器非常相似，只是激活函数不同。
在模型训练时把原来的sign函数改成了purelin函数：y = x

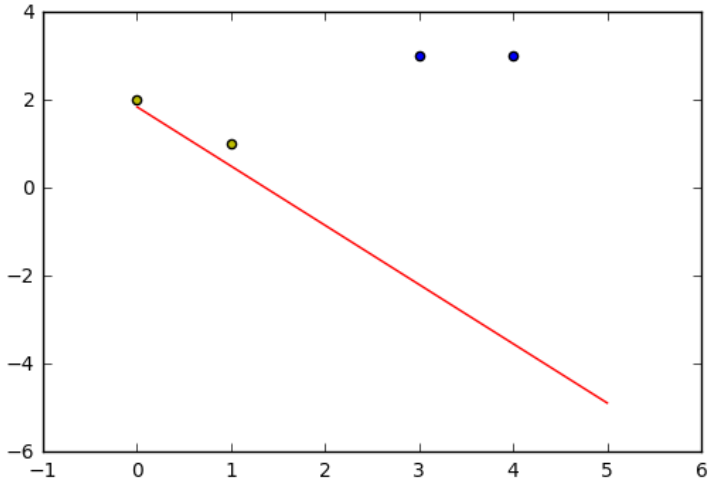
线性神经网络

线性神经网络分类

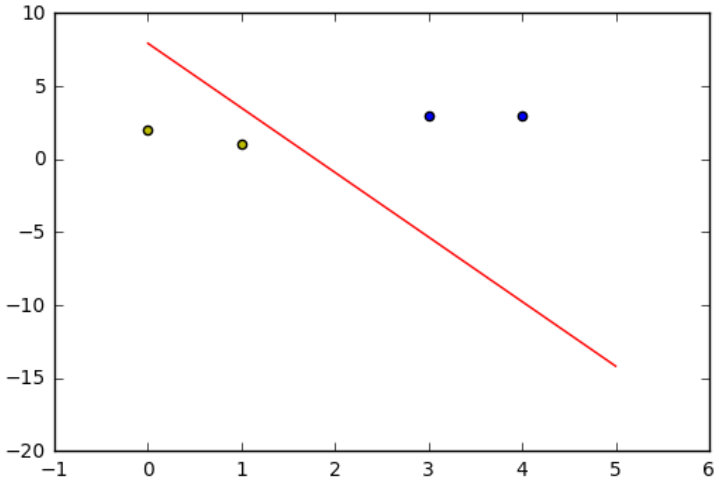
Step1



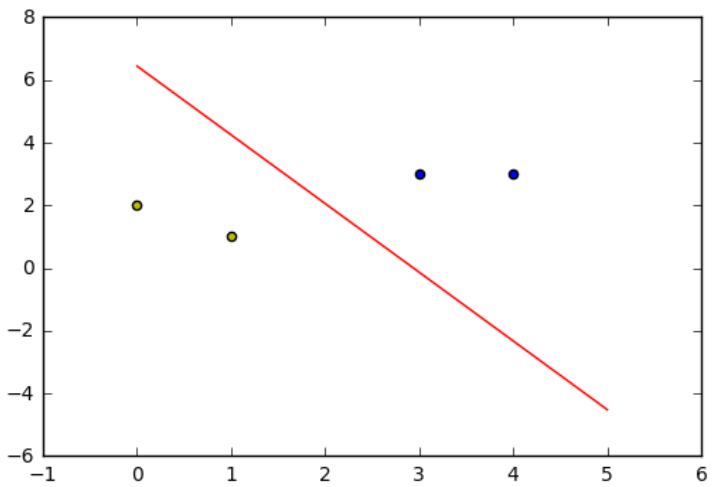
Step2



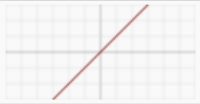
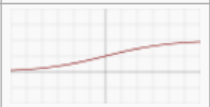
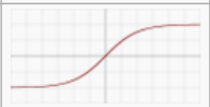
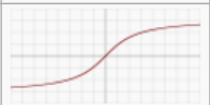
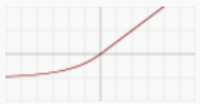
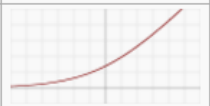
Step3



Step4



激活函数

Name	Plot	Equation	Derivative
Identity		$f(x) = x$	$f'(x) = 1$
Binary step		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x \neq 0 \\ ? & \text{for } x = 0 \end{cases}$
Logistic (a.k.a Soft step)		$f(x) = \frac{1}{1+e^{-x}}$	$f'(x) = f(x)(1-f(x))$
TanH		$f(x) = \tanh(x) = \frac{2}{1+e^{-2x}} - 1$	$f'(x) = 1 - f(x)^2$
ArcTan		$f(x) = \tan^{-1}(x)$	$f'(x) = \frac{1}{x^2+1}$
Rectified Linear Unit (ReLU)		$f(x) = \begin{cases} 0 & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} 0 & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Parameteric Rectified Linear Unit (PReLU)[2]		$f(x) = \begin{cases} \alpha x & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
Exponential Linear Unit (ELU)[3]		$f(x) = \begin{cases} \alpha(e^x - 1) & \text{for } x < 0 \\ x & \text{for } x \geq 0 \end{cases}$	$f'(x) = \begin{cases} f(x) + \alpha & \text{for } x < 0 \\ 1 & \text{for } x \geq 0 \end{cases}$
SoftPlus		$f(x) = \log_e(1+e^x)$	$f'(x) = \frac{1}{1+e^{-x}}$

Delta学习规则

1986年，认知心理学家McClelland和Rumelhart在神经网络训练中引入了δ规则，该规则也可以称为连续感知器学习规则。

δ学习规则是一种利用梯度下降法的一般性的学习规则。

Delta学习规则

代价函数(损失函数)(Cost Function,Lost Function)

二次代价函数：

$$E = \frac{1}{2}(t-y)^2 = \frac{1}{2}[t - f(WX)]^2$$

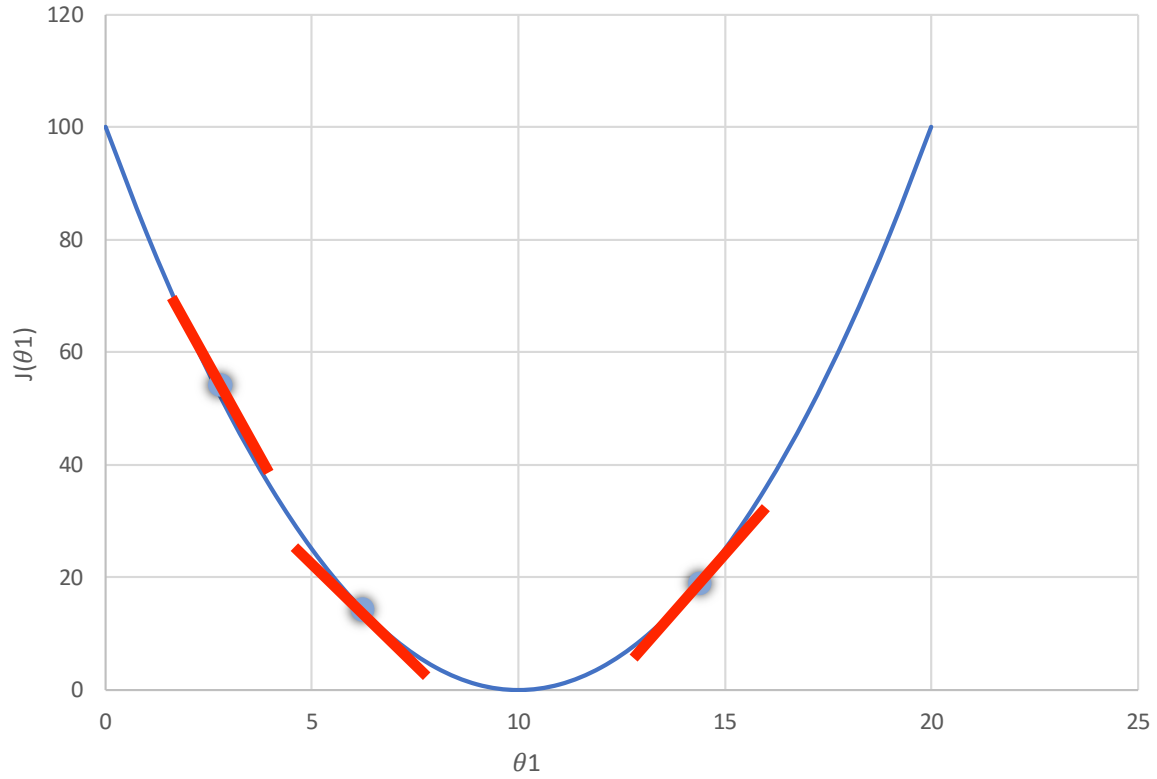
误差E是权向量W的函数，我们可以使用梯度下降法来最小化E的值：

$$\Delta W = -\eta E' = \eta X^T (t-y) f'(WX) = \eta X^T \delta$$

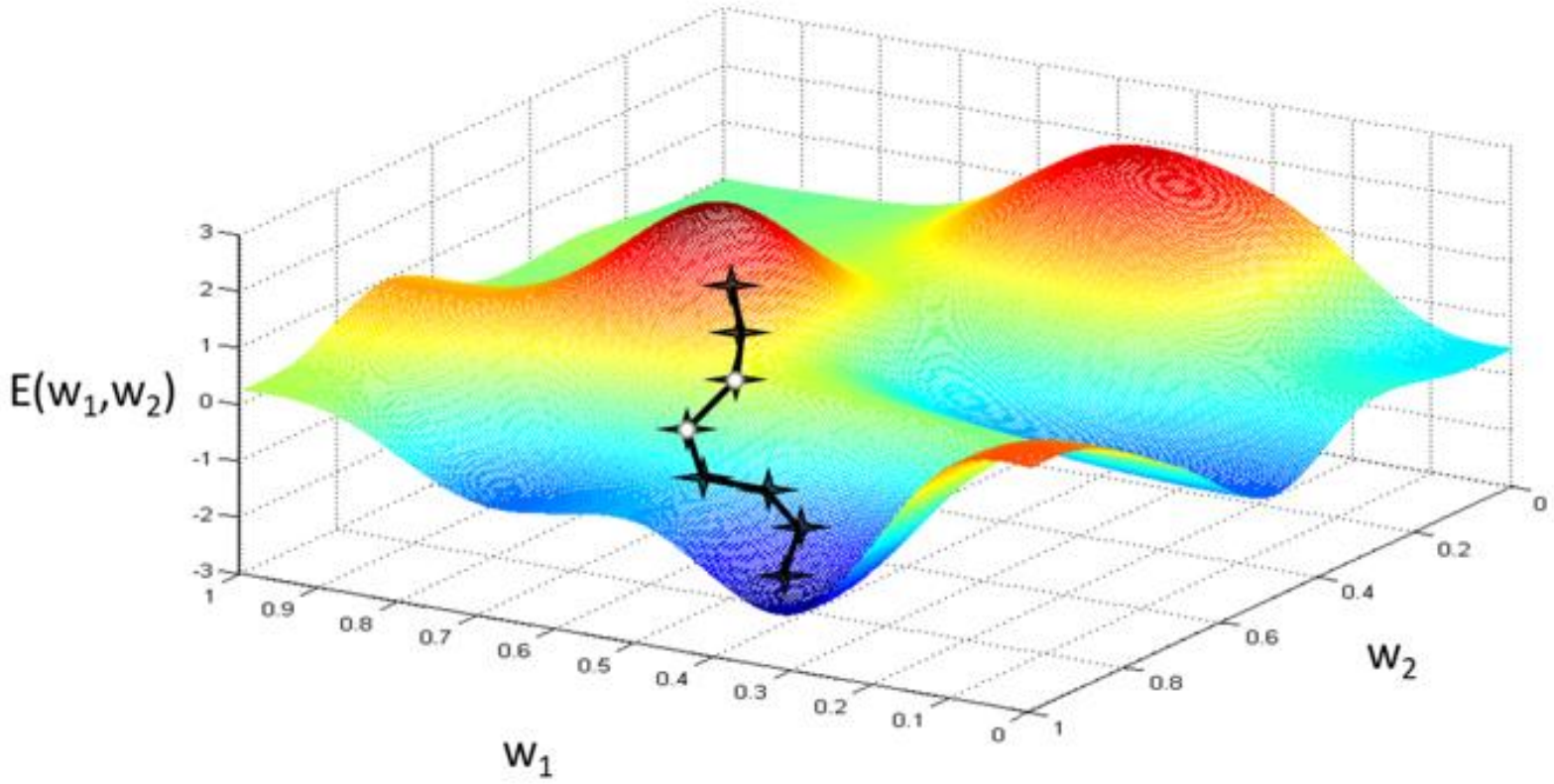
$$\Delta w_i = -\eta E' = \eta x_i (t-y) f'(WX) = \eta x_i \delta$$

梯度下降法-一维情况

$$w = w + \Delta w$$

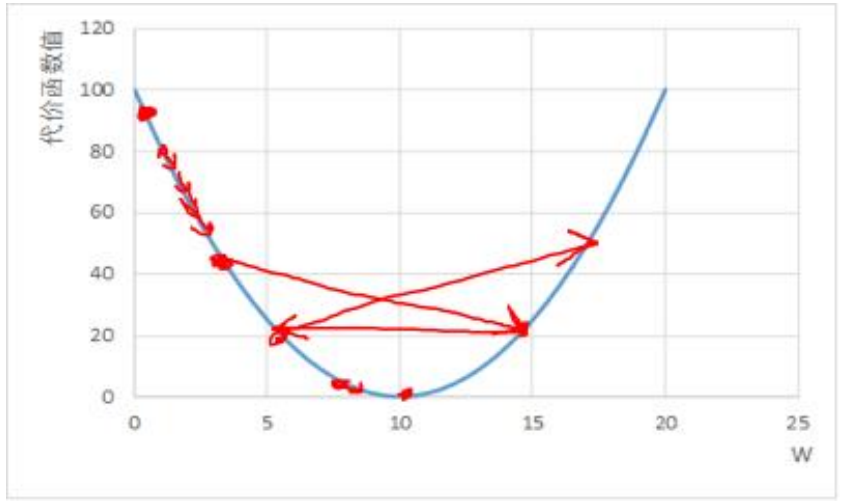


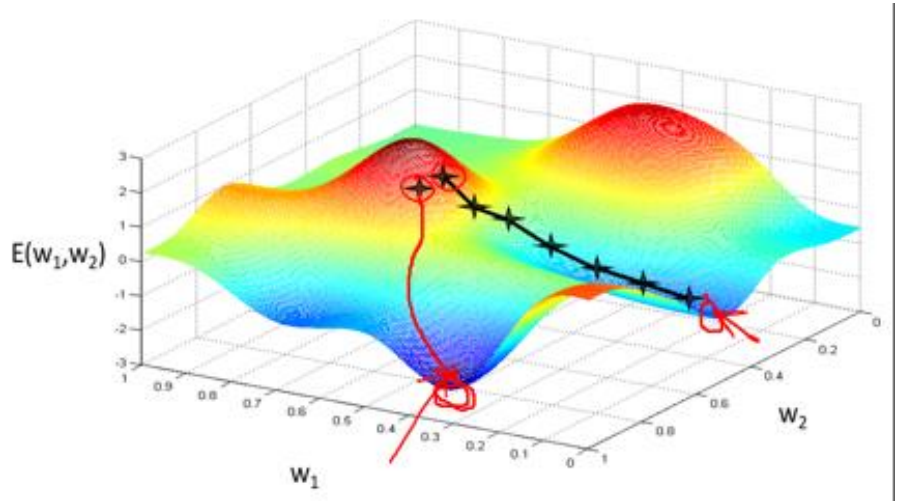
梯度下降法-二维情况



梯度下降法的问题

1.学习率难以选取，太大会产生震荡，太小收敛缓慢
2.容易陷入局部最优解(局部极小值)





解决异或问题

Madaline 可以用一种间接的方式解决线性不可分的问题，方法是用多个线性函数对区域进行划分，然后对各个神经元的输出做逻辑运算。如图 5-3 所示，Madaline 用两条直线实现了异或逻辑。

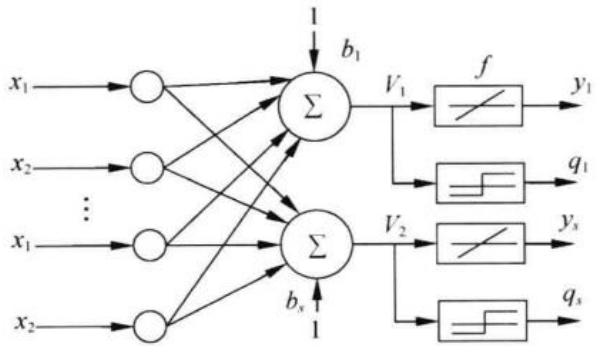


图 5-2　Madaline 结构图

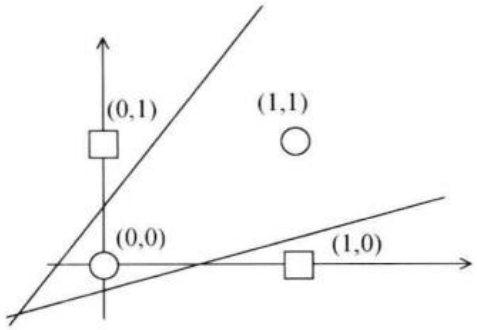


图 5-3　Madaline 实现异或

线性神经网络解决线性不可分问题的另一个方法是，对神经元添加非线性输入，从而引入非线性成分，这样做会使等效的输入维度变大，如图 5-4 所示。

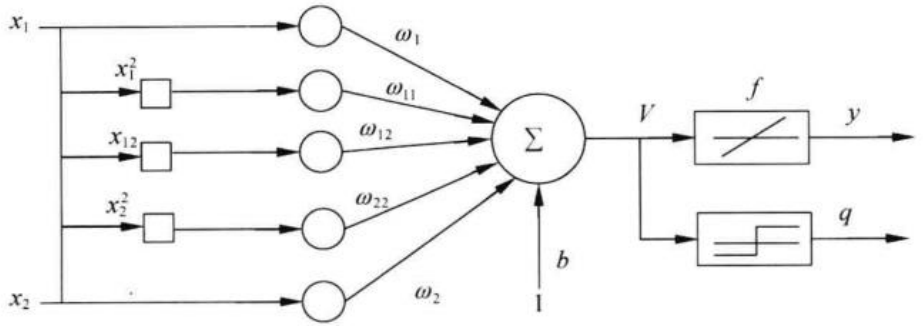


图 5-4　线性网络解决非线性问题

线性神经网络-异或问题

BP神经网络

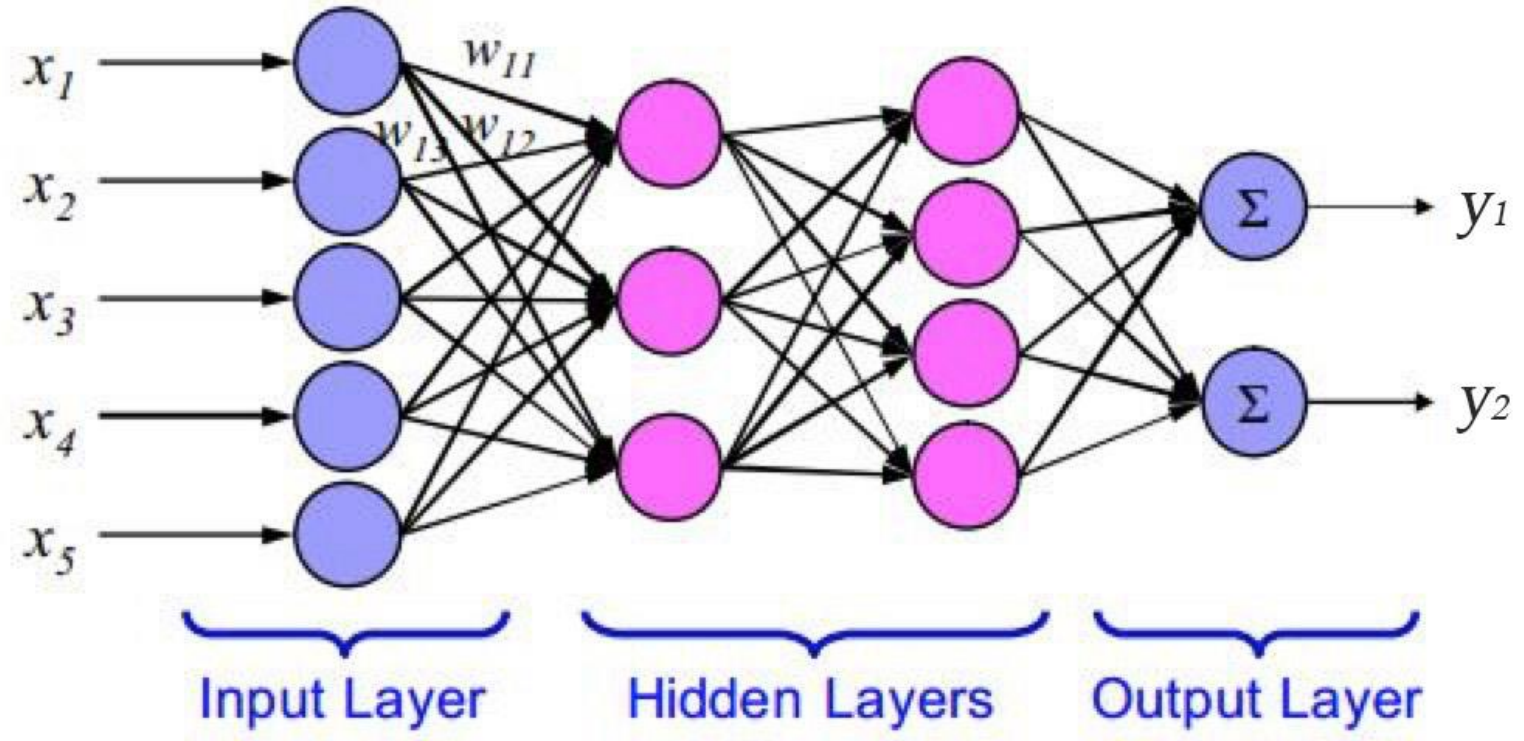
Back Propagation Neural Network

BP(Back Propagation)神经网络

1986年，由McClelland和Rumelhart为首的科学家小组提出，解决了多层神经网络的学习问题，极大促进了神经网络的发展。

BP神经网络也是整个人工神经网络体系中的精华，广泛应用于分类识别，逼近，回归，压缩等领域。在实际应用中，大约80%的神经网络模型都采取了BP网络或BP网络的变化形式。

网络结构


x_1
x_2
x_3
x_4
x_5
w_{11}
w_{12}
w_{13}
Σ
Σ
y_1
y_2
Input Layer
Hidden Layers
Output Layer

BP算法

$$E = \frac{1}{2}(t-y)^2$$

Delta学习规则

$$\frac{\partial E}{\partial W^l} = -\left(X^l\right)^T \delta^l$$

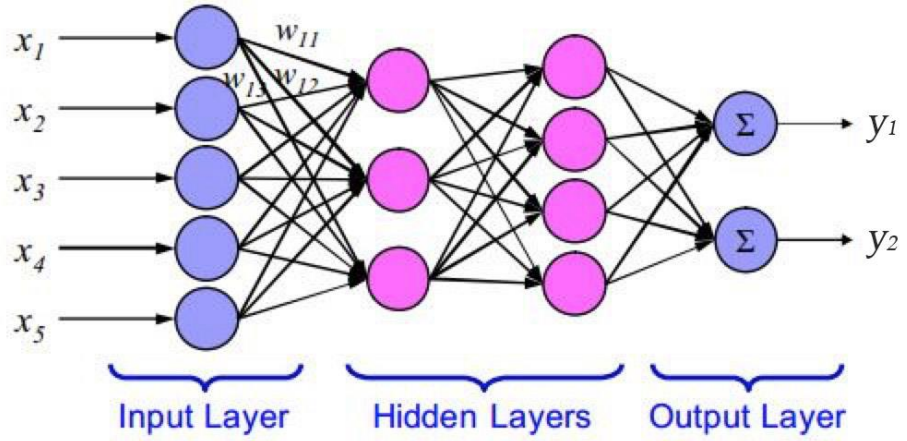
$$\Delta W^l = -\eta \frac{\partial E}{\partial W^l} = \eta\left(X^l\right)^T \delta^l$$

$$\delta^L = (t-y) f'(X^L W^L)$$

$$\delta^l = \delta^{l+1}\left(W^{l+1}\right)^T f'(X^l W^l)$$

δ^l第l层学习信号
δ^L输出层学习信号
W^l第l-l+1层权值
X^l第l层的输入信号

BP算法



$$\Delta W^{l3} = -\eta\delta^{l3}X^{l3} = \eta(t-y)f'\left(X^{l3}W^{l3}\right)X^{l3}$$

$$\delta^{l3} = (t-y)f'(X^{l3}W^{l3})$$

$$\Delta W^{l2} = -\eta\delta^{l2}X^{l2} = \eta\delta^{l3}\left(W^{l3}\right)^{T}f'(X^{l2}W^{l2})X^{l2}$$

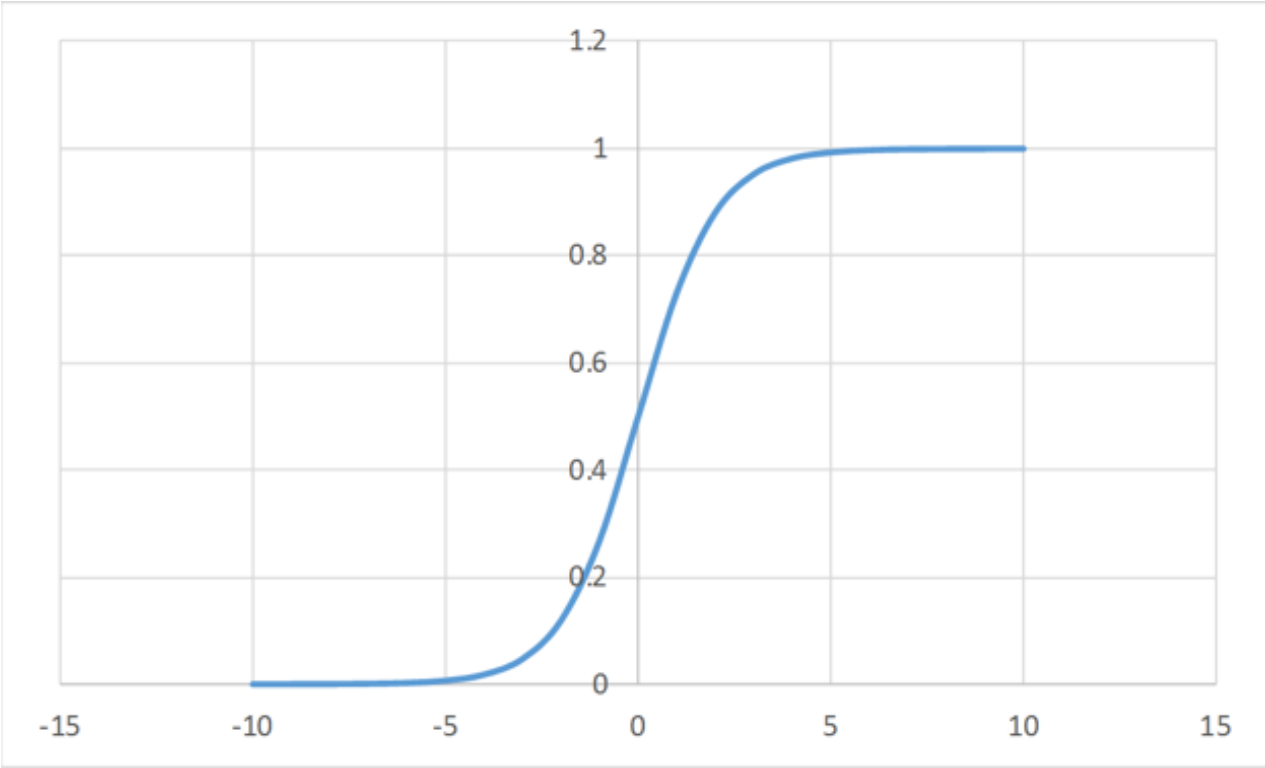
$$\delta^{l2} = \delta^{l3}\left(W^{l3}\right)^{T}f'\left(X^{l2}W^{l2}\right)$$

$$\Delta W^{l1} = -\eta\delta^{l1}X^{l1} = \eta\delta^{l2}\left(W^{l2}\right)^{T}f'\left(X^{l1}W^{l1}\right)X^{l1}$$

$$\delta^{l1} = \delta^{l2}\left(W^{l2}\right)^{T}f'\left(X^{l1}W^{l1}\right)$$

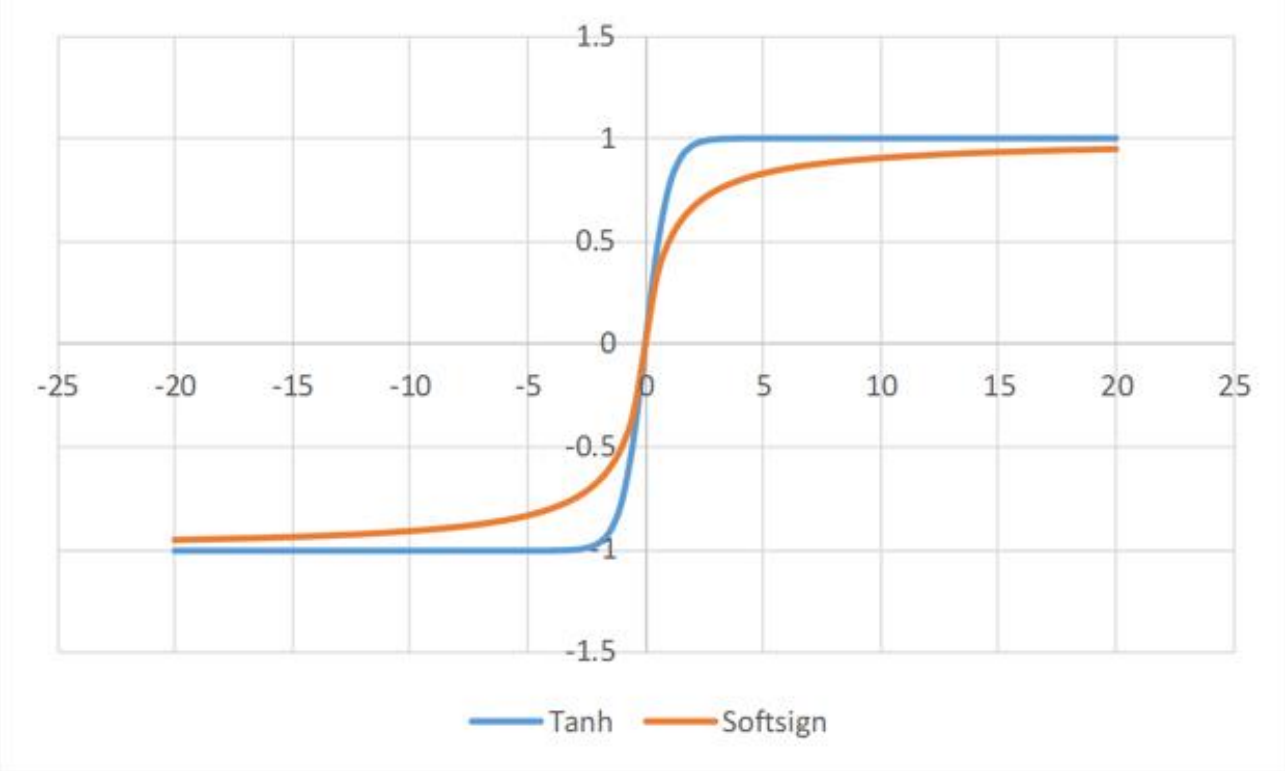
Sigmoid函数

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

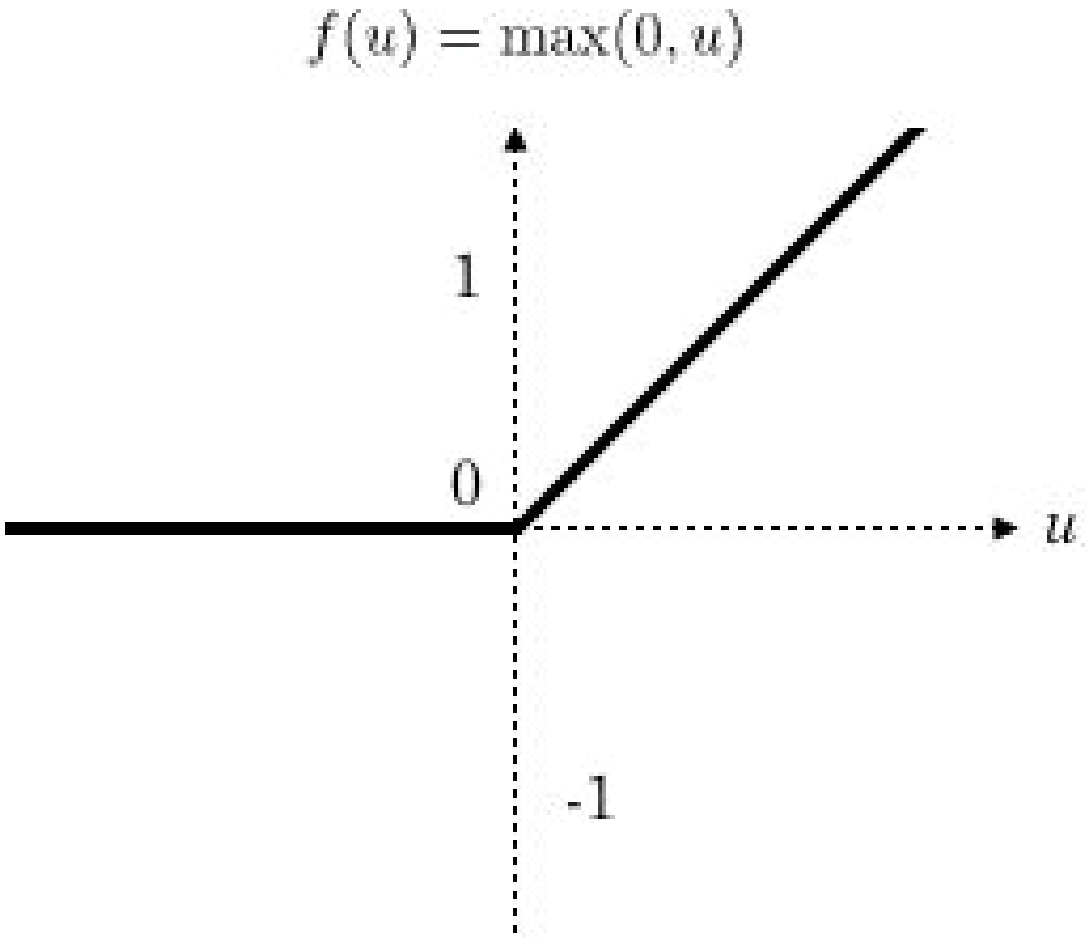


Tanh函数和Softsign函数

Tanh函数：$\frac{e^x - e^{-x}}{e^x + e^{-x}}$　　Softsign函数：$\frac{x}{1 + |x|}$



ReLU函数



BP网络解决异或问题

BP网络-手写数字识别

混淆矩阵

假设有一个用来对猫（cats）、狗（dogs）、兔子（rabbits）进行分类的系统，混淆矩阵就是为了进一步分析性能而对该算法测试结果做出的总结。

		Predicted class		
		Cat	Dog	Rabbit
Actual class	Cat	5	3	0
	Dog	2	3	1
	Rabbit	0	2	11

在这个混淆矩阵中，实际有 8只猫，但是系统将其中3只预测成了狗。对于6条狗，其中有1条被预测成了兔子，2条被预测成了猫。

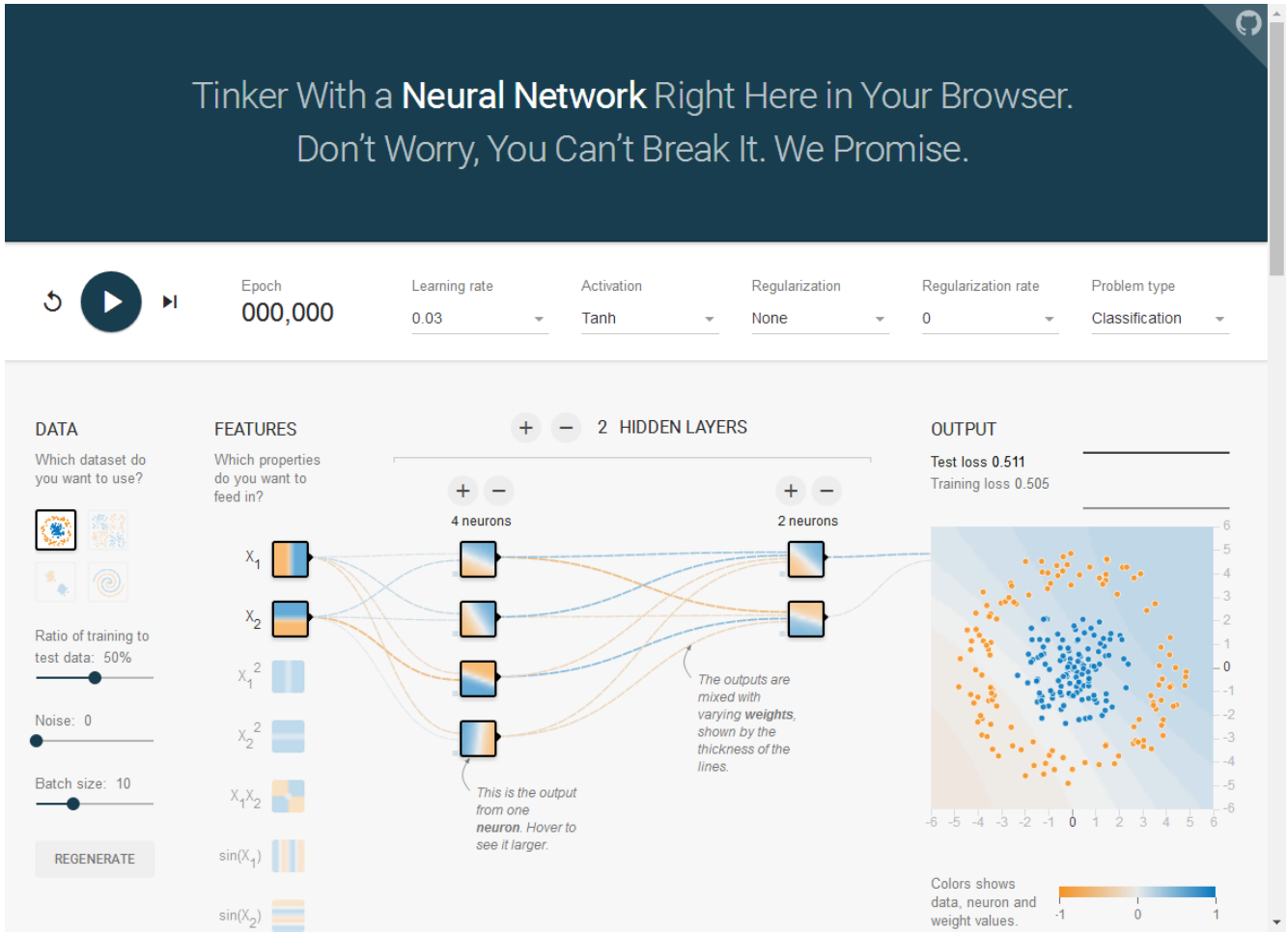
sklearn-神经网络-手写数字识别

论文

Understanding the difficulty of training deep feedforward neural networks

Google神经网络演示平台

http://playground.tensorflow.org/



葡萄酒品质分类

我们将使用一个葡萄酒数据集。它具有不同葡萄酒的各种化学特征，均在意大利同一地区生长，但数据标签分类为三种不同的品种。我们将尝试建立一个可以根据其化学特征对葡萄酒品种进行分类的神经网络模型。

sklearn-神经网络-葡萄酒分类

AI MOOC

「关注 AI MOOC 官方公众号」

这里持续分享
Python、机器学习和深度学习的
各种资源干货及
有趣又实用的硬知识

AI MOOC

覃秉丰

AI MOOC创始人

网易计算机视觉微专业核心讲师

机器学习、深度学习多年开发经验

受邀为中国移动、国家电网、华夏银行、
太平洋保险等世界五百强企业总部做AI内训