



# 深度学习基础

## 深度学习概述

## 授课内容

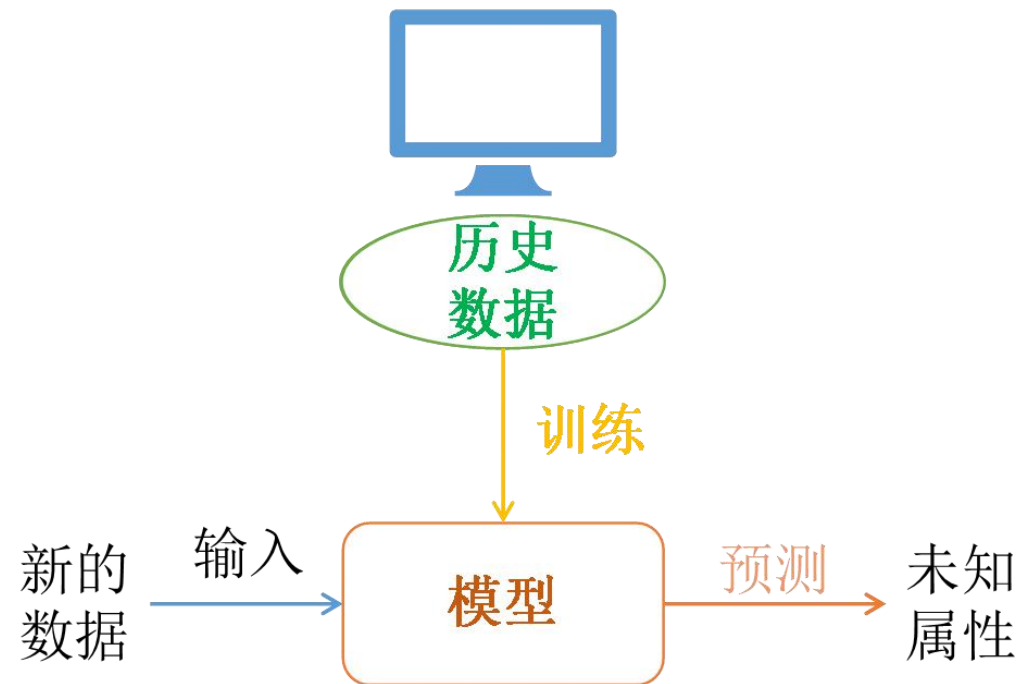
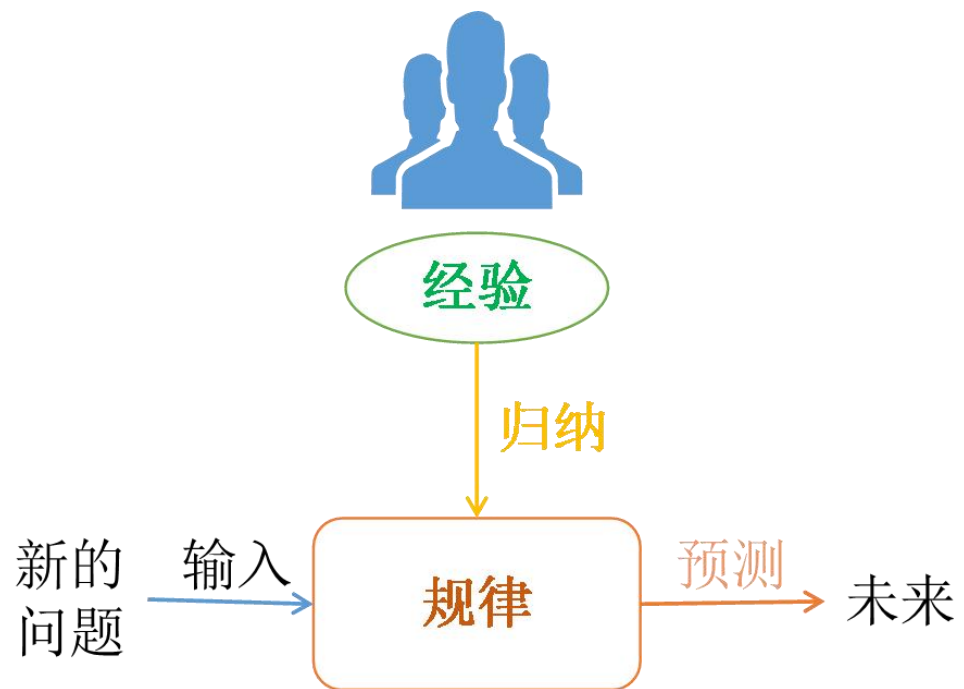
目标：回顾学习深度学习基础内容

掌握技能点：PyTorch框架的使用、BP神经网络、全连接神经网络等基础知识点；

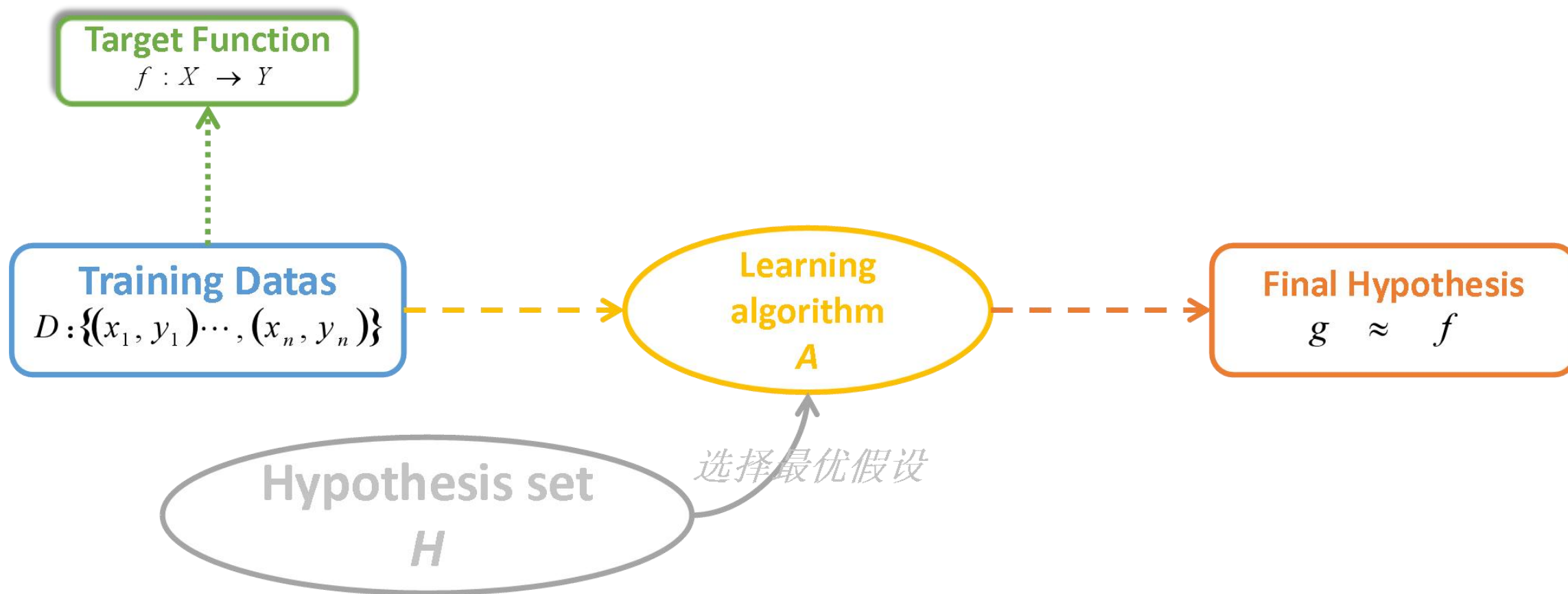
## 深度学习应用

- 1、大规模图片识别：图像分类、人脸识别、车牌识别等
- 2、以图搜图
- 3、**目标检测**
- 4、图像分割
- 5、图像生成
- 6、推荐系统
- 7、搜索系统
- 8、聊天机器人
- 9、命名实体识别
- 10、文本检索系统

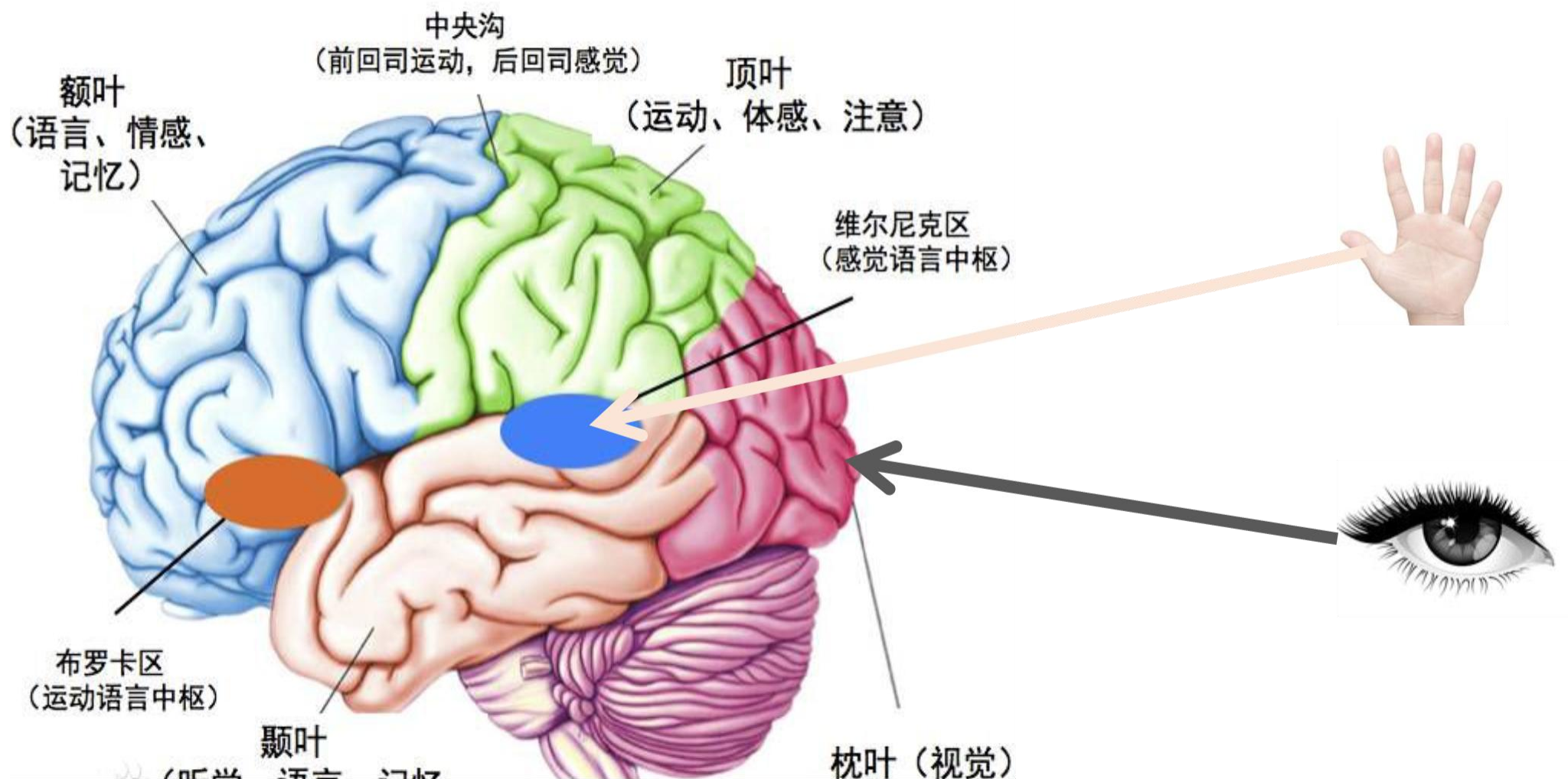
# 机器学习回顾



# 机器学习回顾

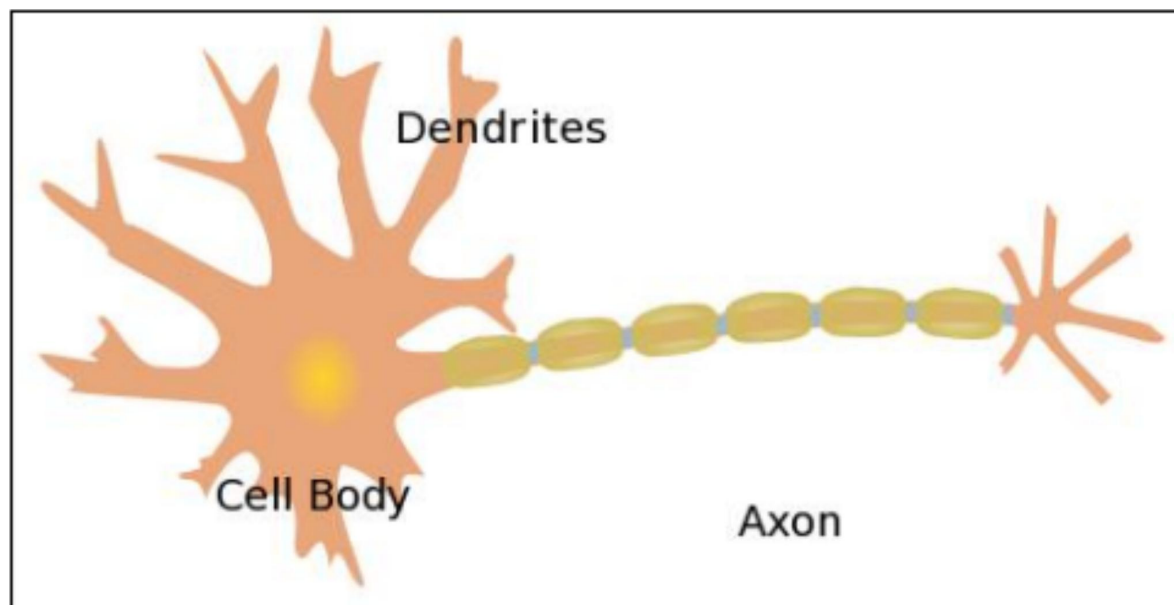


# 神经网络的来源



## 神经网络的来源

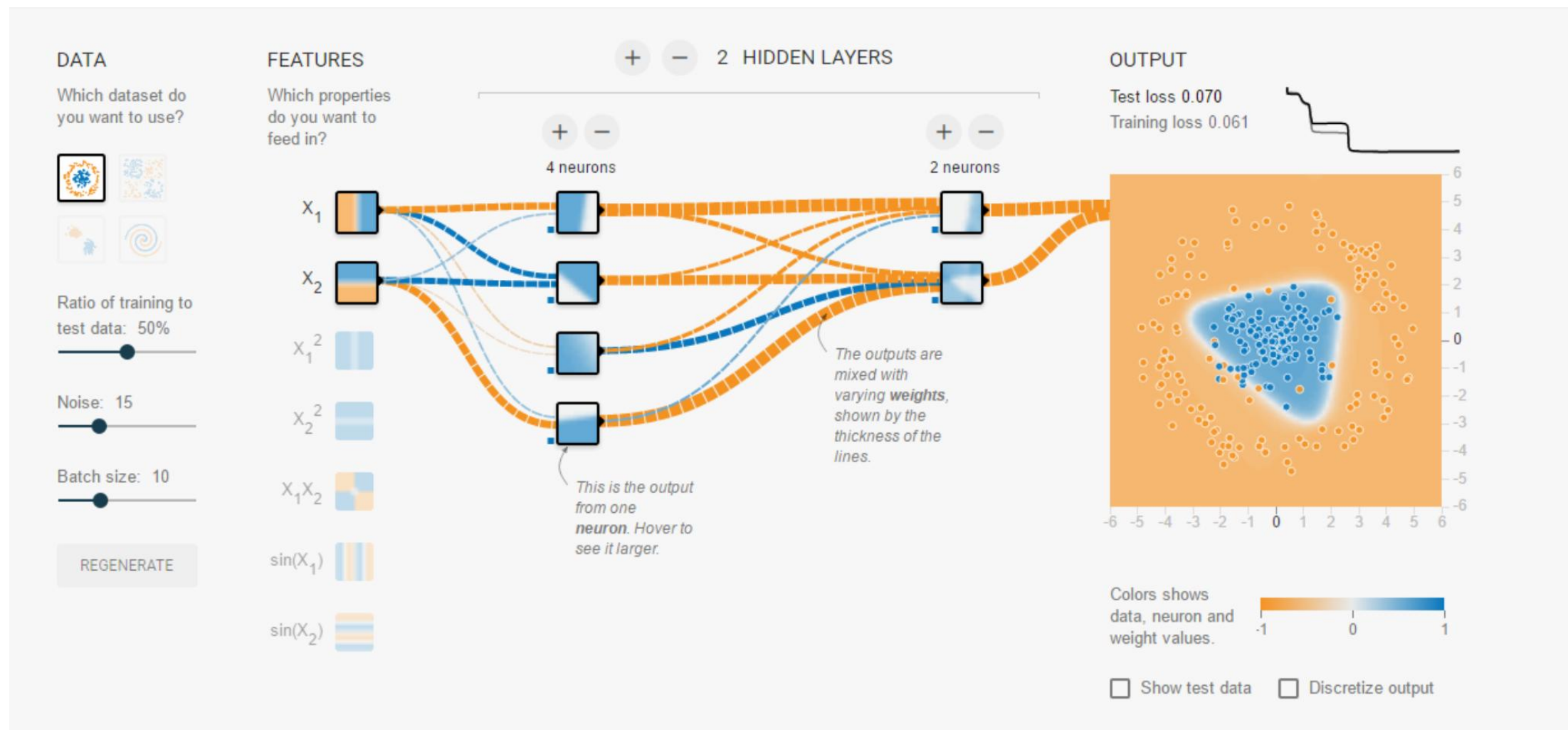
- 大脑是由处理信息的神经元细胞和连接神经元的细胞进行信息传递的突触构成的。树突(Dendrites)从一个神经元接受电信号，信号在细胞核(Cell Body)处理后，然后通过轴突(Axon)将处理的信号传递给下一个神经元。
- 一个神经元可以看作是将一个或多个输入处理成一个输出的计算单元。
- 通过多个神经元的传递，最终大脑会得到这个信息，并可以对这个信息给出一个合适的反馈。





# 直观案例

## 案例理解

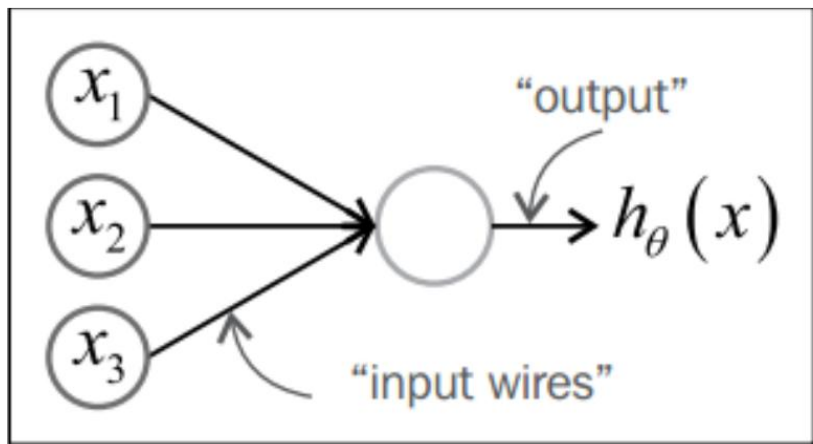




## 感知器模型/Logistic回归

感知器是一种模拟人的神经元的一种算法模型，是一种研究单个训练样本的二元分类器，是SVM和人工神经网络(ANN, Artificial Neural Networks)的基础。

一个感知器接受几个二进制的输入，并产生一个二进制的输出，通常的表达方式如下：

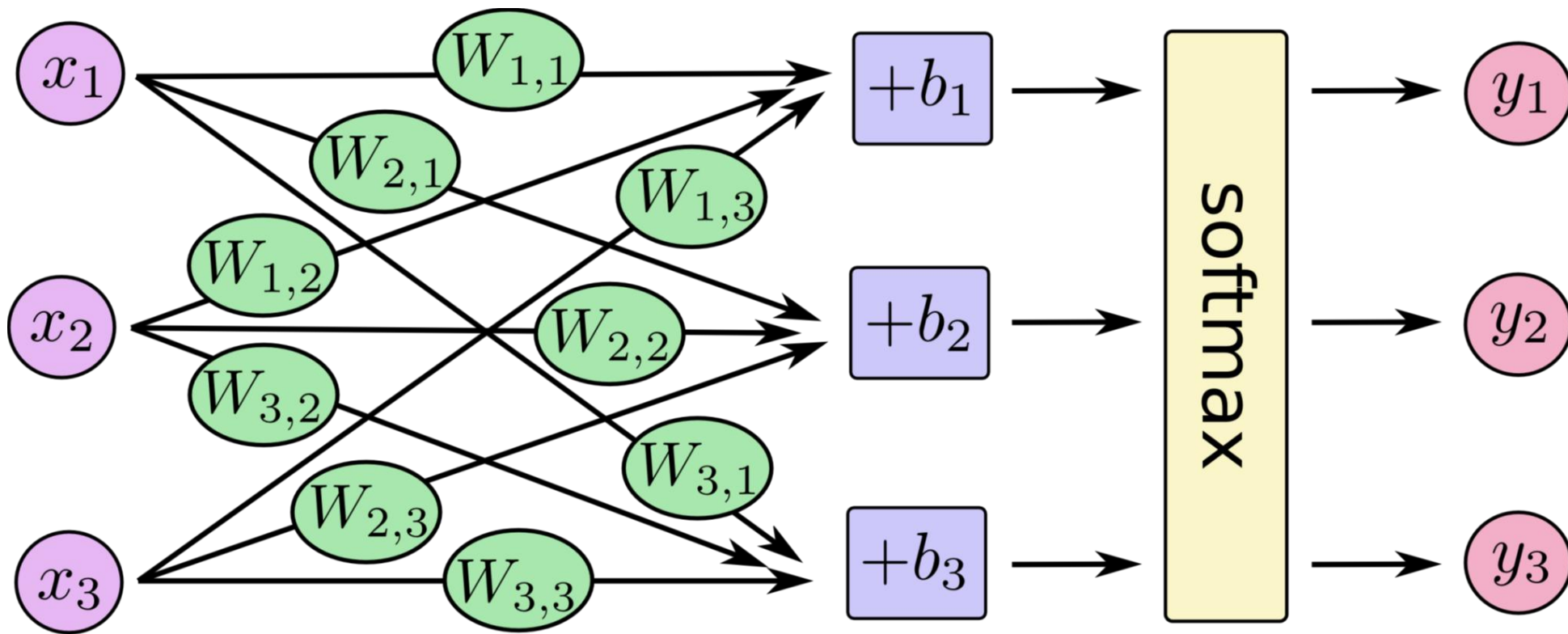


$$output = \begin{cases} 0, & \text{if } \sum_j w_j x_j \leq threshold \\ 1, & \text{if } \sum_j w_j x_j > threshold \end{cases}$$

$$output = \begin{cases} 0, & \text{if } w \cdot x + b \leq 0 \\ 1, & \text{if } w \cdot x + b > 0 \end{cases}$$

## Softmax回顾

Softmax回归是logistic回归的一般化，适用于K分类的问题。



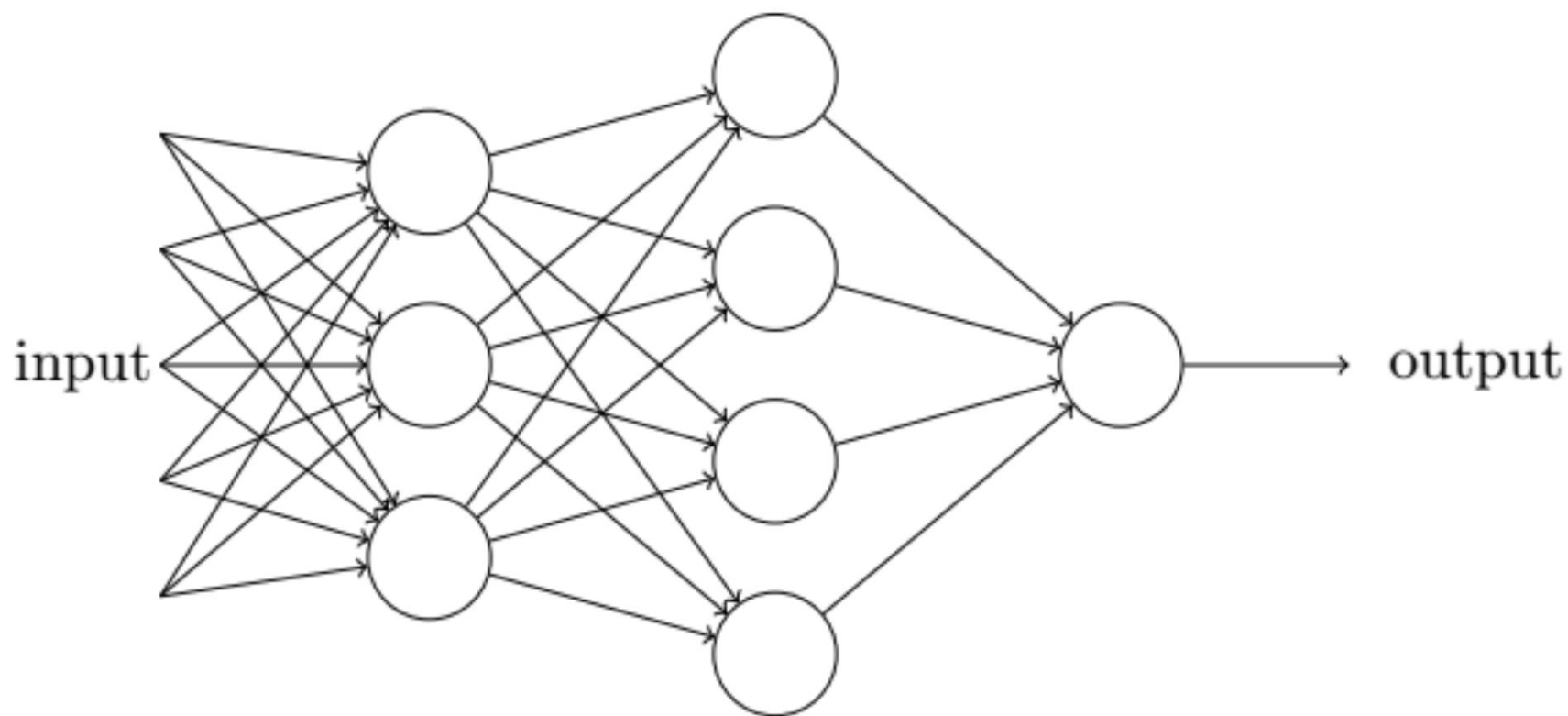
## Softmax回顾

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \left( \begin{bmatrix} W_{1,1}x_1 + W_{1,2}x_2 + W_{1,3}x_3 + b_1 \\ W_{2,1}x_1 + W_{2,2}x_2 + W_{2,3}x_3 + b_2 \\ W_{3,1}x_1 + W_{3,2}x_2 + W_{3,3}x_3 + b_3 \end{bmatrix} \right)$$

$$\begin{bmatrix} y_1 \\ y_2 \\ y_3 \end{bmatrix} = \text{softmax} \left( \begin{bmatrix} W_{1,1} & W_{1,2} & W_{1,3} \\ W_{2,1} & W_{2,2} & W_{2,3} \\ W_{3,1} & W_{3,2} & W_{3,3} \end{bmatrix} \cdot \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} + \begin{bmatrix} b_1 \\ b_2 \\ b_3 \end{bmatrix} \right)$$

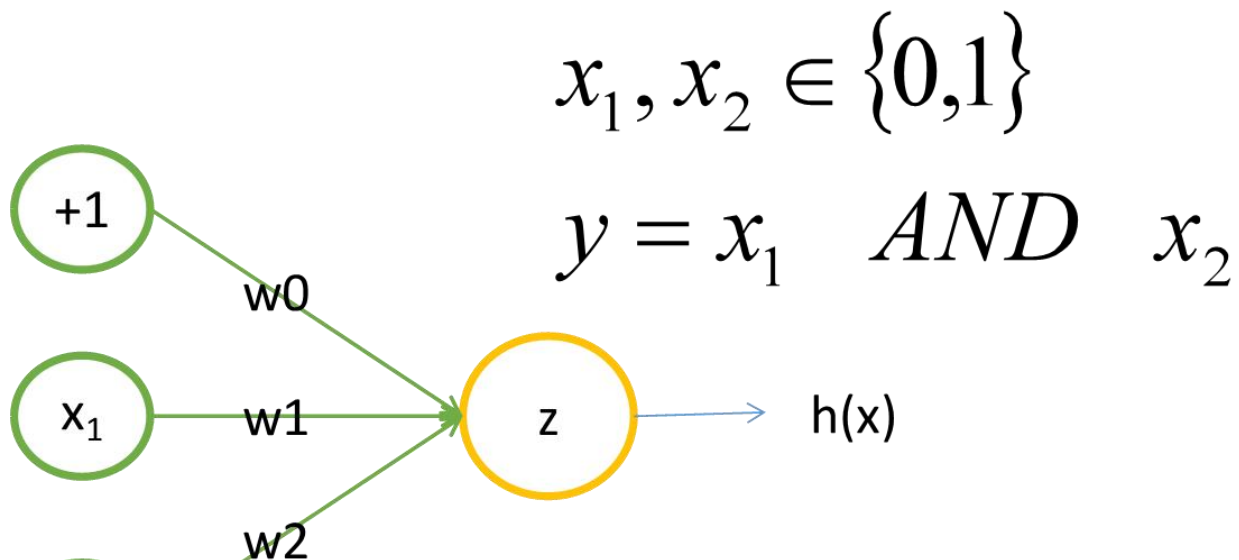
## 多层感知机

将多个感知器进行组合，就可以得到一个**多层感知器**的网络结构，网络中的每个节点我们叫做**神经元**，这个结构也就叫做神经网络。



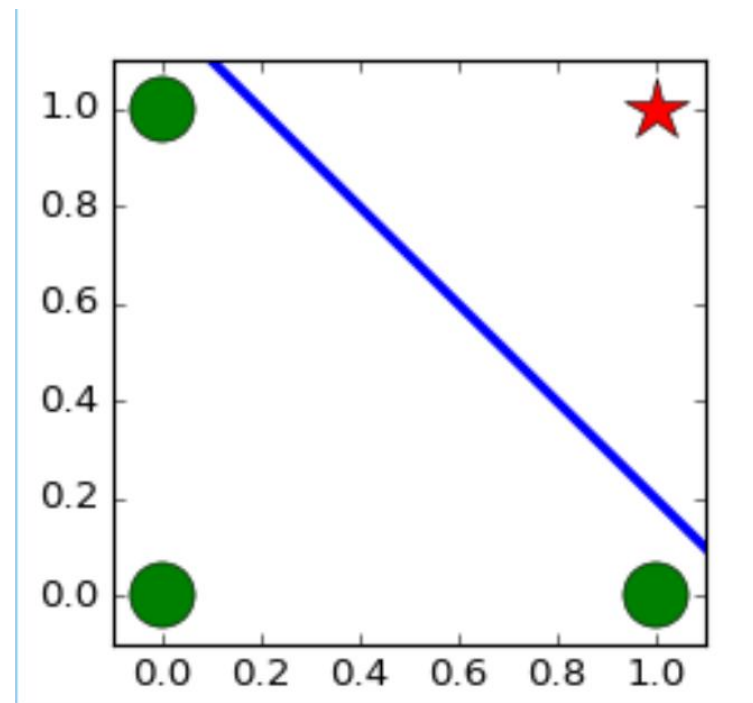
## 感知器神经元直观理解之逻辑与

单个神经元完成**逻辑与**功能



$$W = (-3, 2, 2)$$

$$h_w(z) = h(-3 + 2 \cdot x_1 + 2 \cdot x_2) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



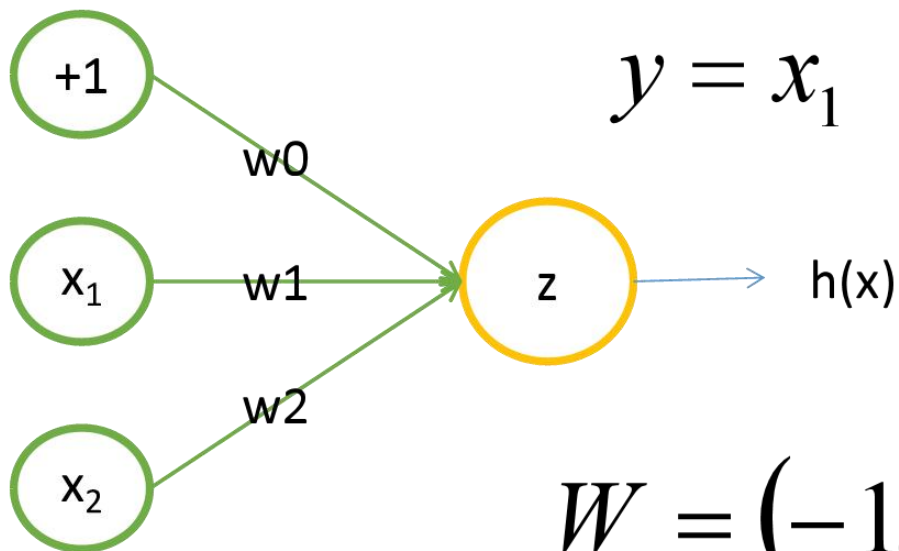
$x_1$	$x_2$	$h_{\Theta}(x)$
0	0	0
0	1	0
1	0	0
1	1	1

## 感知器神经元直观理解之逻辑或

单个神经元完成逻辑或功能

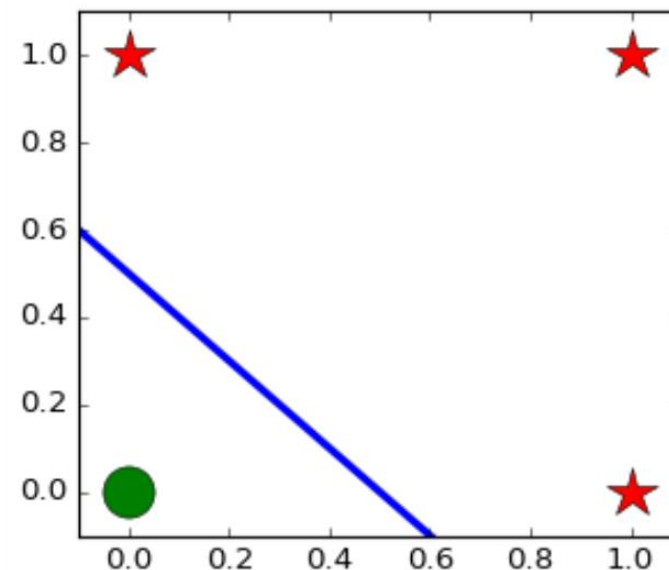
$$x_1, x_2 \in \{0, 1\}$$

$$y = x_1 \text{ OR } x_2$$



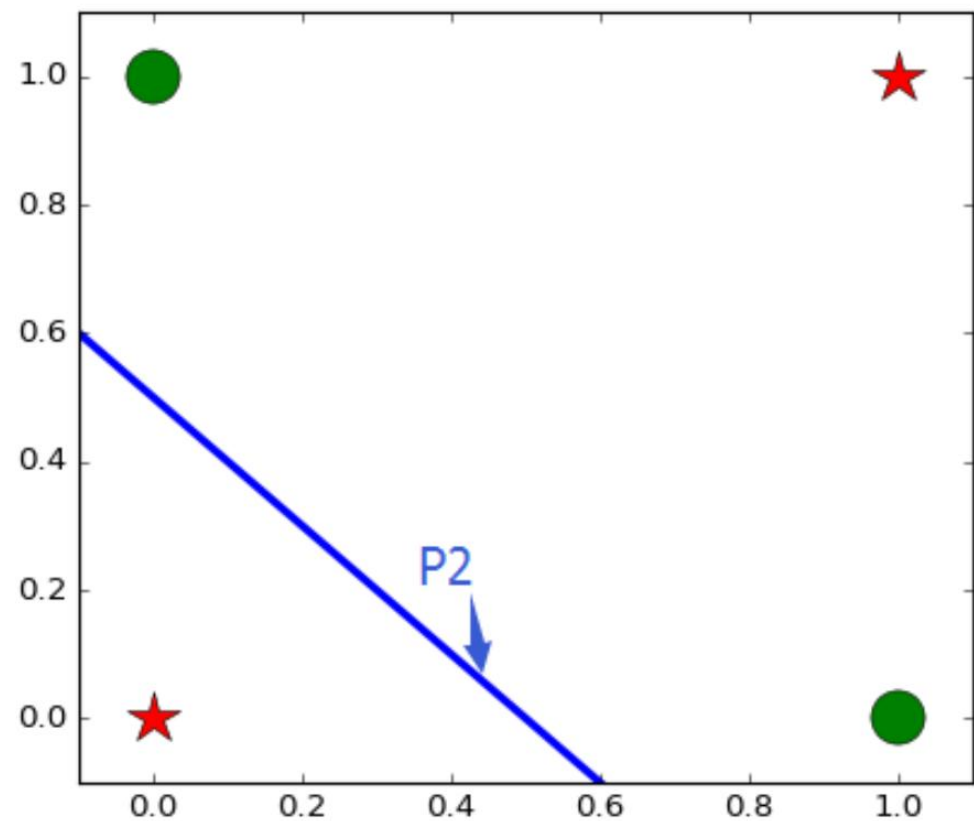
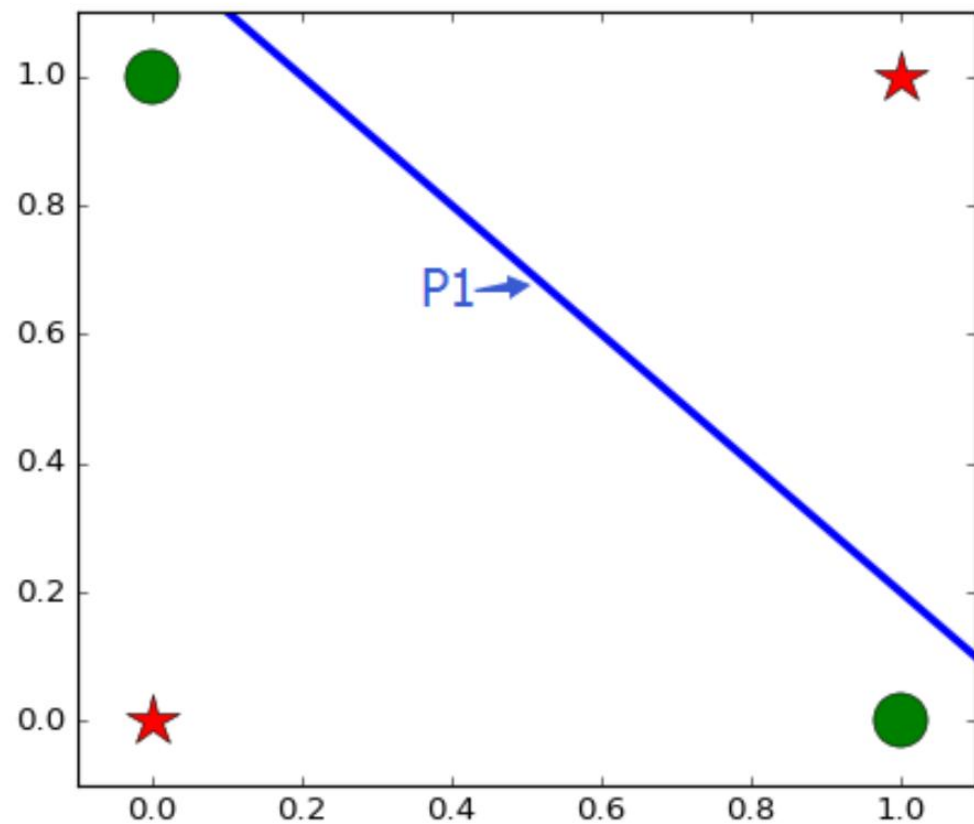
$$W = (-1, 2, 2)$$

$$h_w(z) = h(-1 + 2 \cdot x_1 + 2 \cdot x_2) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$



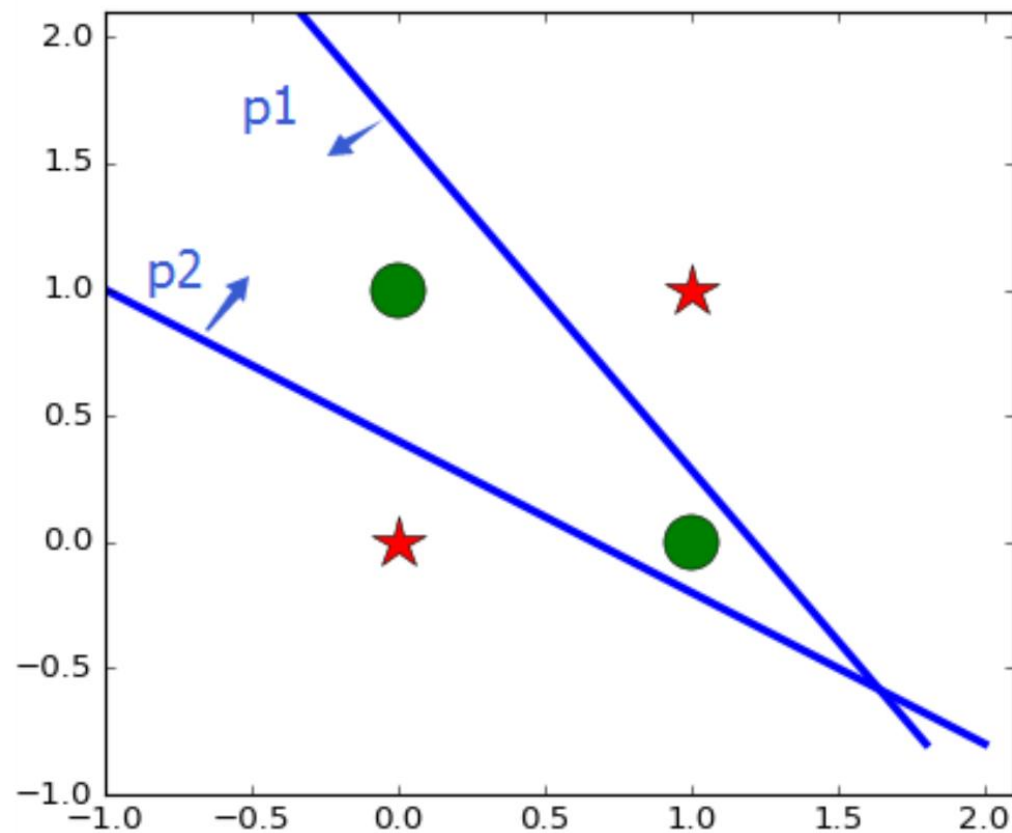
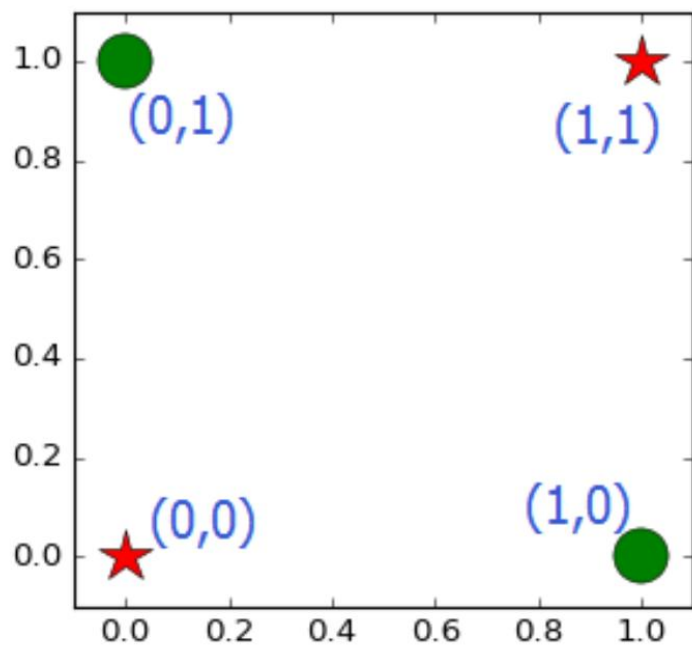
$x_1$	$x_2$	$h_{\Theta}(x)$
0	0	0
0	1	1
1	0	1
1	1	1

## 感知器神经元直观理解之非线性可分

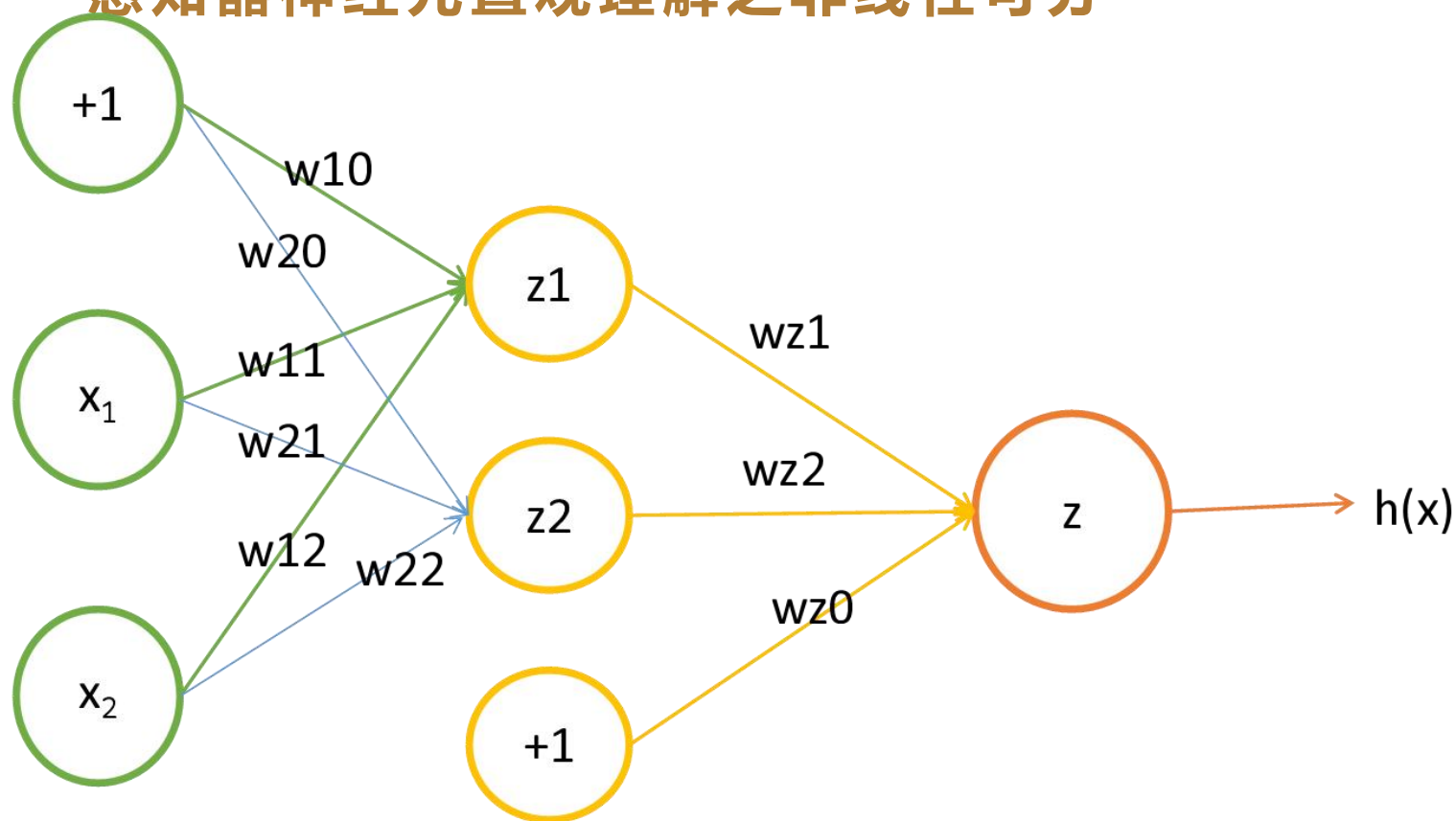




## 感知器神经元直观理解之非线性可分



## 感知器神经元直观理解之非线性可分



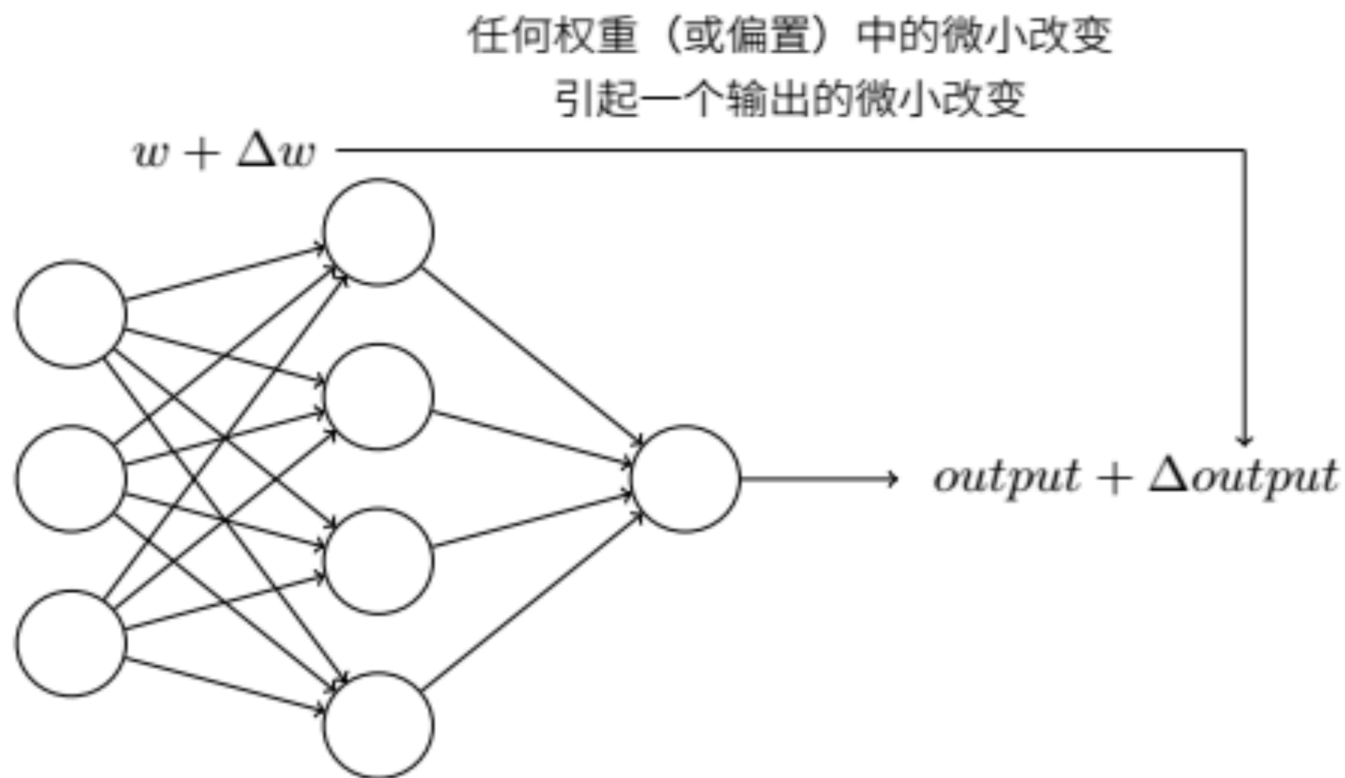
$x_1$	$x_2$	$z_1$	$z_2$	$h(x)$
0	0	0	0	1
0	1	0	1	0
1	0	0	1	0
1	1	1	1	1

$$W_1 = (-3, 2, 2) \quad W_2 = (-1, 2, 2) \quad W_z = (1, 3, -3)$$

$$h_w(z) = h(Wx) = \begin{cases} 0, & z < 0 \\ 1, & z \geq 0 \end{cases}$$

## 感知器网络理解以及S型神经元

其实只要将网络中的权重或者偏置项稍微的做一点小的改动，都会导致最终的输出发生一定的变化。但是在感知器神经网络中，单个感知器上的权重或者偏置项发现一点小的变化，最终的输出要不不变，要不完全翻转(因为只有两种取值0和1)，这种翻转会导致接下来的感知器可能发生复杂的完全没法控制的变化，这样会导致我们的网络很难得到最终的逼近结果。

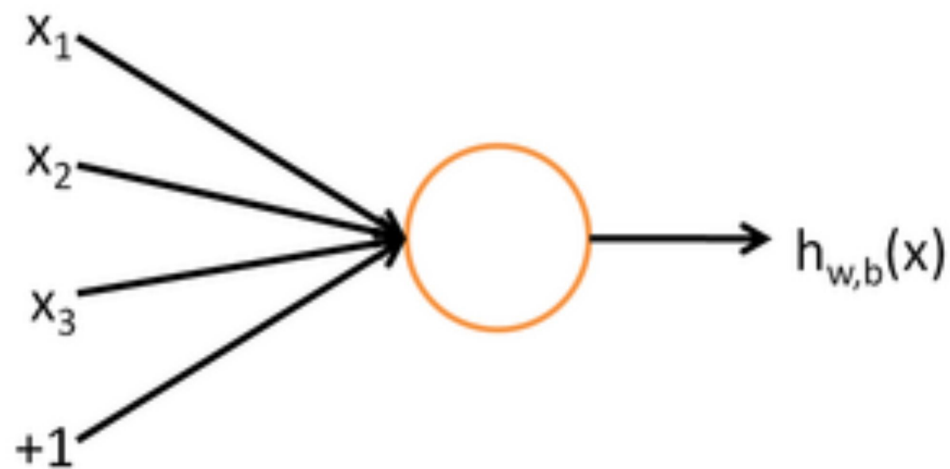


## 感知器网络理解以及S型神经元

1. 针对感知器网络的这种很难学习的问题，引入S型神经元来代替感知器，从而解决这个问题。
2. 从感知器模型中，我们可以将单个神经元的计算过程看成下列两个步骤：
  - 先计算权重 $w$ 和输入值 $x$ 以及偏置项 $b$ 之间的线性结果值 $z$ ： $z=wx+b$
  - 然后对结果值 $z$ 进行一个数据的 $\text{sign}$ 函数(变种)转换，得到一个离散的0/1值： $y=\text{int}((\text{sign}(z)+1)/2)$
3. 在S型神经元中，和感知器神经元的区别在于：
  - 对于结果值 $z$ 的转换，采用的不是 $\text{sign}$ 函数进行转换，是采用**平滑类型**的函数进行转换，让输出的结果值 $y$ 最终是一个连续的，S型神经元转指使用的是sigmoid函数。该转换函数也被称为**激活函数**。

## 神经网络来源之“神经元”

1. 输入：  $x_1$ 、 $x_2$ 、 $x_3$ 和截距+1;
2. 输出： 函数 $h_{w,b}(x)$ ， 其中 $w$ 权重和 $b$ 偏置项是参数



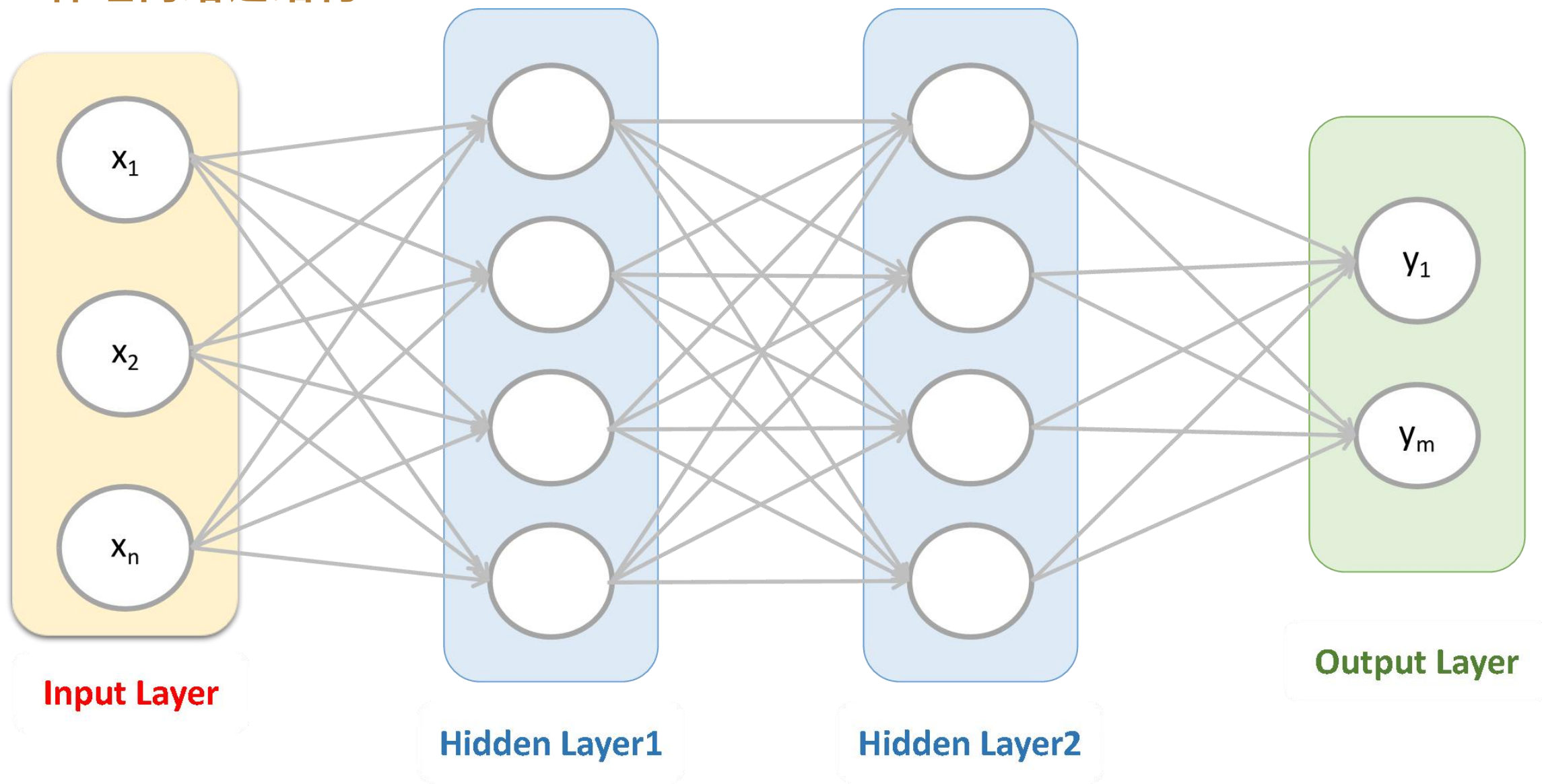
$$h_{W,b}(x) = f(W^T x, b) = f\left(\sum_{i=1}^3 W_i x_i + b\right)$$

## 激活函数

激活函数的主要作用是提供网络的**非线性建模**能力。如果没有激活函数，那么该网络仅能够表达线性映射，此时即便有再多的隐藏层，其整个网络跟单层神经网络也是等价的。因此也可以认为，只有加入了激活函数之后，深度神经网络才具备了分层的非线性映射学习能力。激活函数的主要特性是：**可微性、单调性、输出值的范围可控**；

常见的激活函数：Sign函数、Sigmoid函数、Tanh函数、ReLU函数、P-ReLU函数、Leaky-ReLU函数、ELU函数等

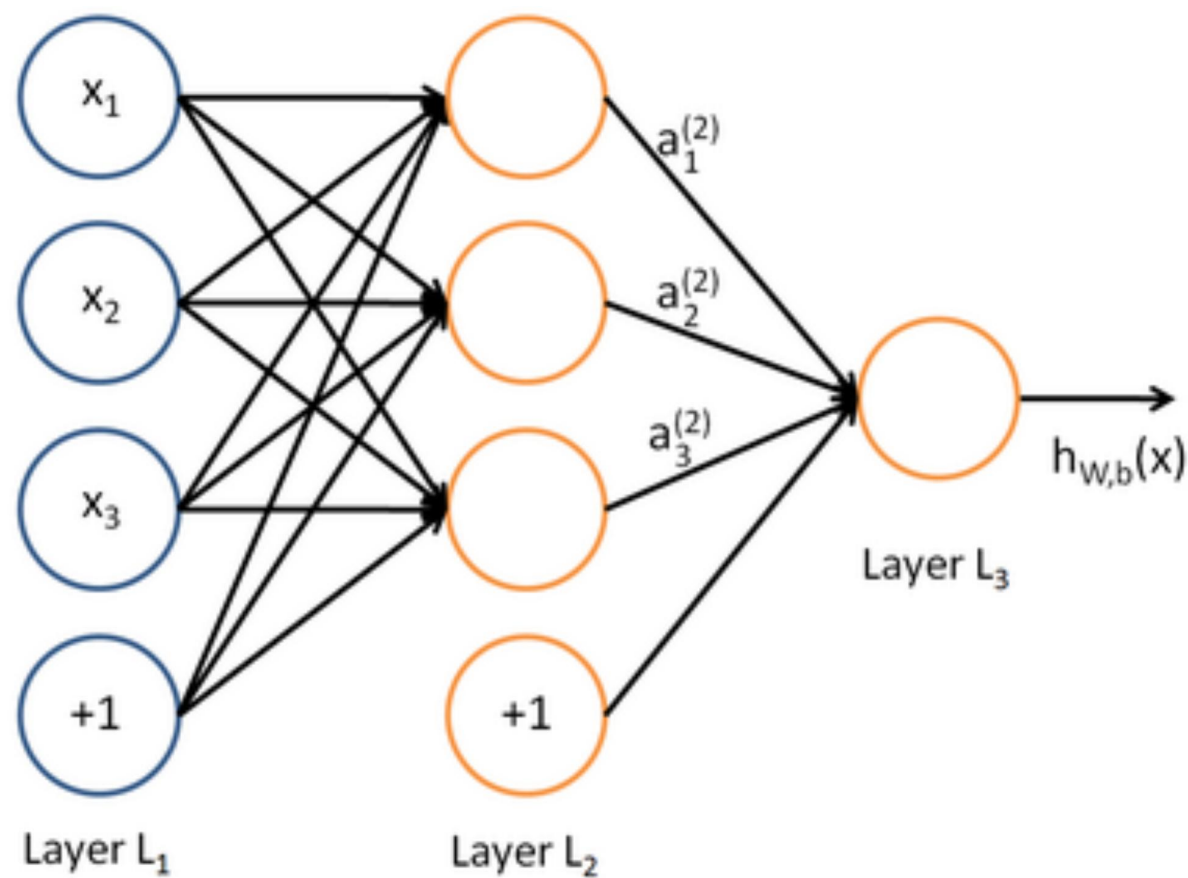
## 神经网络之结构





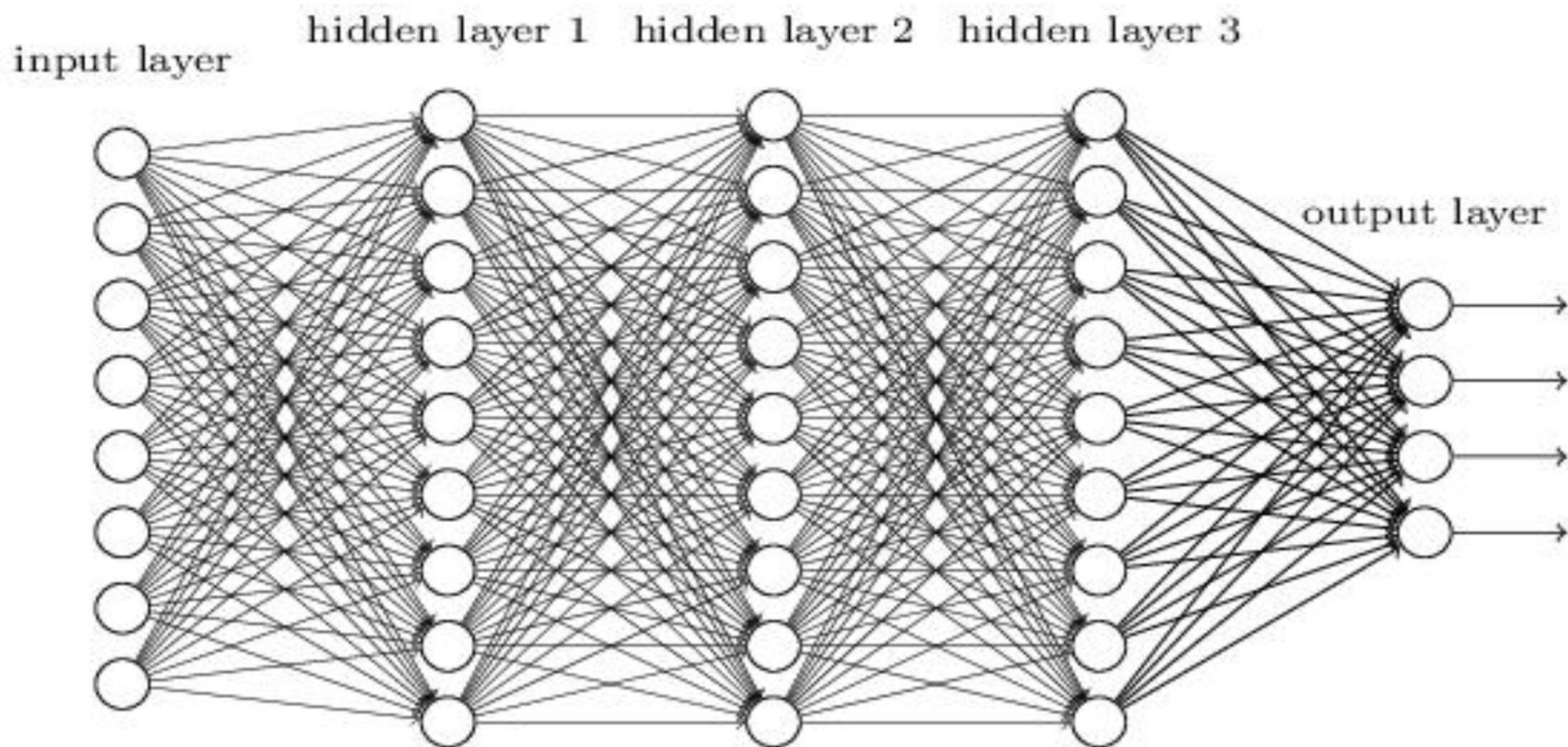
## 神经网络之浅层神经网络

添加少量隐层的神经网络就叫做浅层神经网络；也叫作传统神经网络，一般为2隐层的神经网络(超过两隐层的话，效果会差很多)



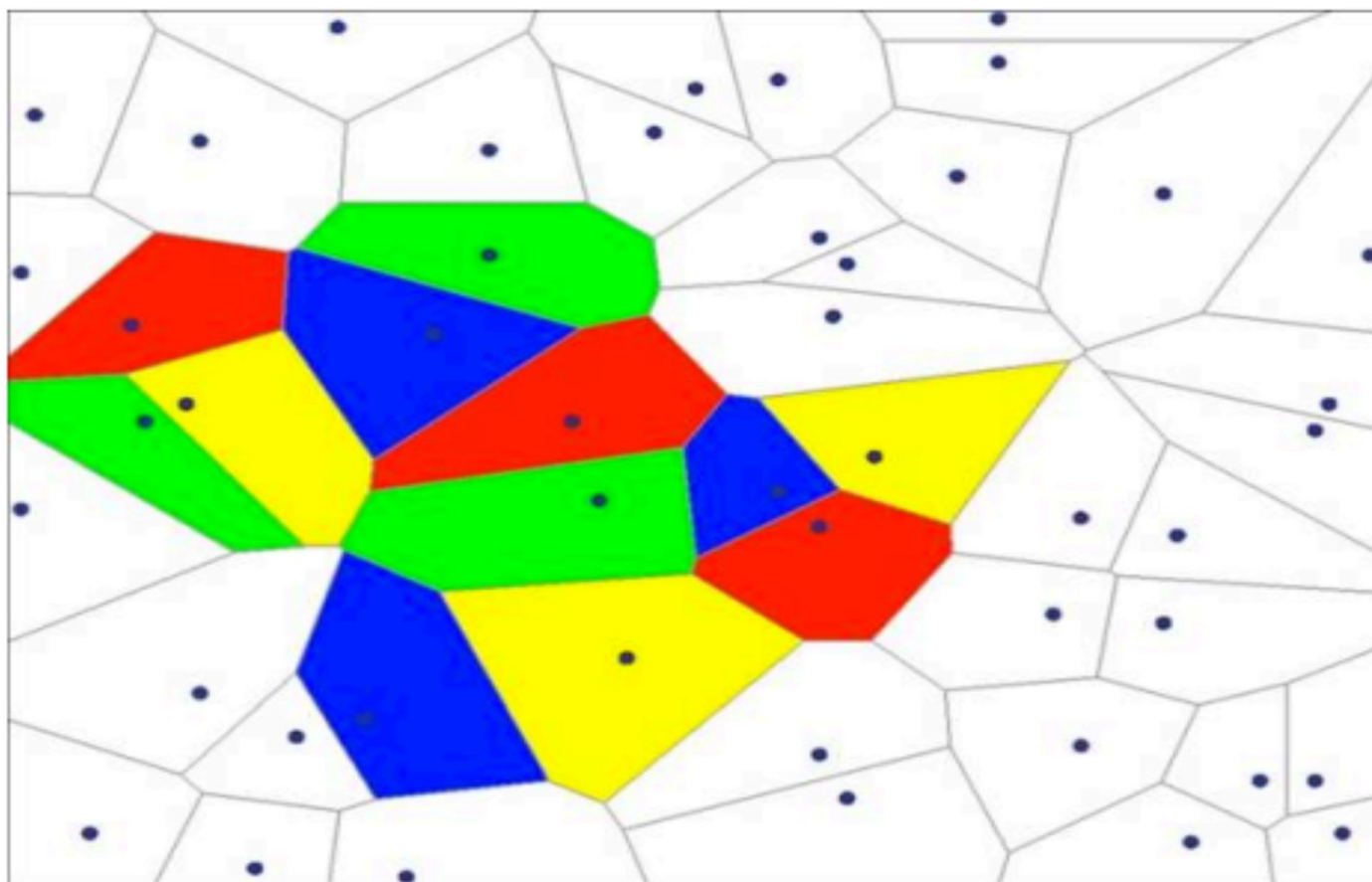
## 神经网络之深度神经网络

增多中间层(隐层)的神经网络就叫做深度神经网络(DNN); 可以认为深度学习是神经网络的一个发展。

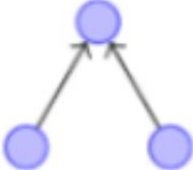

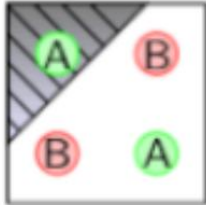
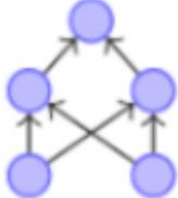

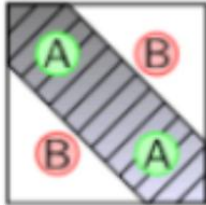
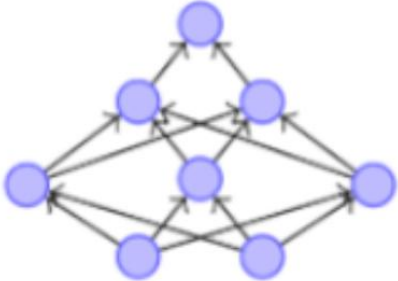

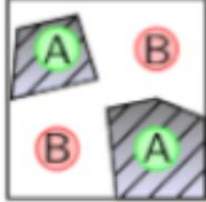


## 神经网络之非线性可分

对线性分类器的与和或的组合可以完成非线性可分的问题；即通过多层的神经网络中加入激活函数的方式可以解决非线性可分的问题。



# 神经网络

结构	决策区域类型	区域形状	异或问题
无隐层 	由一超平面分成两个		
单隐层 	开凸区域或闭凸区域		
双隐层 	任意形状（其复杂度由单元数目确定）		

## 神经网络之过拟合

1. 理论上讲，单隐层的神经网络可以逼近任何连续函数（只要隐层的神经元个数足够的多<一个神经元将数据集分为两类>）
2. 虽然从数学表达上来讲，效果一样，但是在网络工程效果中，多隐层的神经网络效果要比单隐层的神经网络效果好
3. 对于一些分类的问题来讲，三层的神经网络效果优于两层的神经网络，但是如果把层次不断增加(4,5,6,7....)，对于最终的效果不会产生太大的变化
4. 提升隐层层数或者神经元个数，神经网络的“容量”会变大，那么空间表达能力会变强，（模型的预测能力），从而有可能导致过拟合的问题
5. 对于视频/图片识别等问题，传统的神经网络(全连接神经网络)不太适合

THANKS!