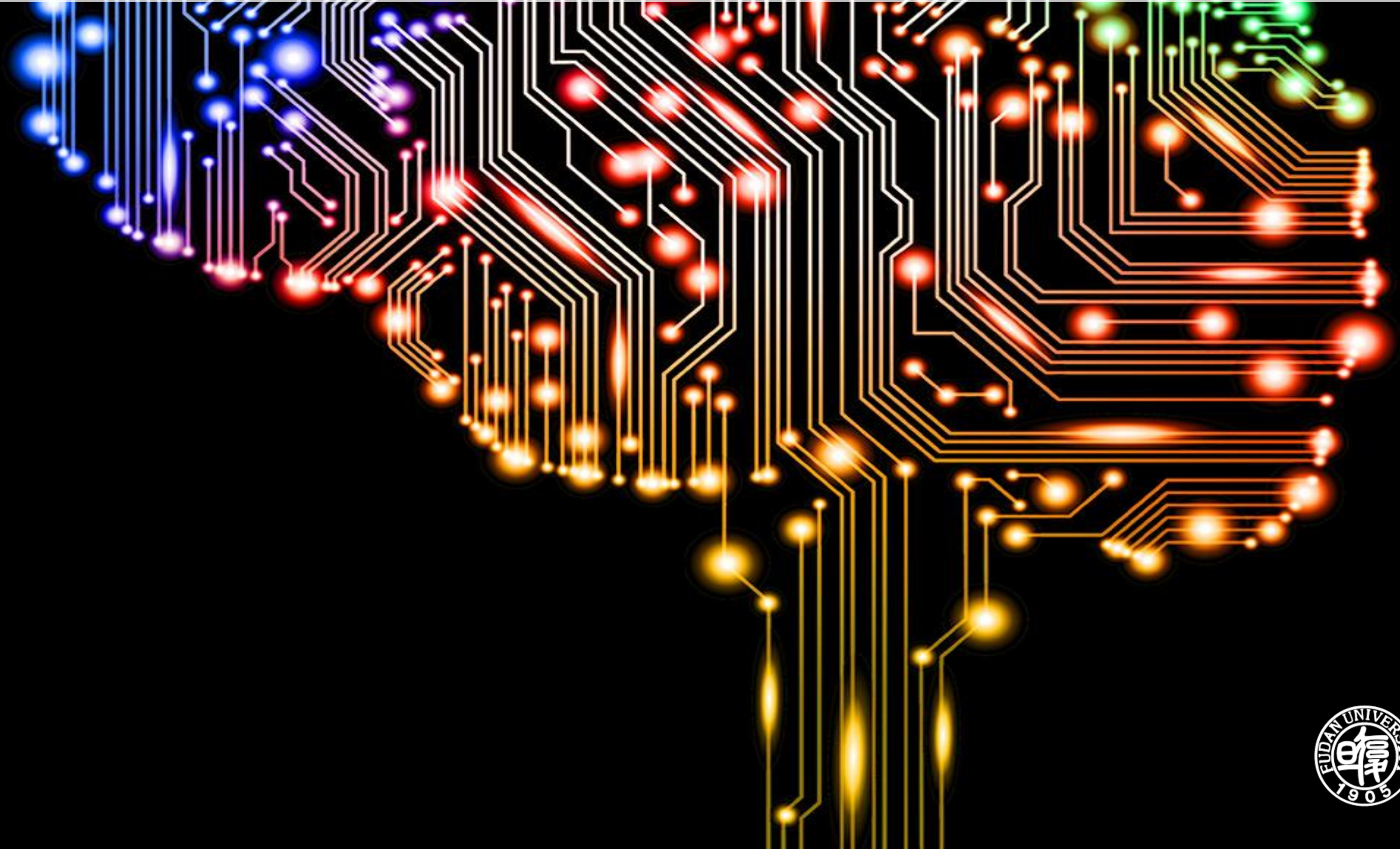
# 数据驱动的人工智能（3a）神经网络基础

## Data Driven Artificial Intelligence

邬学宁 SAP硅谷创新中心

2017 / 03



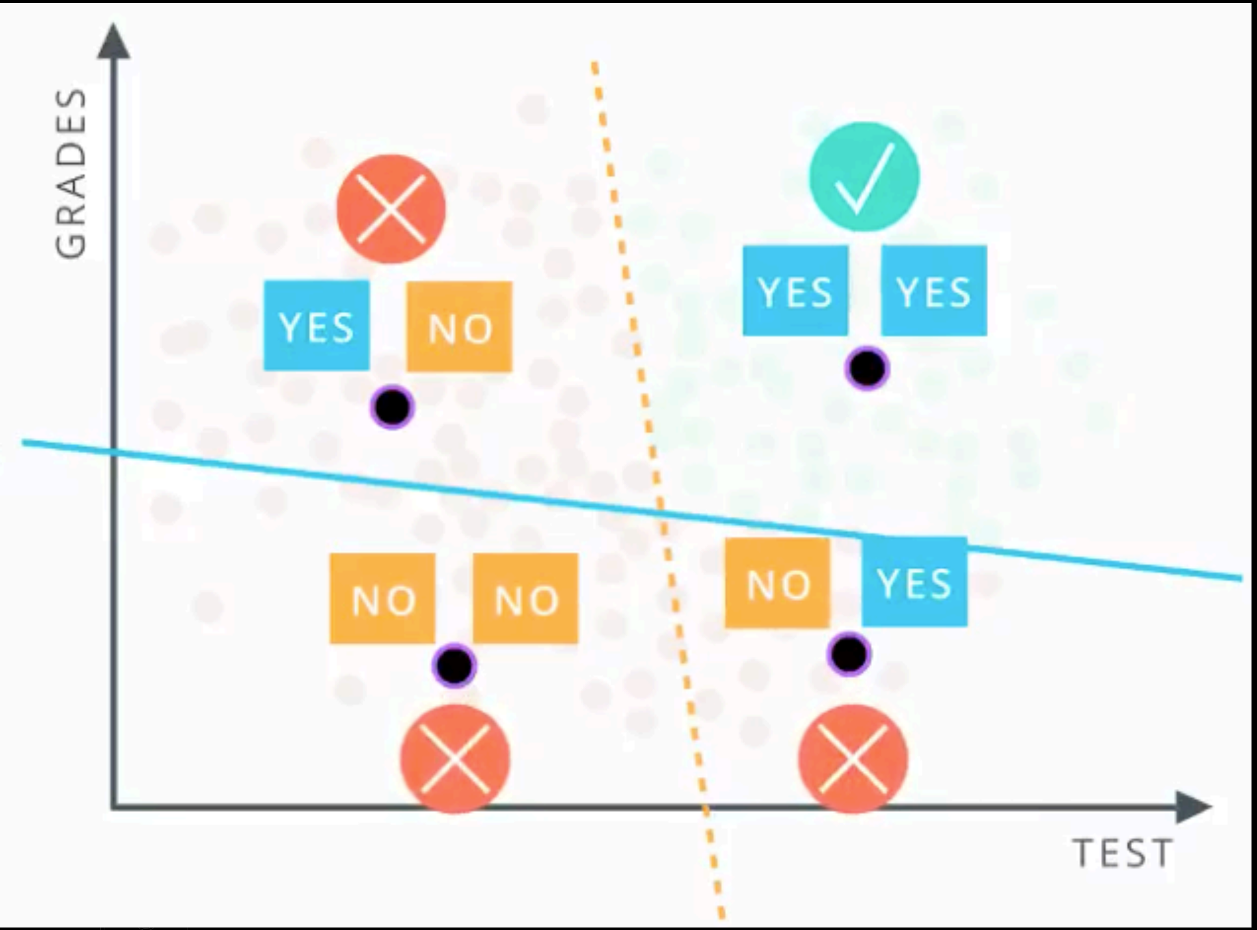
# 日程

From Logistic Regression to ANN
Perceptron
Feeding Forward MLP

# Neural Networks (1) 学生入学问题

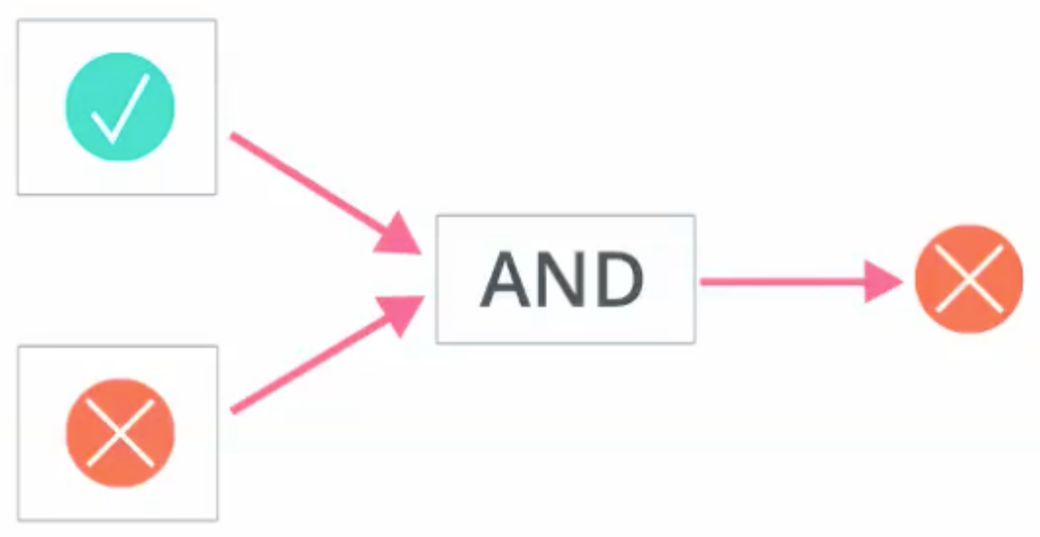

Source: Udacity

# Neural Networks (2) 两个分类器



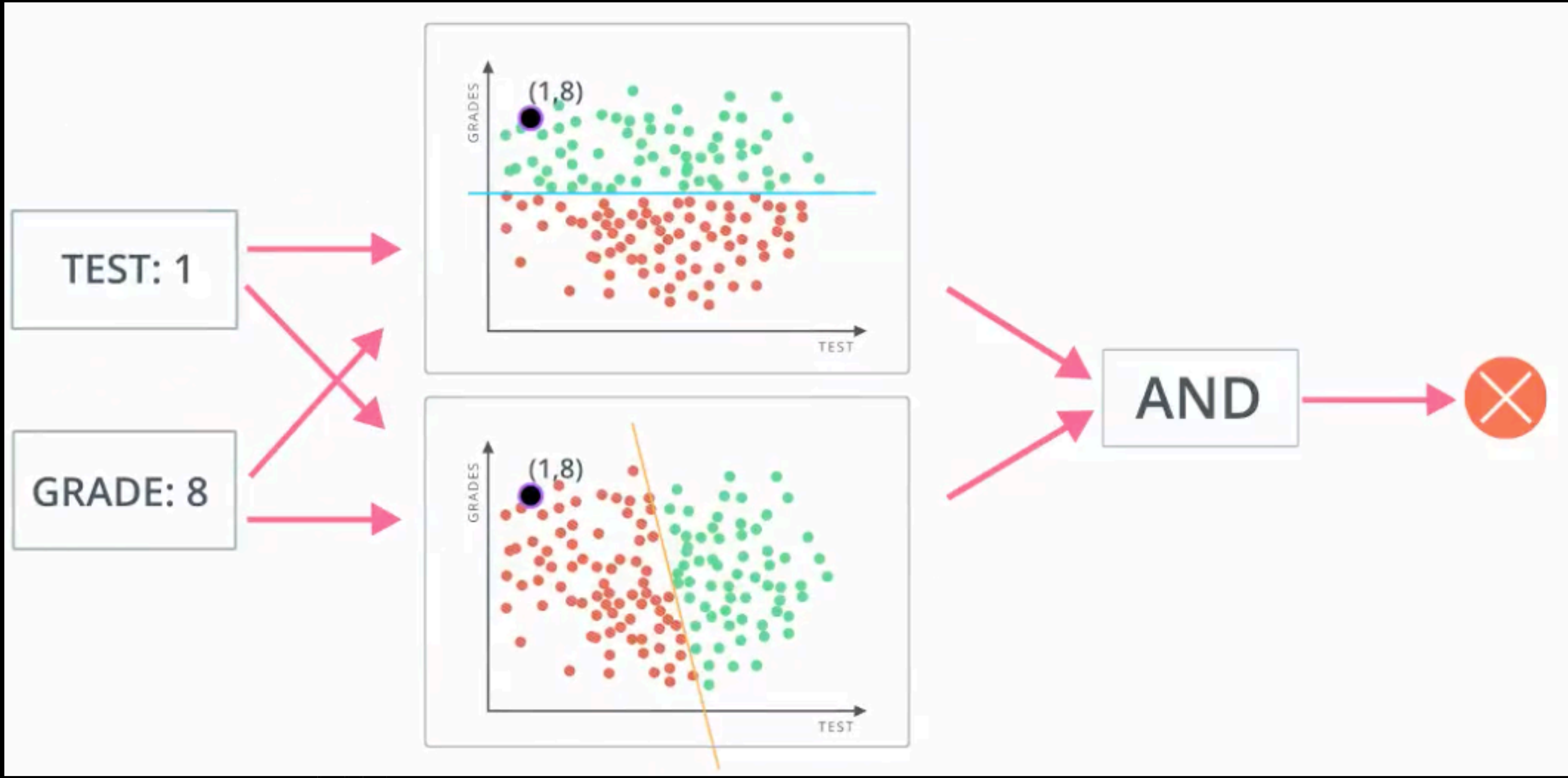

Source: Udacity

# Neural Networks (3) 3个问题

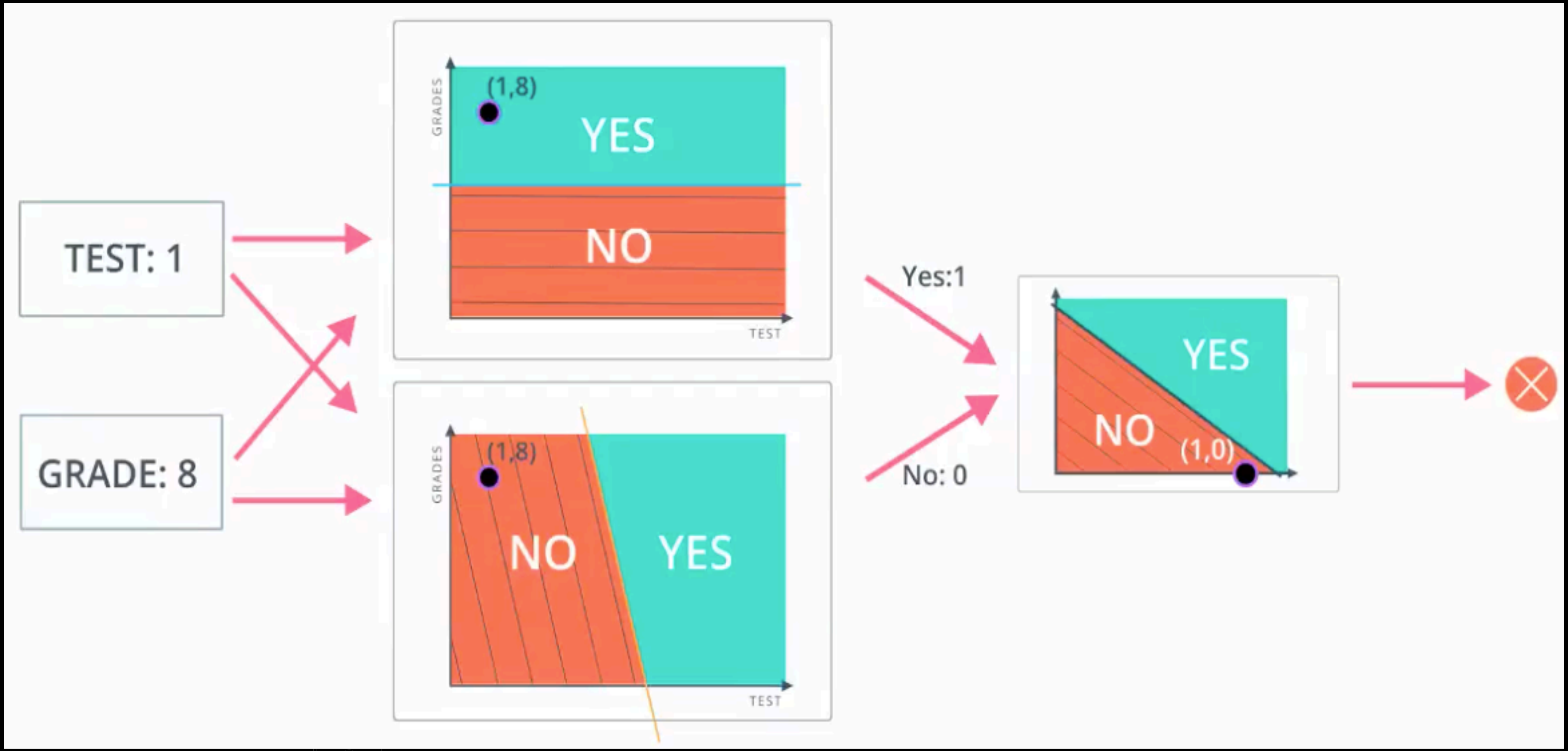

Source: Udacity

# Neural Networks (4): 5个节点



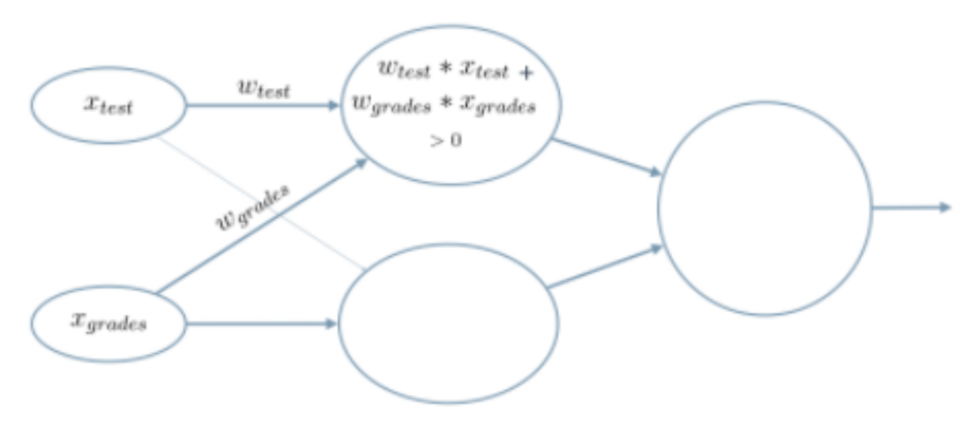

Source: Udacity

# Neural Networks (5)



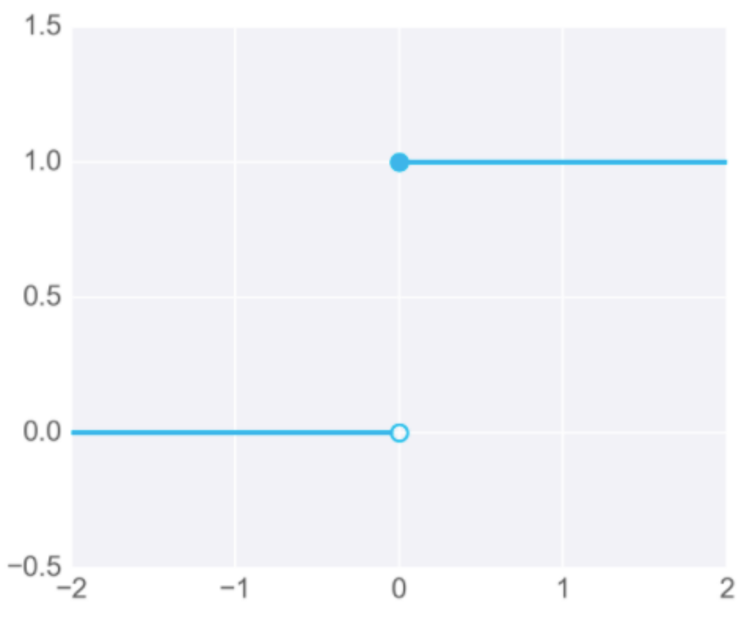

Source: Udacity

# Neural Networks (6) Weights & Activation




Source: Udacity

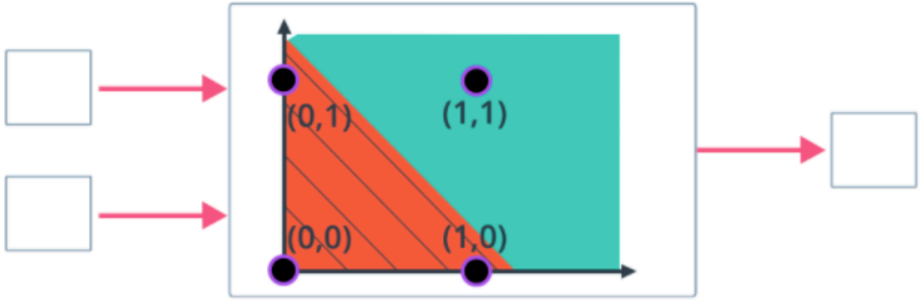
$$h_\Theta(x) \approx \begin{bmatrix} 1 \\ 0 \\ 0 \\ 0 \end{bmatrix}$$




Heaviside Step Function

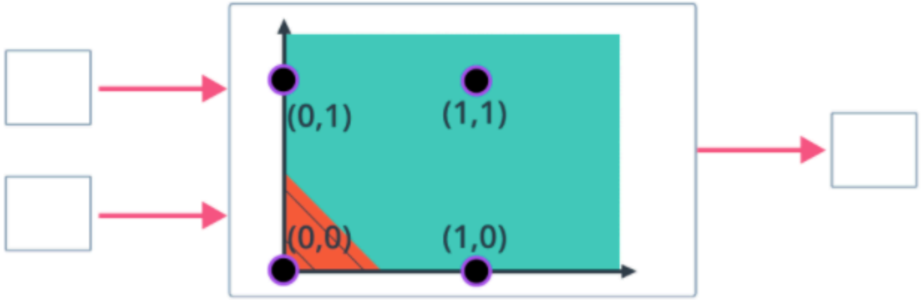
# And / Or Perceptron 感知器

## AND




| IN | IN | OUT |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 0 |
| 0 | 1 | 0 |
| 0 | 0 | 0 |

## OR

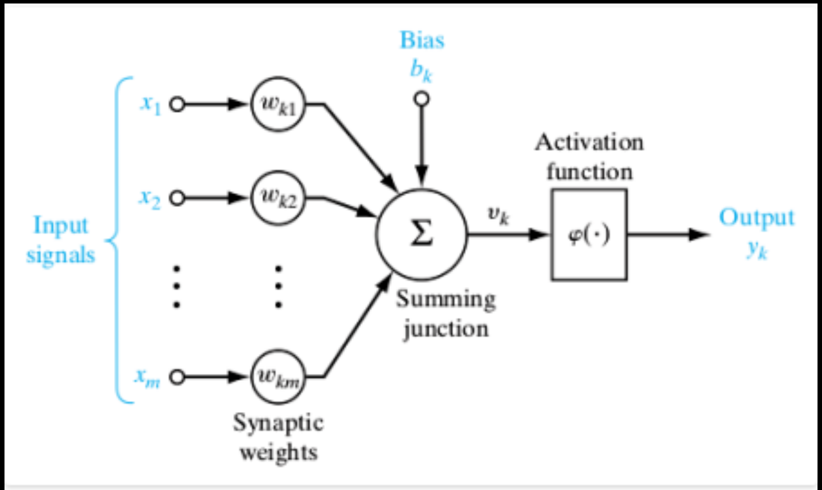



| IN | IN | OUT |
|---|---|---|
| 1 | 1 | 1 |
| 1 | 0 | 1 |
| 0 | 1 | 1 |
| 0 | 0 | 0 |

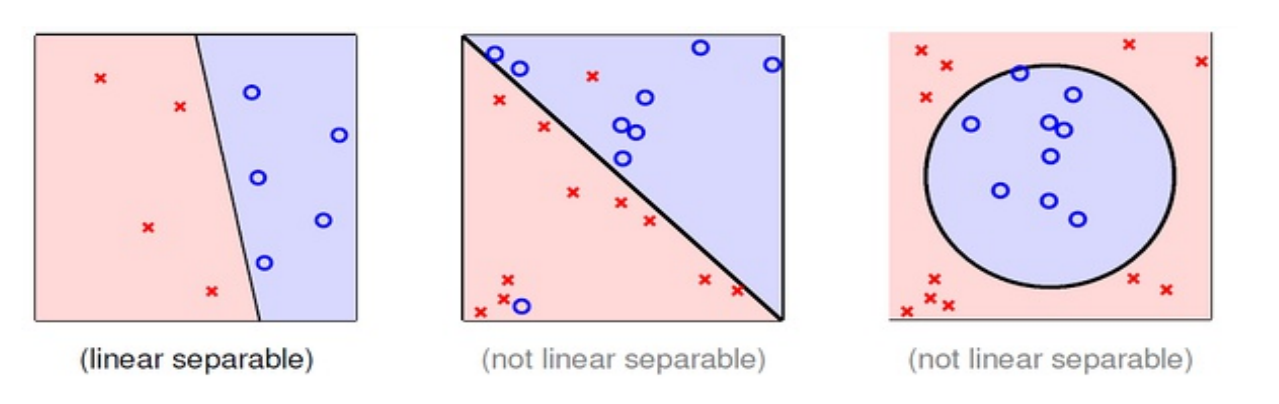
Source: Udacity

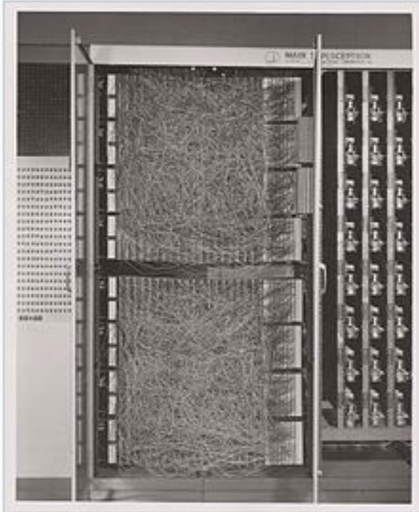
# Perceptron 感知器

- Warren McCulloch & Walter Pitts proposed the math model of artificial neural (1943)
- Donald Hebb proposed Hebb learning rule (1949)




- 单层的感知器仅能处理线性可分(Linear Separable )的情况。



The Mark I Perceptron machine was the first implementation of the perceptron algorithm. The machine was connected to a camera that used 20×20 cadmium sulfide photocells to produce a 400-pixel image. The main visible feature is a patchboard that allowed experimentation with different combinations of input features. To the right of that are arrays of potentiometers that implemented the adaptive weights.[2]:213

- 1957年，在美国海军研究办公室的资助下，Cornell 航空实验室的Frank Rosenblatt发明了感知器。

Source: Yaser, Malik, Hsuan-Tien Lin

# Perceptron 感知器

感知器是一种具有学习能力的分类器算法，是一个将输入的实数矢量$x$映射为输出值$h(x)$的函数。

$$h(x) = \begin{cases} 1 & if\ w^T \cdot x > 0 \\ -1 & otherwise \end{cases}$$

$*$ 类似与线性回归，为了方便数学表达，我们规定$x_0 = 1$。

$$\begin{aligned} h(\mathbf{x}) &= \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) - \text{threshold}\right) \\ &= \text{sign}\left(\left(\sum_{i=1}^{d} w_i x_i\right) + \underbrace{(-\text{threshold})}_{w_0} \cdot \underbrace{(+1)}_{x_0}\right) \\ &= \text{sign}\left(\sum_{i=0}^{d} w_i x_i\right) \\ &= \text{sign}\left(\mathbf{w}^T \mathbf{x}\right) \end{aligned}$$

Source: Yaser, Malik, Hsuan-Tien Lin

**Steps:**

1. 初始化权重与偏置（Bias）（0或小随机数）
2. 感知器使用以下简单的Iteration方法，对权重进行更新：

a) $h_j(t) = sign(w(t) \cdot x_j)$ 在第$t$次迭代中，计算每个训练样本的预测值

b) $w_{t+1} = w_t + y_t x_t$ 对于任何一个被错误分离的样本，更新权重

# 感知器基本思想

- 目标：对于训练样本（$x, y$），预测值与实际值一致, 即$yh_t(x) > 0$
- 而对于被错误分类的样本而言：

$$yh_t(x) < 0$$

- 希望每一步（t）迭代：$yh_{t+1}(x) \geq yh_t(x)$ 而 $h_t(x) = sign(w(t) \cdot x)$
- $w_{t+1} = w_t + y_t x_t$

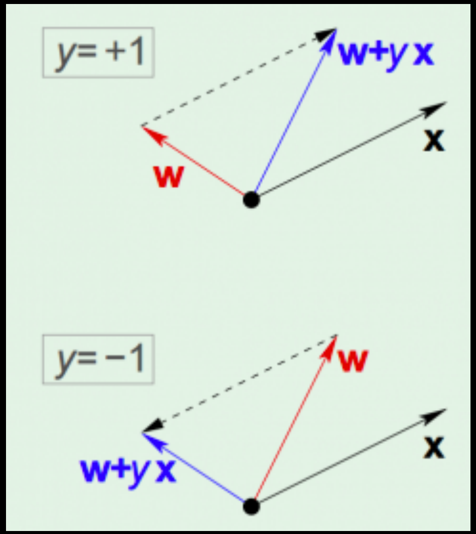
类似梯度下降

- $yh_{t+1}(x) = yw_{t+1}x$

$$= y(w_t + yx)x$$

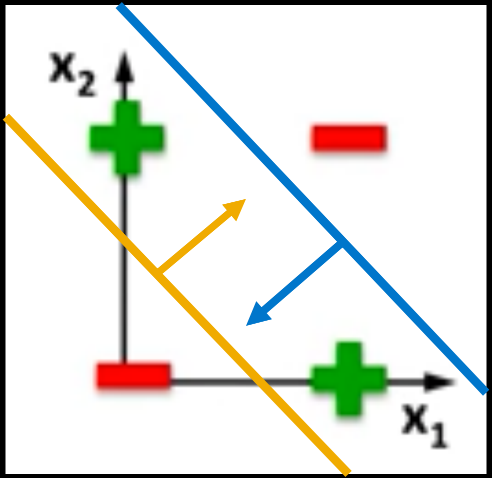
$$= yw_t x + y^2 x^2 \geq yw_t x$$

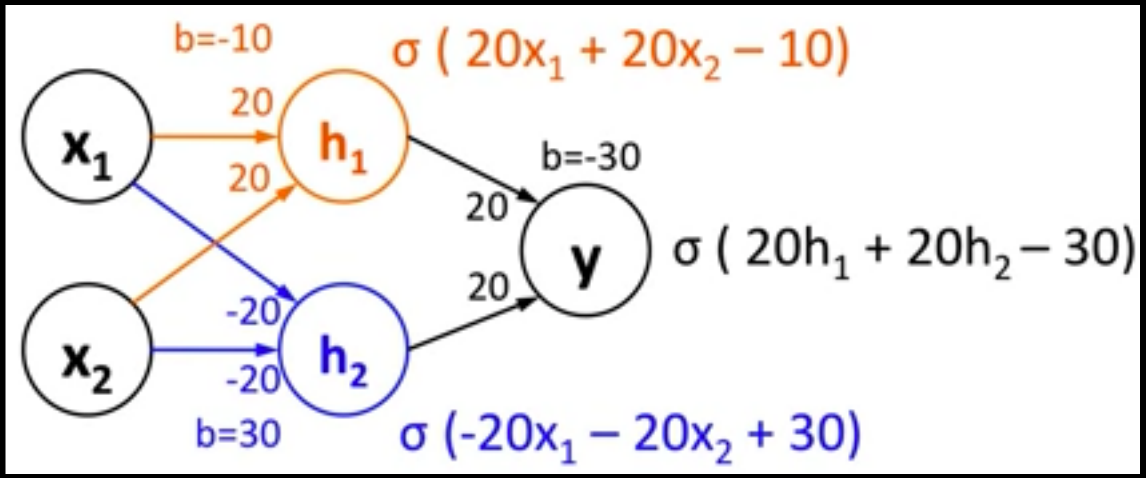
感知器直观的几何解释




Source: Yaser, Malik, Hsuan-Tien Lin

# XOR can be handled by MLP

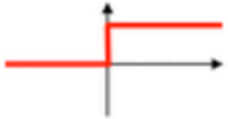
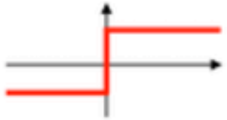
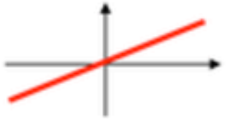
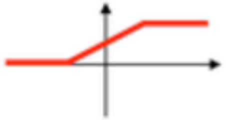
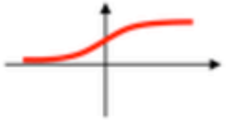



| | | |
|---|---|---|
| $\sigma(20*0 + 20*0 - 10) \approx 0$ | $\sigma(-20*0 - 20*0 + 30) \approx 1$ | $\sigma(20*0 + 20*1 - 30) \approx 0$ |
| $\sigma(20*1 + 20*1 - 10) \approx 1$ | $\sigma(-20*1 - 20*1 + 30) \approx 0$ | $\sigma(20*1 + 20*0 - 30) \approx 0$ |
| $\sigma(20*0 + 20*1 - 10) \approx 1$ | $\sigma(-20*0 - 20*1 + 30) \approx 1$ | $\sigma(20*1 + 20*1 - 30) \approx 1$ |
| $\sigma(20*1 + 20*0 - 10) \approx 1$ | $\sigma(-20*1 - 20*0 + 30) \approx 1$ | $\sigma(20*1 + 20*1 - 30) \approx 1$ |

Victor Lavrenko

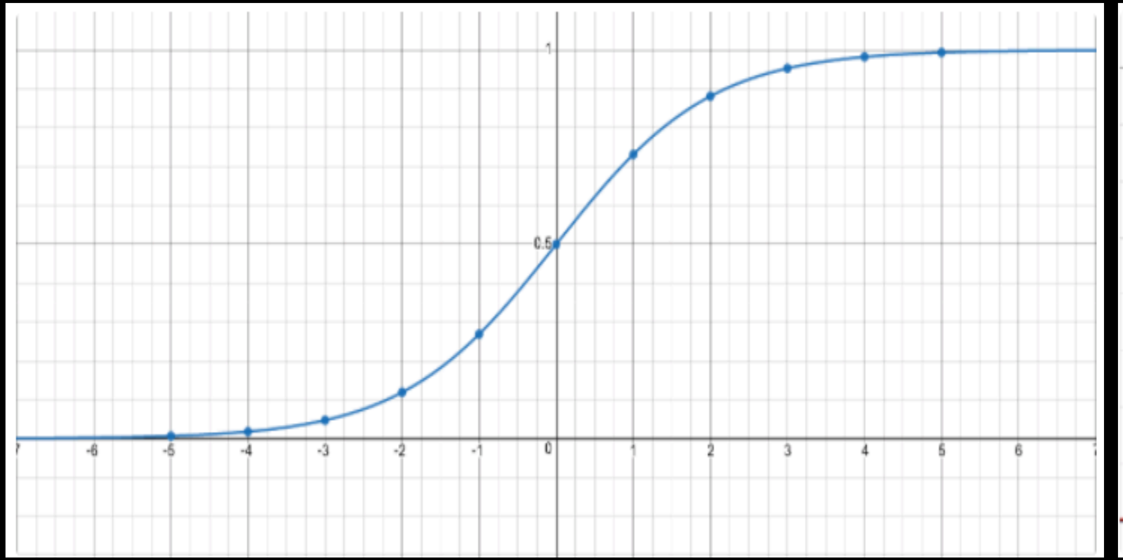
# Activation Function

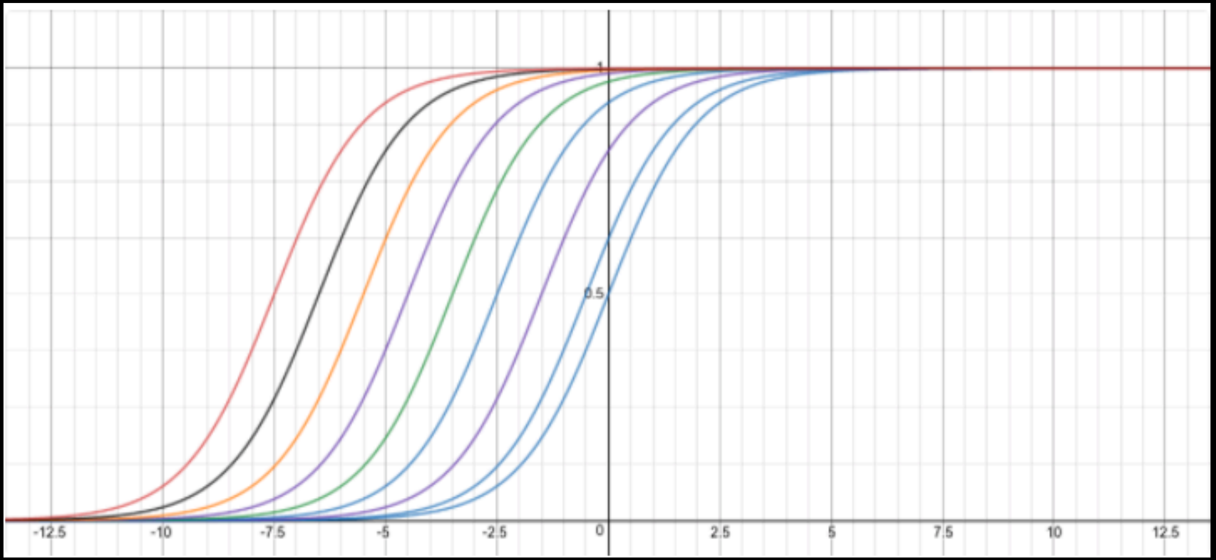
| Activation function | Equation | Example | 1D Graph |
|---|---|---|---|
| Unit step (Heaviside) | $\phi(z) = \begin{cases} 0, & z < 0, \\ 0.5, & z = 0, \\ 1, & z > 0, \end{cases}$ | Perceptron variant |  |
| Sign (Signum) | $\phi(z) = \begin{cases} -1, & z < 0, \\ 0, & z = 0, \\ 1, & z > 0, \end{cases}$ | Perceptron variant |  |
| Linear | $\phi(z) = z$ | Adaline, linear regression |  |
| Piece-wise linear | $\phi(z) = \begin{cases} 1, & z \geq \frac{1}{2}, \\ z + \frac{1}{2}, & -\frac{1}{2} < z < \frac{1}{2}, \\ 0, & z \leq -\frac{1}{2}, \end{cases}$ | Support vector machine |  |
| Logistic (sigmoid) | $\phi(z) = \frac{1}{1 + e^{-z}}$ | Logistic regression, Multi-layer NN |  |
| Hyperbolic tangent | $\phi(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$ | Multi-layer NN |  |

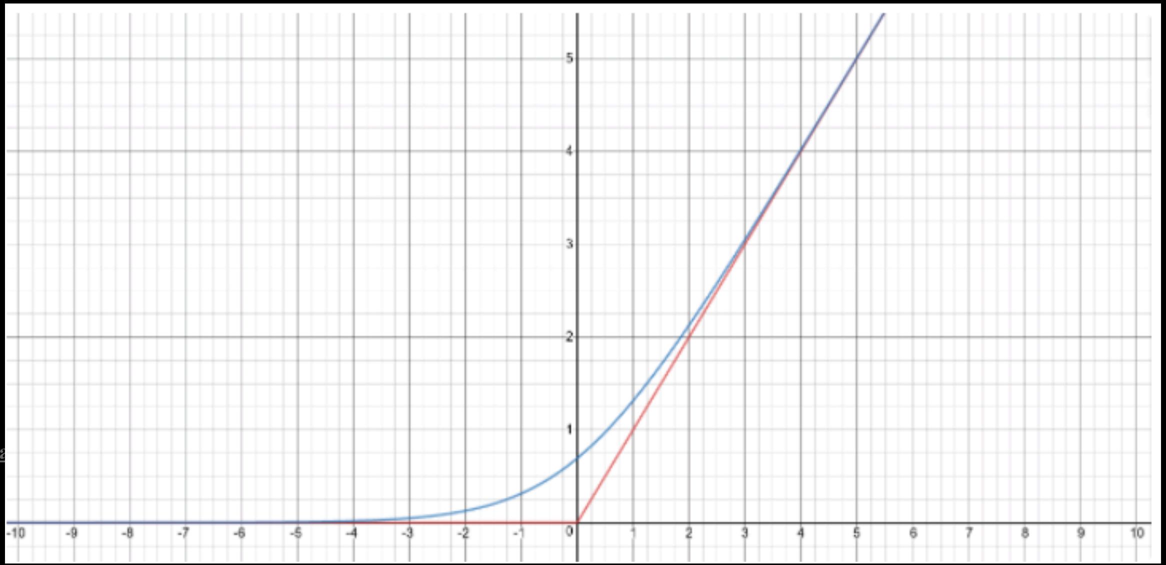
Sebastian Raschka

復旦大學

# From Sigmoid to ReLU (Rectified Linear Unit)



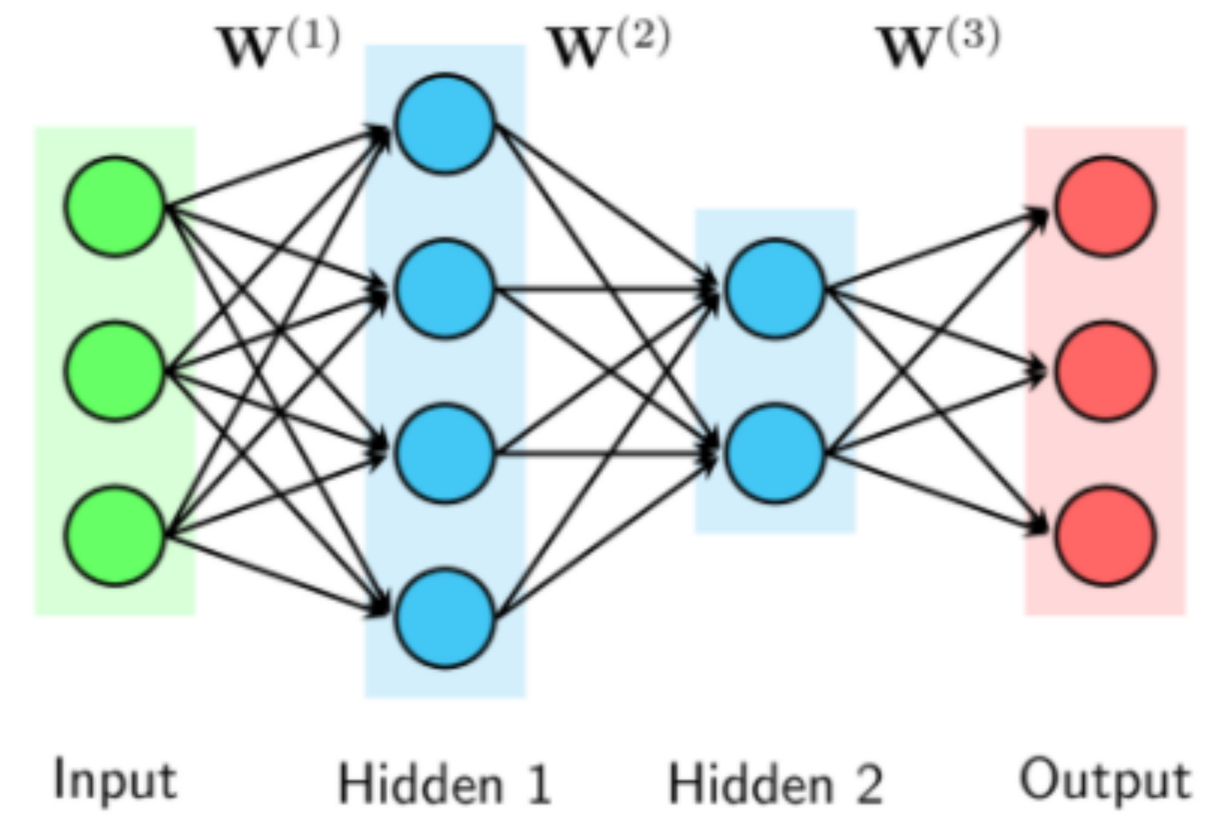

SSU， Stepped Sigmoid Unit with offset 0.5,1.5,2.5,3.5,…

$$\sum_{n=1}^{\infty} \sigma(x + 0.5 - n) = \log(1 + e^{x})$$

Softplus Function

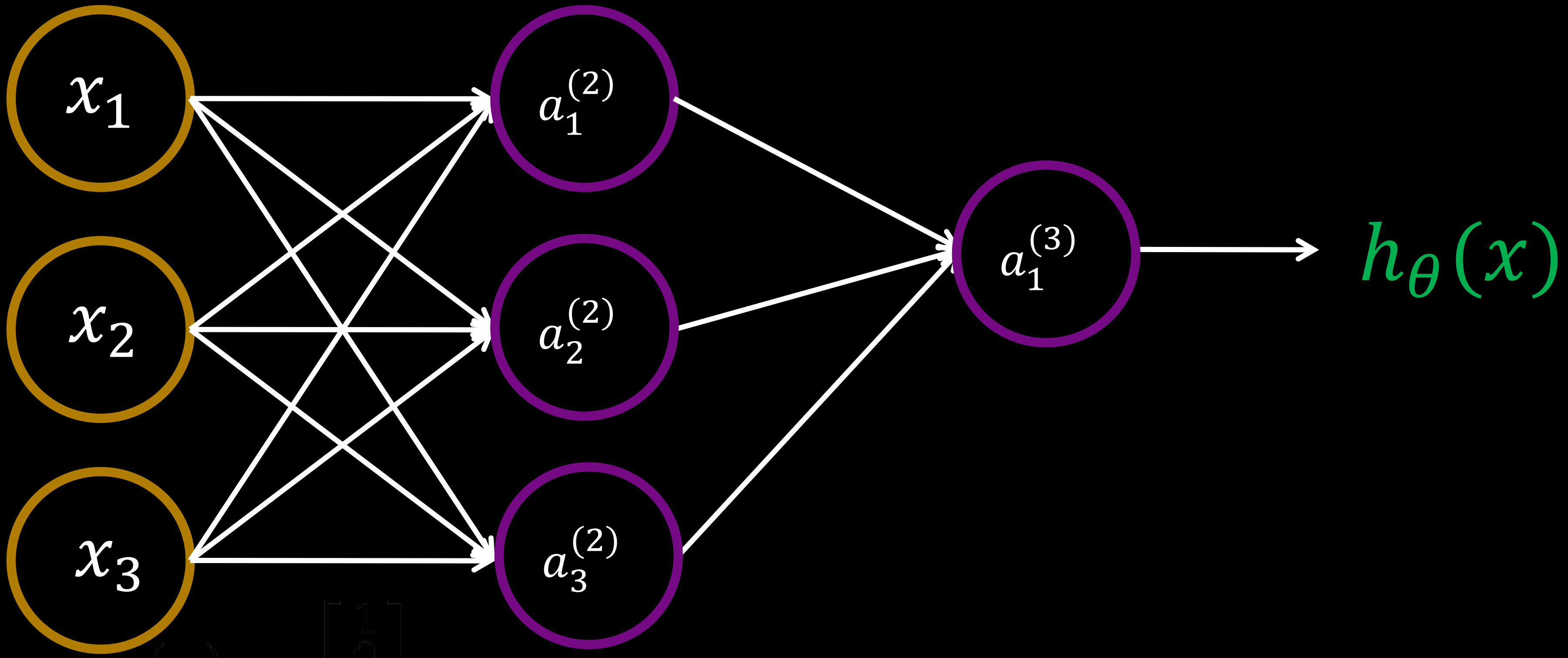
Sebastia

# 多层感知器

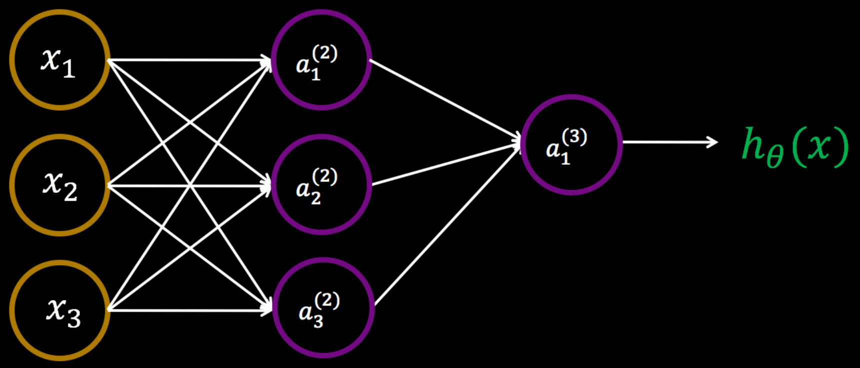



$h_\Theta(x) \approx$

# Neural Network



# Neural Network: Feeding Forward




- $a_i^{(j)}$ 第$j$层第$i$个神经元的Activation
- $\Theta^{(j)}$ 控制着从第$j$层到第 $j+1$层映射的参数矩阵

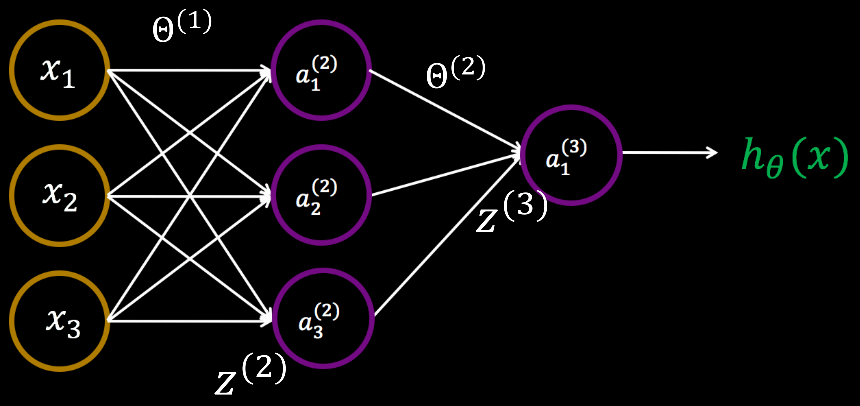
$$a_1^{(2)} = \sigma(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = \sigma(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = \sigma(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\theta(x) = a_1^{(3)} = \sigma(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

# Neural Network: Vectorization




$$a_1^{(2)} = \sigma(\Theta_{10}^{(1)}x_0 + \Theta_{11}^{(1)}x_1 + \Theta_{12}^{(1)}x_2 + \Theta_{13}^{(1)}x_3)$$

$$a_2^{(2)} = \sigma(\Theta_{20}^{(1)}x_0 + \Theta_{21}^{(1)}x_1 + \Theta_{22}^{(1)}x_2 + \Theta_{23}^{(1)}x_3)$$

$$a_3^{(2)} = \sigma(\Theta_{30}^{(1)}x_0 + \Theta_{31}^{(1)}x_1 + \Theta_{32}^{(1)}x_2 + \Theta_{33}^{(1)}x_3)$$

$$h_\theta(x) = a_1^{(3)} = \sigma(\Theta_{10}^{(2)}a_0^{(2)} + \Theta_{11}^{(2)}a_1^{(2)} + \Theta_{12}^{(2)}a_2^{(2)} + \Theta_{13}^{(2)}a_3^{(2)})$$

$$x = \begin{bmatrix} x_0 \\ x_1 \\ x_2 \\ x_3 \end{bmatrix} \qquad z^{(2)} = \begin{bmatrix} z_0^{(2)} \\ z_0^{(2)} \\ z_0^{(2)} \\ z_0^{(2)} \end{bmatrix}$$

$$z^{(2)} = \Theta^{(1)}x$$

$$a^{(2)} = \sigma(z^{(2)})$$

$$add\ a_0^{(2)} = 1$$

$$z^{(3)} = \Theta^{(2)}a^{(2)}$$

$$h_\theta(x) = a_1^{(3)} = \sigma(z^{(3)})$$

# Recap：Chain Rule

- $f(x,y,z) = (x+y)z$　　可以写成$f = qz$，其中$q = x + y$

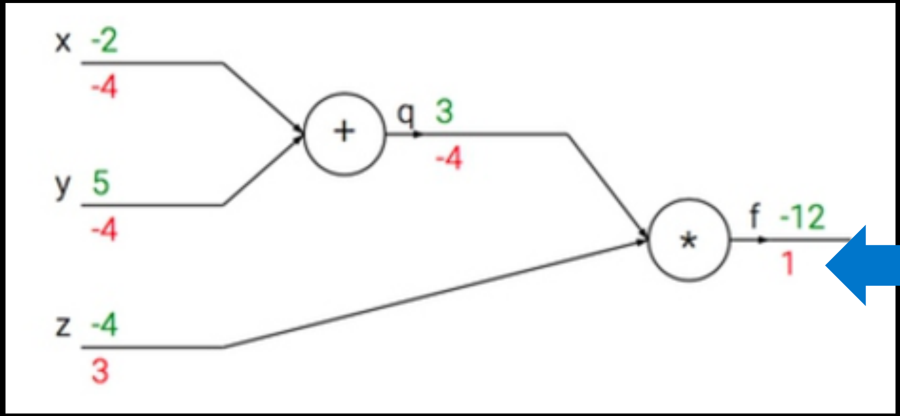
$$q = x + y. \qquad \frac{dq}{dx} = 1, \frac{dq}{dy} = 1$$

$$f = qz. \qquad \frac{df}{dq} = z, \frac{df}{dz} = q$$

$$\frac{df}{dx} = \frac{df}{dq}\frac{dq}{dx} = z = -4$$

$$\frac{df}{dy} = \frac{df}{dq}\frac{dq}{dy} = z = -4$$

$$\frac{df}{dz} = q = x + y = 3$$




FeiFei LI & Andrej Karpathy & Justin Johnson

$$\frac{df}{dq} = z = -4$$

$$\frac{df}{df} = 1$$

# Thank you!

Contact information:

邬学宁 (i025497)
Chief Data Scientist, SAP Silicon Valley Innovation Center
Address: No. 1001, Chenghui Road, Shanghai, 201023
Phone number: +8621-6108 5287
Email: x.wu@sap.com