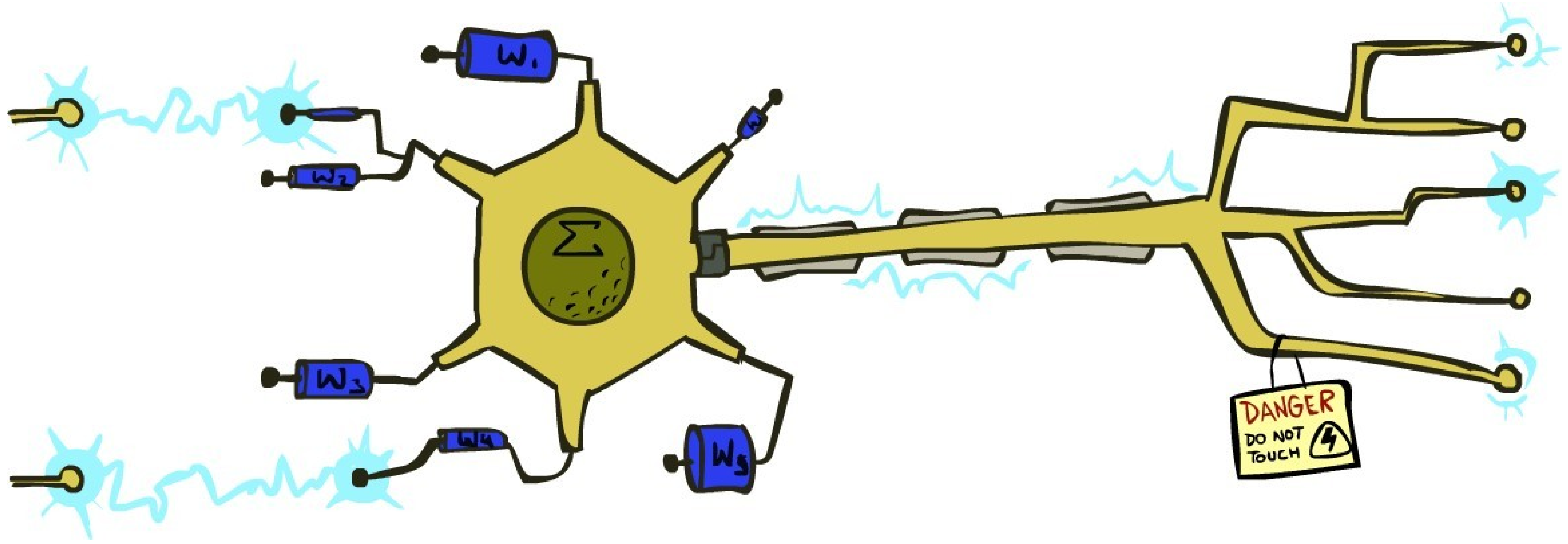
第十八章 样例学习

Linear Regression and Perceptrons



Feature Vectors

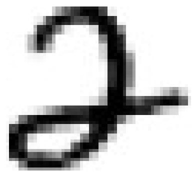
x $\quad$ $f(x)$ $\quad$ y

```
Hello,

Do you want free printr
cartriges?   Why pay
more when you can get
them ABSOLUTELY FREE!
             Just
```

```
# free        : 2
YOUR_NAME     :
0
MISSPELLED    :
2
FROM_FRIEND   : 0
...
```

SPAM
or
+



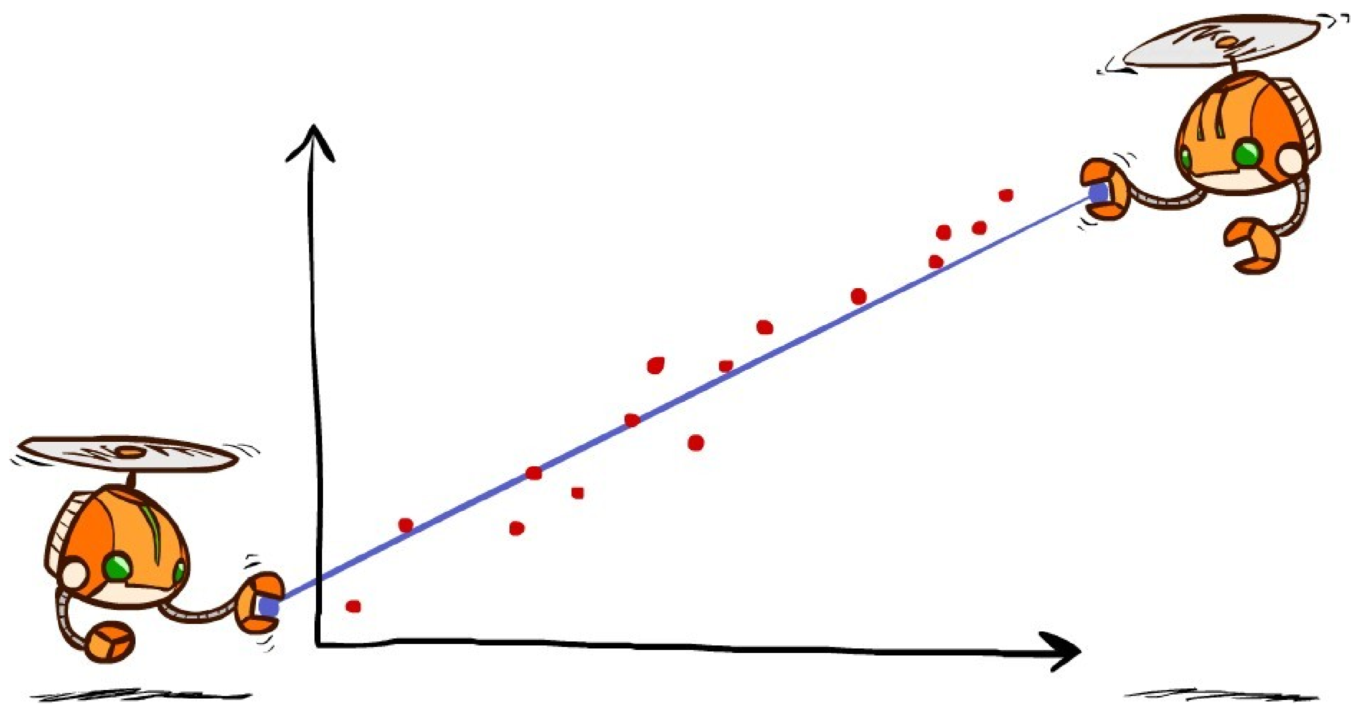
```
PIXEL-7,12   : 1
PIXEL-7,13   : 0
...
NUM_LOOPS    : 1
...
```

“2”

Outline

- 18.6 线性回归模型
- 18.7 神经网络
  - M-P 模型、感知机与多层感知机
  - 神经网络中的学习 (BP 算法)

线性回归

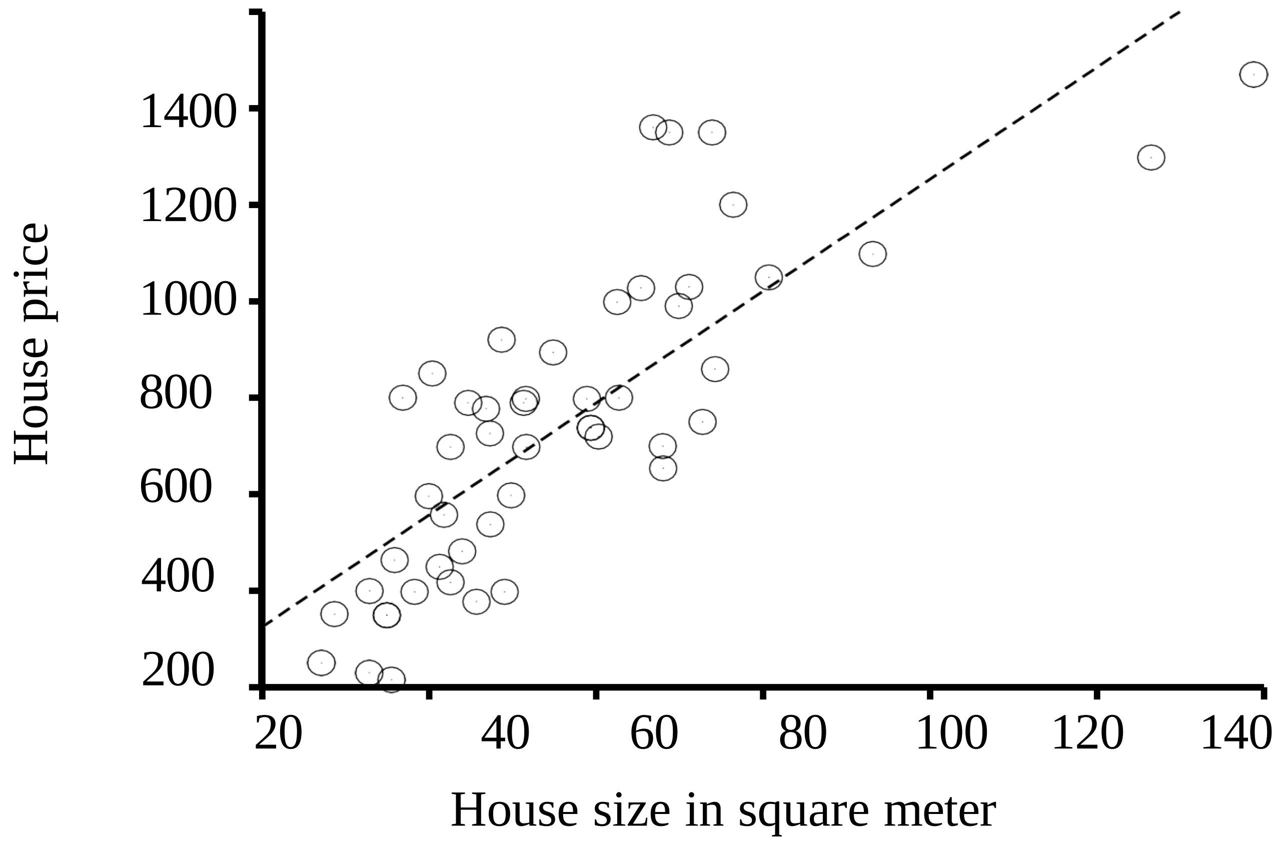


Hypothesis family: Linear functions

线性回归

(x, y=f(x)), x: house size

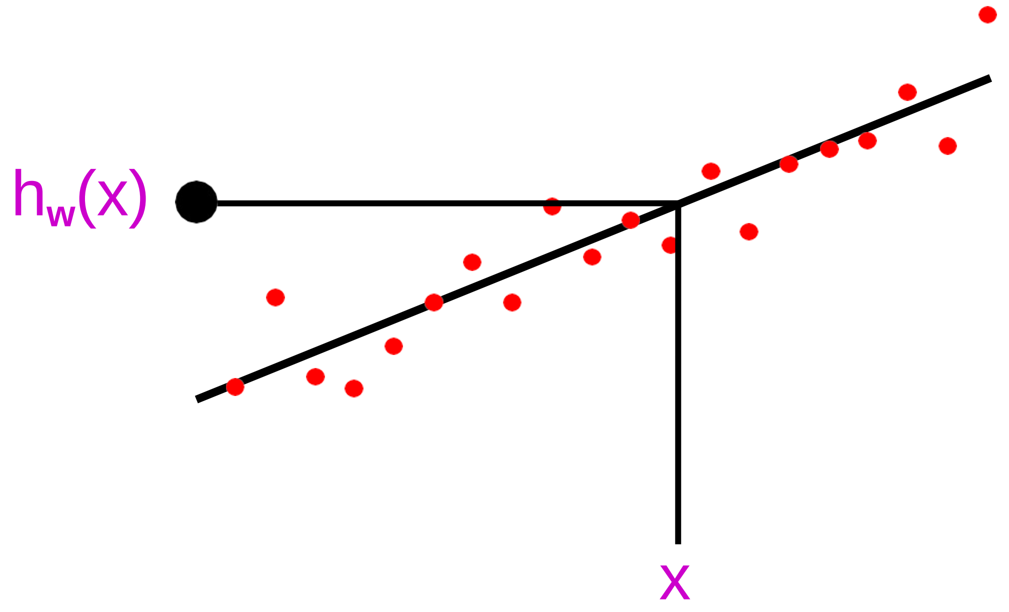
y: house price

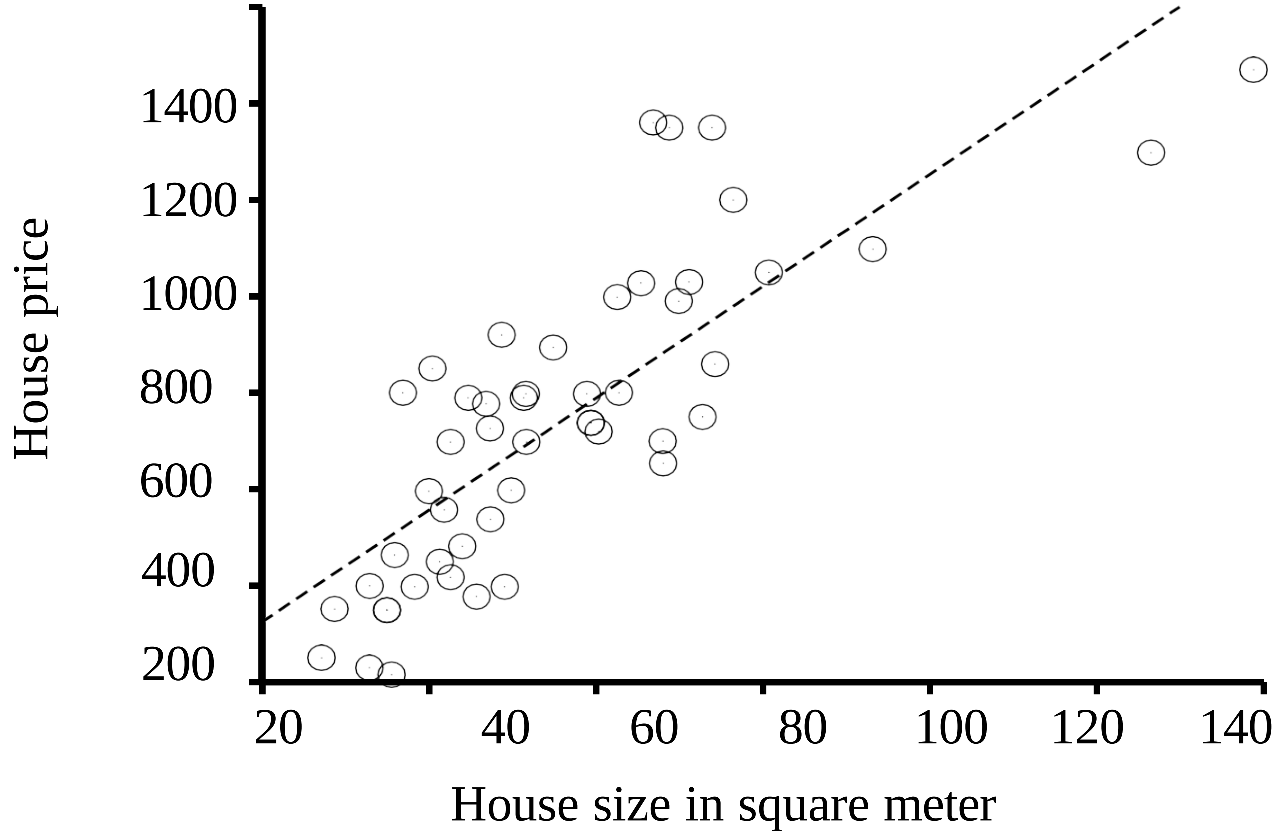


Prediction: $h_w(x) = w_0 + w_1 x$

线性回归

线性回归 = 拟合直线 / 超平面



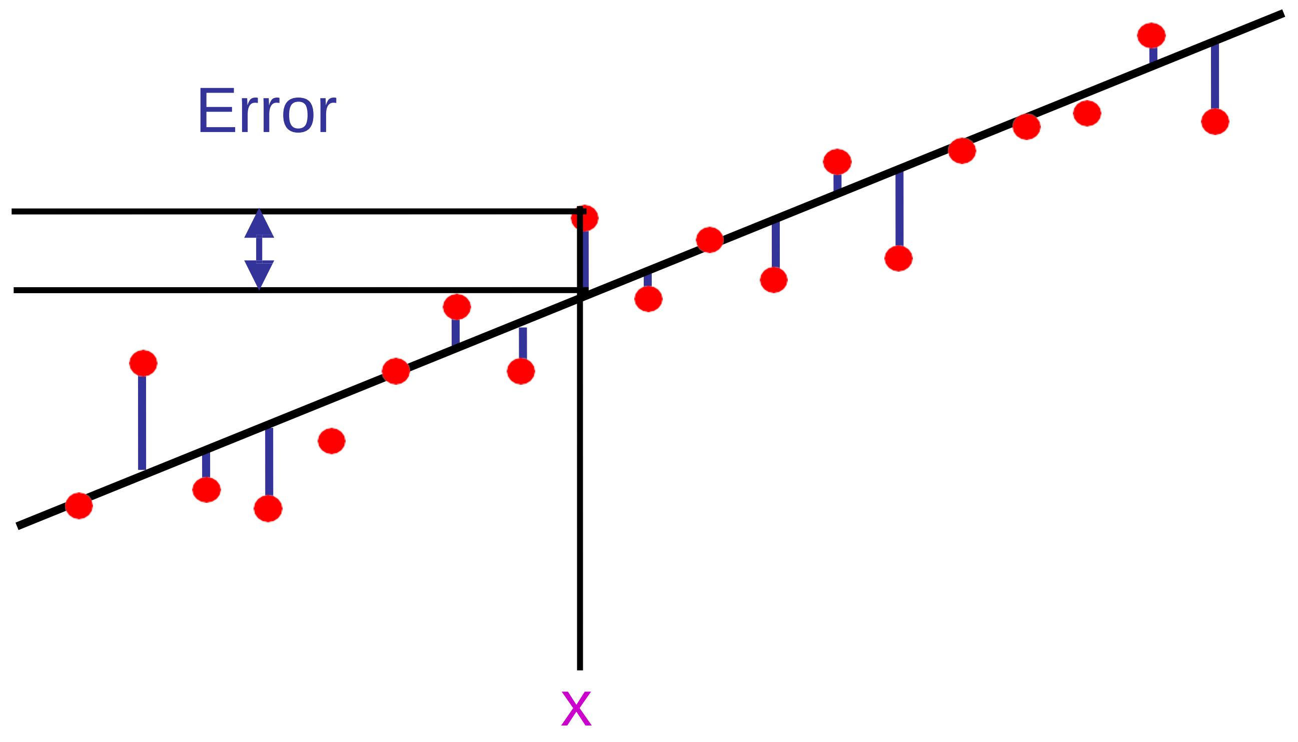


Prediction: $h_w(x) = w_0 + w_1x$

- Define loss function
- Minimize loss function to find w*

预测误差

单个样本的误差 : $y - h_w(x)$



损失函数： 估量模型的预测值 $h_w(x)$ 与真实值 y 的不一致程度

L2 损失： 所有样例上的均方误差

$$\text{Loss} = \sum_j (y_j - h_w(x_j))^2 = \sum_j (y_j - (w_0 + w_1 x_j))^2$$

- Define loss function
- Minimize loss function to find w^*

最小二乘法：最小化平方误差

- L2 损失函数：所有样例上的均方误差

  $$Loss = \sum_j (y_j - h_w(x_j))^2 = \sum_j (y_j - (w_0 + w_1 x_j))^2$$

- 计算 w* ，使得损失最小化（求导为零）
  - $\partial Loss / \partial w_0 = -2 \sum_j (y_j - (w_0 + w_1 x_j)) = 0$
  - $\partial Loss / \partial w_1 = -2 \sum_j (y_j - (w_0 + w_1 x_j))\, x_j = 0$

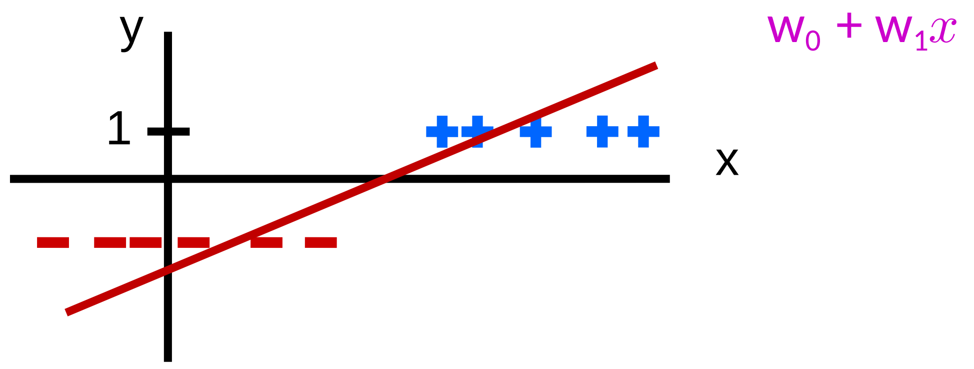
- 矩阵求解最小二乘法
  - 数据矩阵 **X**（每行一个样本）；标签 **y**
  - $w^* = (\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}$

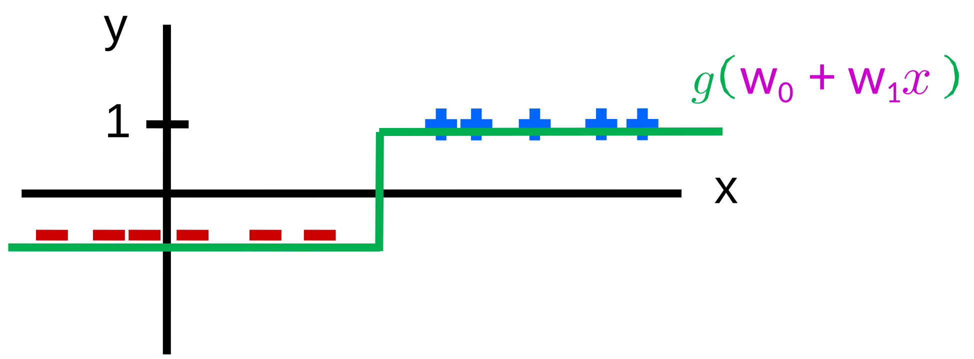
除了线性模型之外，这种定义的最小损失方程的求解方式，经常是没有封闭解的。

优化方法：梯度下降

回归 vs 分类

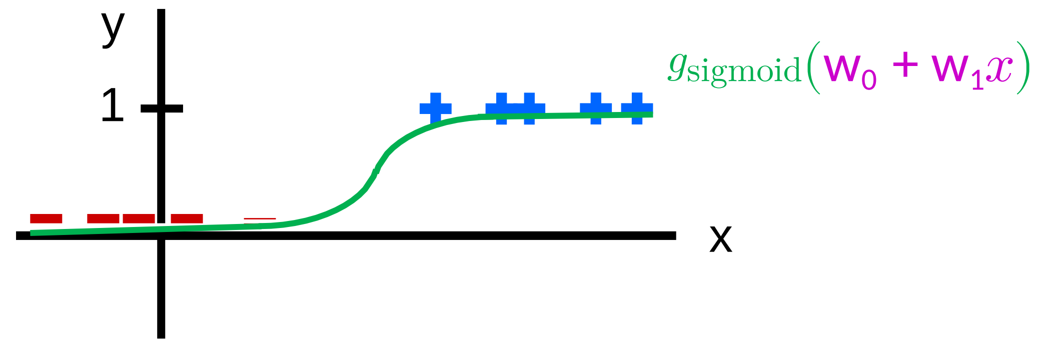
- 线性回归
  - $h_w(x) = w_0 + w_1 x$
- 线性分类
  - 输出离散数值
  - $h_w(x) = g(w_0 + w_1 x) = 1, \quad \text{if } w_0 + w_1 x \geq 0$
  - $h_w(x) = g(w_0 + w_1 x) = -1, \quad \text{if } w_0 + w_1 x < 0$
  - 阈值激活函数g





- 逻辑回归 Logistic Regression

$$g_{\text{sigmoid}}(w_0 + w_1 x) = \frac{1}{1 + e^{-(w_0 + w_1 x)}}$$

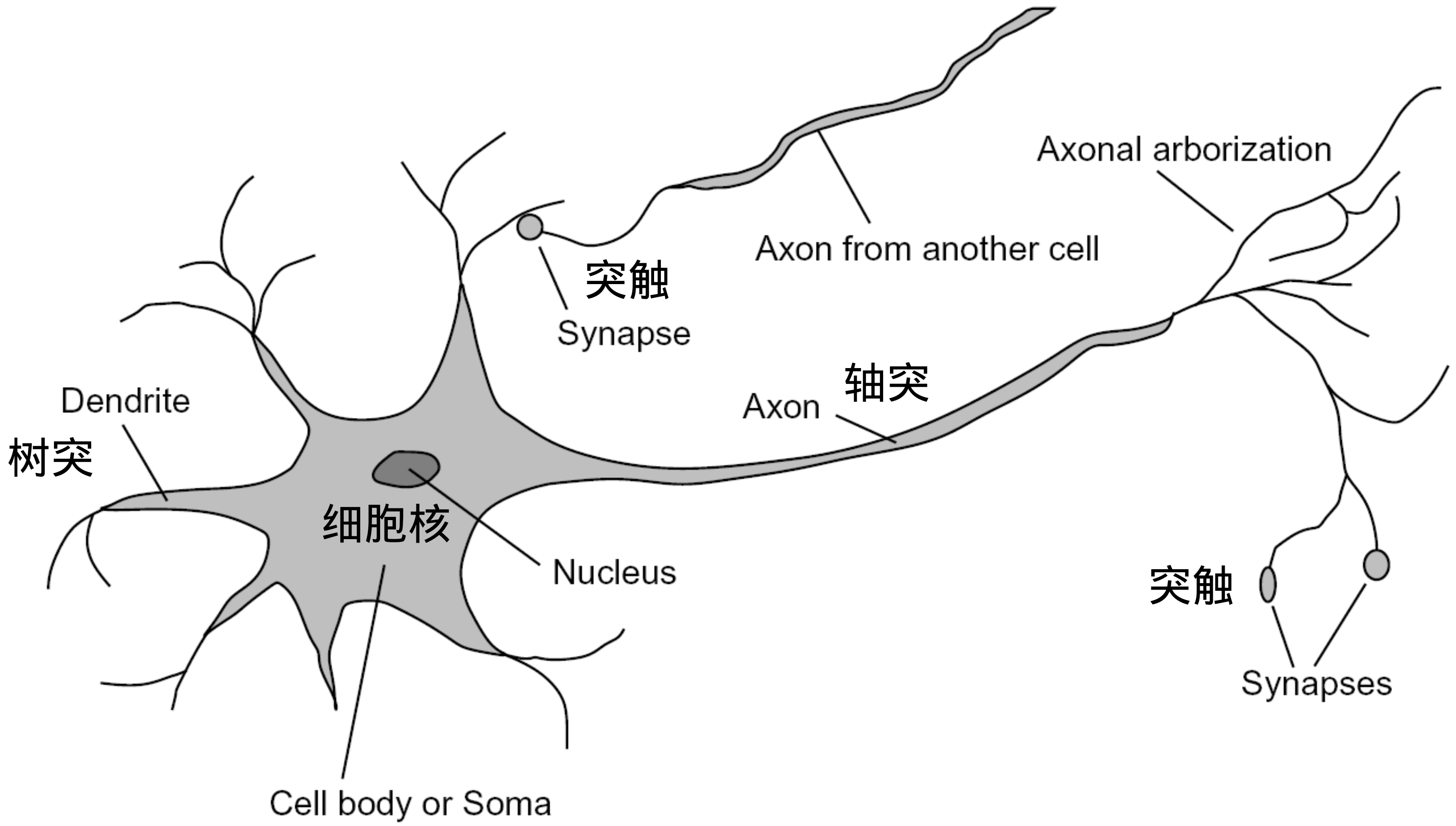


Outline

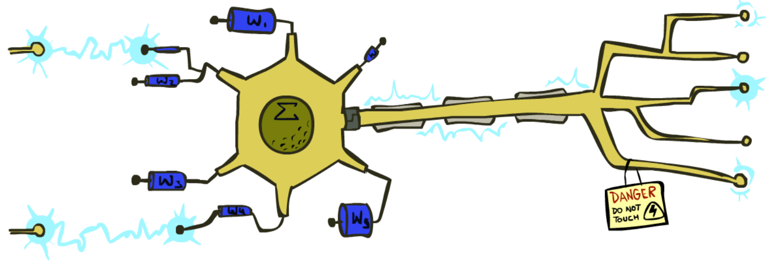
- 18.6 线性回归模型
- 18.7 神经网络
  - M-P 模型、感知机、多层感知机
  - 神经网络中的学习 (BP 算法)

Some (Simplified) Biology

- 灵感：人类神经元



神经网络模型

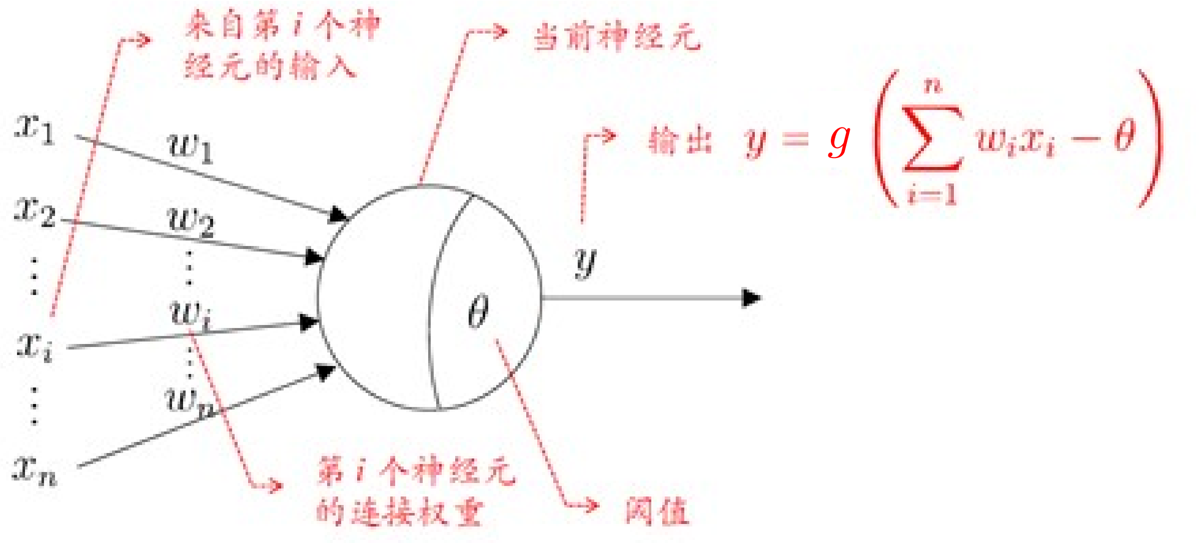


M-P 神经元模型 [McCulloch and Pitts，1943]

线性分类

$y=g(w_0+w_1x_1+w_2x_2+\dots)$

- 输入：来自其他 *n* 个神经元传递过来的输入信号（特征值）
- 处理：输入信号通过带权重的连接进行传递，神经元接受到总输入值将与神经元的阈值进行比较
- 输出：通过激活函数g的处理得到输出

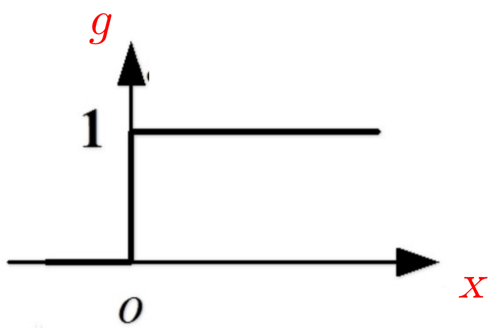


来自第 i 个神经元的输入
当前神经元
输出 $y = g\left(\sum_{i=1}^{n} w_i x_i - \theta\right)$
x_1
x_2
x_i
x_n
w_1
w_2
w_i
w_n
θ
y
第 i 个神经元的连接权重
阈值

M-P 神经元模型

阈值激活函数：阶跃函数

g(x)=sign(x)



MP 模型是一个计算模型。给定的参数 w、 θ

Weights

Dot product $w \cdot f$ *positive means the positive class (spam)*

$w \quad \cdot \quad f(x_1)$

```
# free      : 4
YOUR_NAME   :-1
MISSPELLED  : 1
FROM_FRIEND :-3
..
```

```
# free      : 2
YOUR_NAME   : 0
MISSPELLED  : 2
FROM_FRIEND : 0
..
```

$w \quad \cdot \quad f(x_2)$

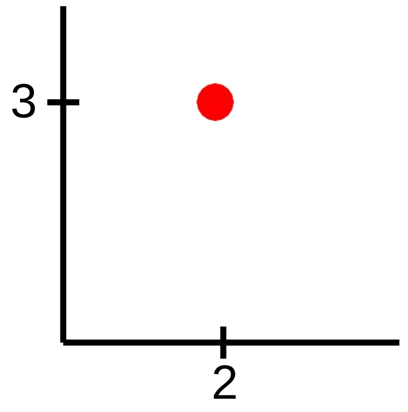
```
# free      : 4
YOUR_NAME   :-1
MISSPELLED  : 1
FROM_FRIEND :-3
..
```

```
# free      : 0
YOUR_NAME   : 1
MISSPELLED  : 1
FROM_FRIEND : 1
..
```

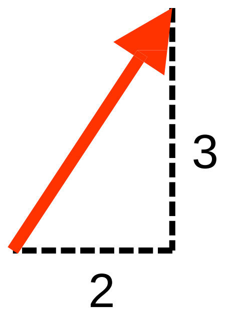
Do these weights make sense for spam classification?

Review: Vectors

- A tuple like (2,3) can be interpreted two different ways:



A **point** on a coordinate
grid

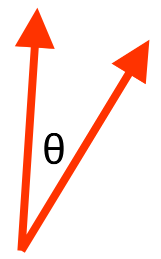


A **vector** in space. Notice we
are
not on a coordinate grid.

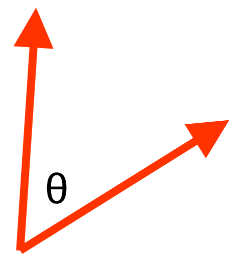
- A tuple with more elements like (2, 7, -3, 6) is a point or vector in higher-dimensional space (hard to visualize)

Review: Vectors

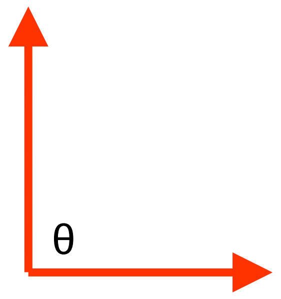
- Definition of dot product:
  - a • b = |a| |b| cos(θ)
  - θ is the angle between the vectors a and b
- Consequences of this definition:
  - Vectors closer together
    = "similar" vectors
    = smaller angle θ between vectors
    = larger (more positive) dot product
  - If θ < 90°, then dot product is positive
  - If θ = 90°, then dot product is zero
  - If θ > 90°, then dot product is negative



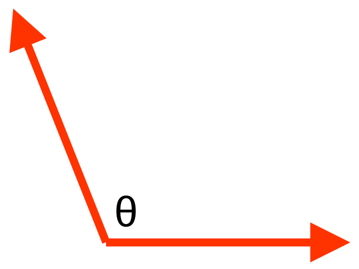
a · b large, positive



a · b small, positive



a · b zero

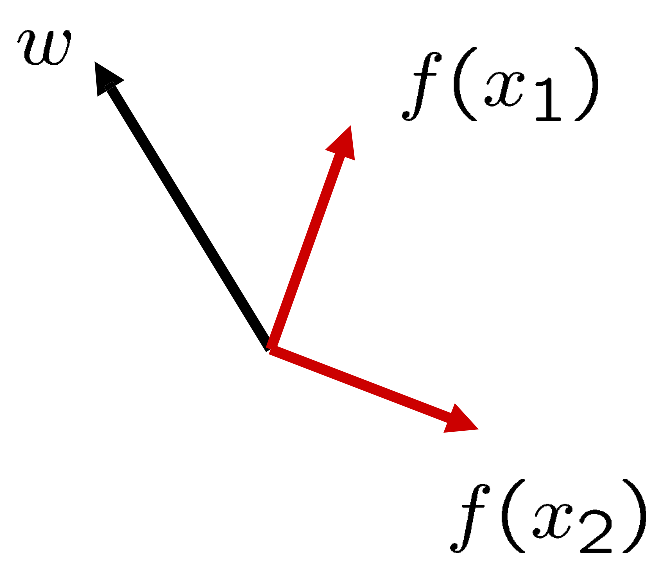


a · b negative

Weights

- Binary case: compare features to a weight vector
- Learning: figure out the weight vector from examples

```
# free      : 4
YOUR_NAME   :-1
MISSPELLED  : 1
FROM_FRIEND :-3
...
```

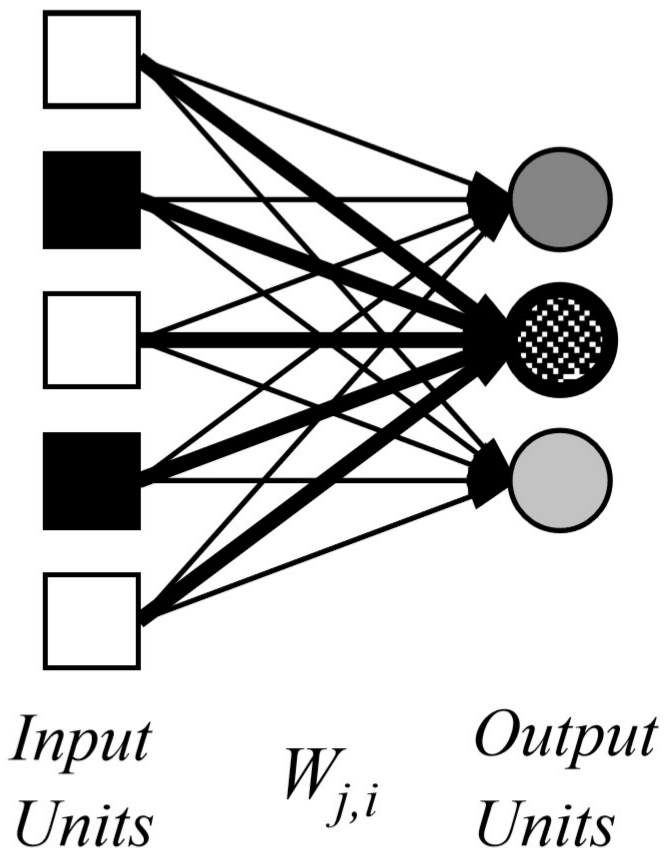


```
# free      : 2
YOUR_NAME   : 0
MISSPELLED  : 2
FROM_FRIEND : 0
...
```

```
# free      : 0
YOUR_NAME   : 1
MISSPELLED  : 1
FROM_FRIEND : 1
...
```

Dot product $w \cdot f$ positive means the positive class

感知机（ Perceptron ）



一个感知机网络

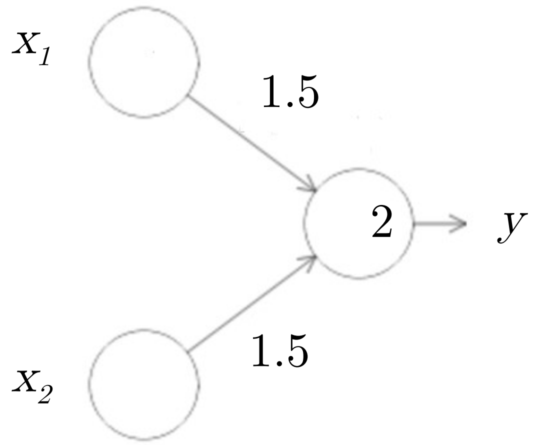
- 1957 年 Rosebblatt 提出感知机，最早的人工神经网络模型。

- 感知机输入层接受外界输入信号传递给输出层，输出层是多个 M-P 神经元（阈值逻辑单元）

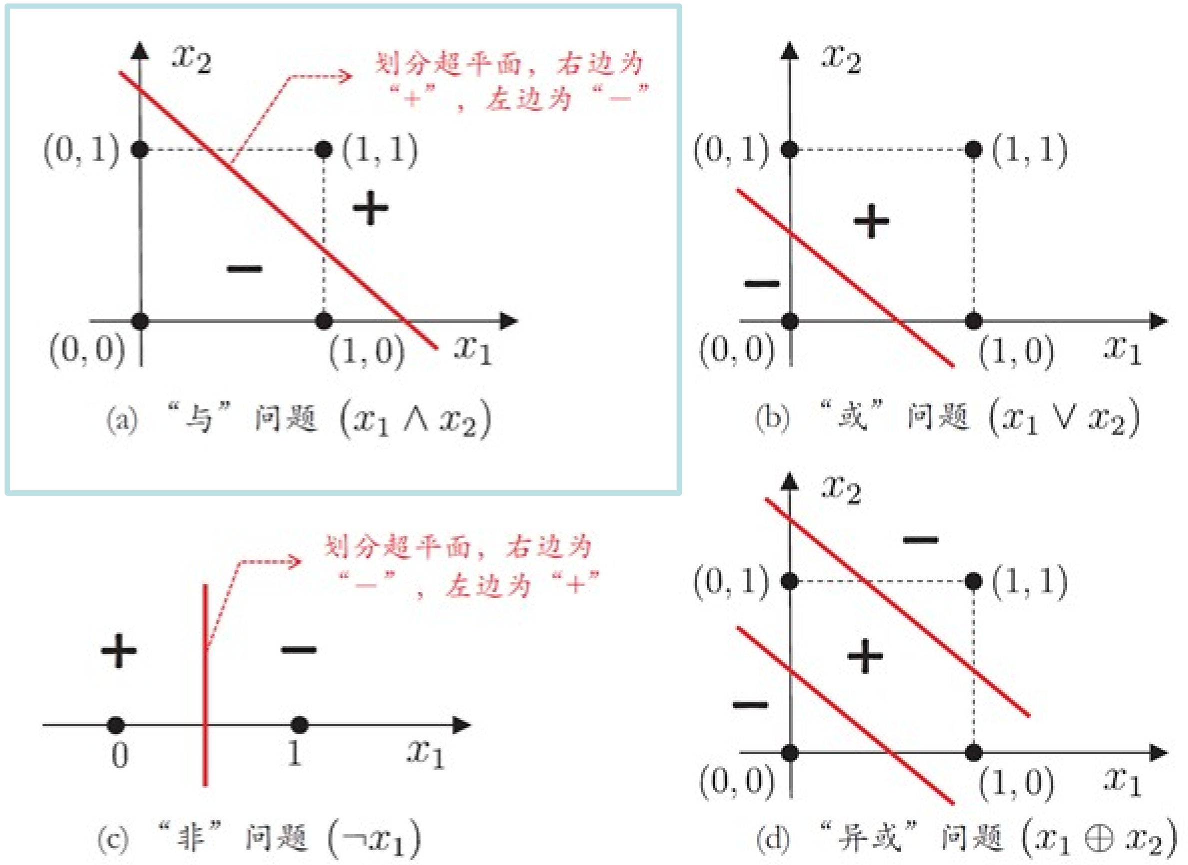
  - 可学习：根据训练数据来调整神经元之间的“连接权”以及每个功能神经元的“阈值”

感知机的表示能力

- 一个采用阈值激活函数的感知机，可以表示逻辑与、或、非、多数函数等线性可分的函数



表示逻辑与的感知机



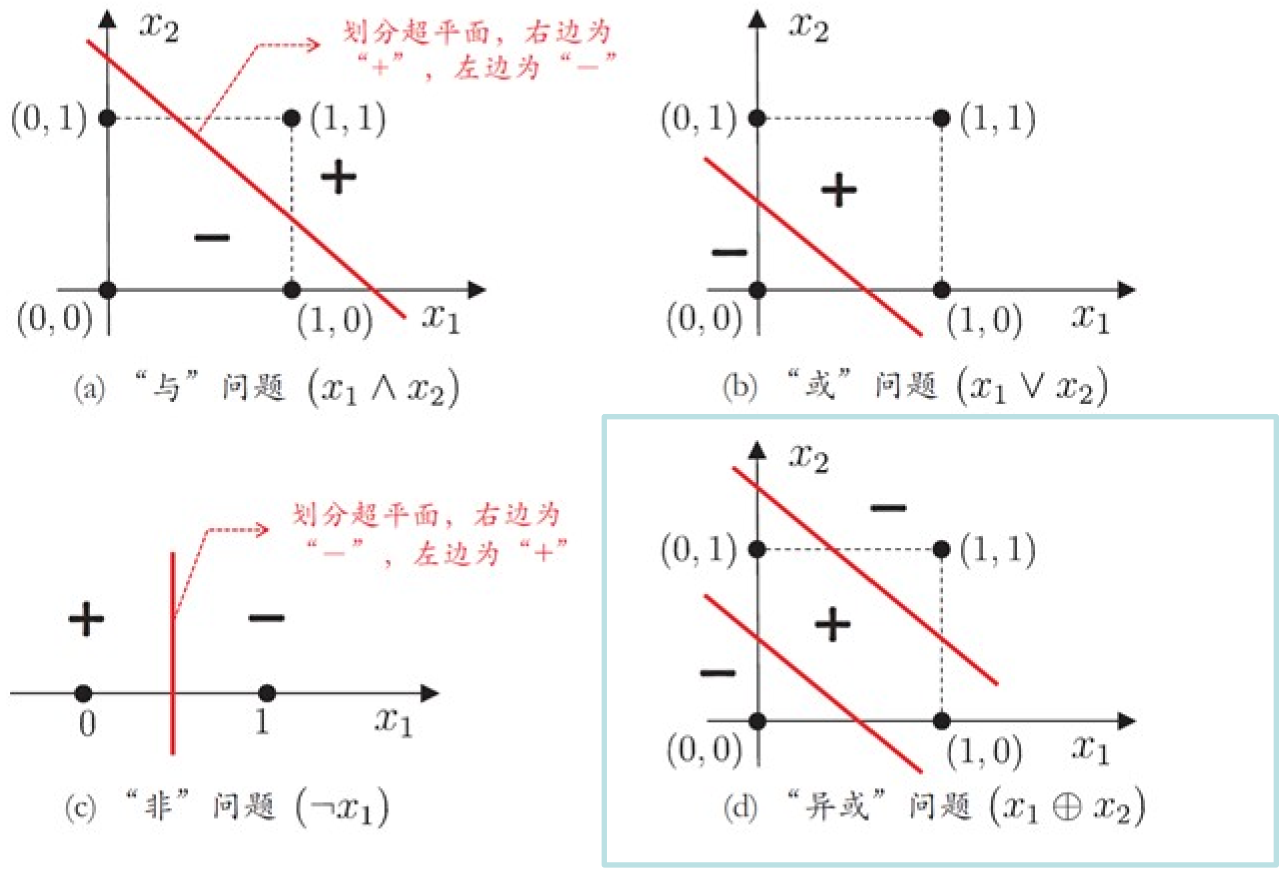
x_2
划分超平面，右边为"+"，左边为"−"
(0, 1)
(1, 1)
+
−
(0, 0)
(1, 0)
x_1
(a) "与"问题 ($x_1 \wedge x_2$)
(b) "或"问题 ($x_1 \vee x_2$)
划分超平面，右边为"−"，左边为"+"
0
1
(c) "非"问题 ($\neg x_1$)
(d) "异或"问题 ($x_1 \oplus x_2$)

线性可分的"与""或""非"问题与非线性可分的"异或"问题

感知机的表示能力

- 一个采用阈值激活函数的感知机，可以表示逻辑与、或、非、多数函数等线性可分的函数

- Minsky & Papert (1969)《Perceptron》感知器无法解决对 XOR（异或）这样的分类任务



划分超平面，右边为"+"，左边为"−"
(a) "与" 问题 $(x_1 \wedge x_2)$
(b) "或" 问题 $(x_1 \vee x_2)$
划分超平面，右边为"−"，左边为"+"
(c) "非" 问题 $(\neg x_1)$
(d) "异或" 问题 $(x_1 \oplus x_2)$

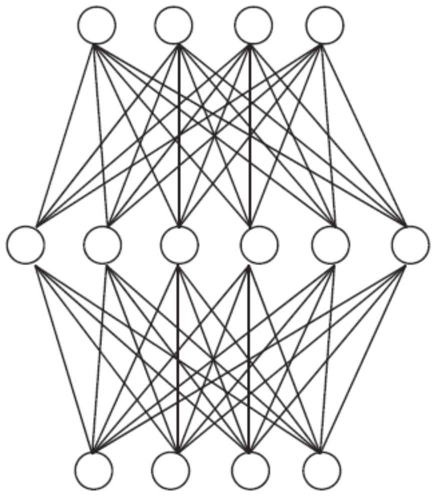
线性可分的"与""或""非"问题与非线性可分的"异或"问题

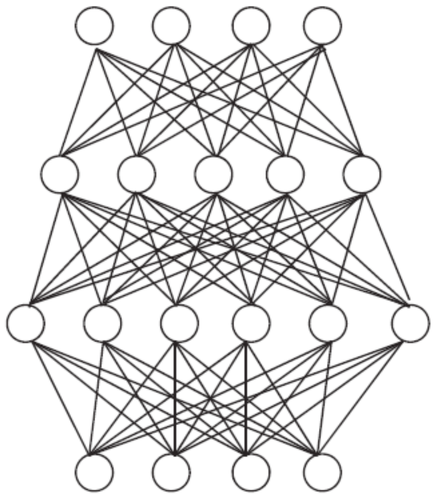
多层感知机 MLP

多层前馈神经网络

- 定义：每层神经元与下一层神经元全互联，神经元之间不存在同层连接也不存在跨层连接
- 前馈：输入层接受外界输入，隐含层与输出层神经元对信号进行加工，最终结果由输出层神经元输出
- 学习：根据训练数据来调整神经元之间的“连

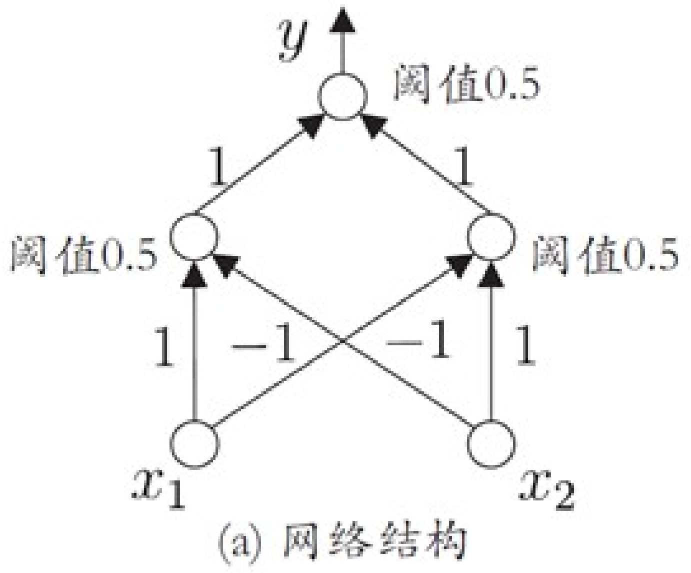
  - 隐藏层和输出层神经元都是具有激活函数的功能神经元


(a) 单隐层前馈网络

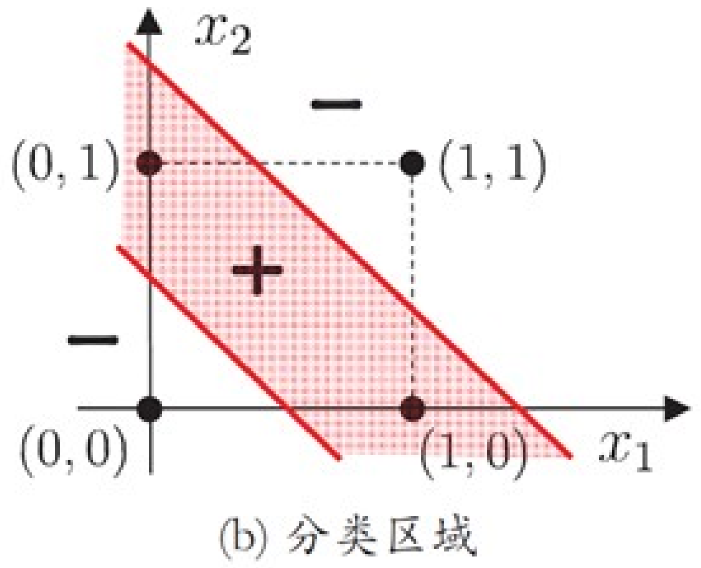

(b) 双隐层前馈网络

多层感知机

- 解决异或问题的两层感知机



(a) 网络结构



(b) 分类区域

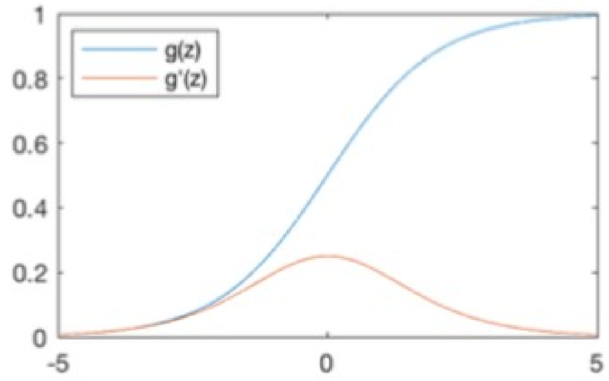
能解决异或问题的两层感知机

非线性激活函数

线性分类

$y = g(w_0 + w_1 x_1 + w_2 x_2 + \ldots)$

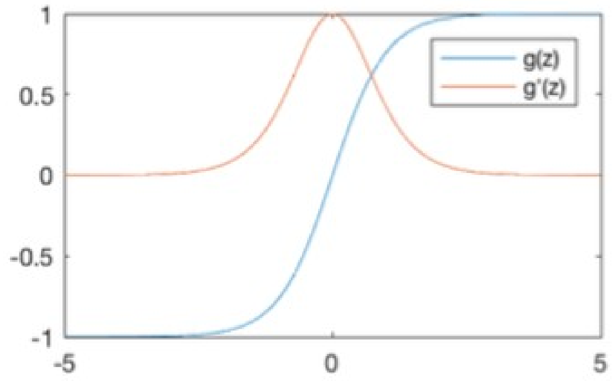
Sigmoid Function



$$g(z) = \frac{1}{1 + e^{-z}}$$

$$g'(z) = g(z)(1 - g(z))$$

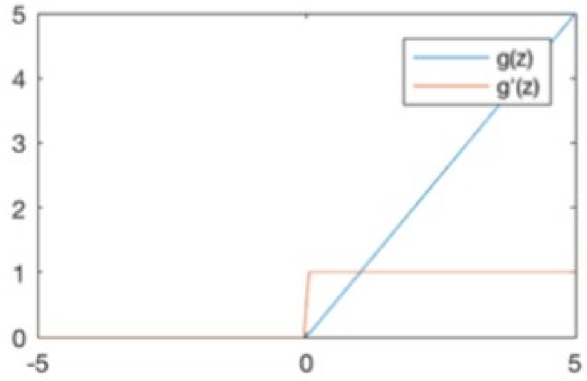
Hyperbolic Tangent



$$g(z) = \frac{e^{z} - e^{-z}}{e^{z} + e^{-z}}$$

$$g'(z) = 1 - g(z)^2$$

Rectified Linear Unit (ReLU)



$$g(z) = \max(0, z)$$

$$g'(z) = \begin{cases} 1, & z > 0 \\ 0, & \text{otherwise} \end{cases}$$

Probabilistic Decisions: Example

$$\frac{1}{1+e^{-wx}}$$

where w is some weight constant (vector) we have to learn, and wx is the dot product of w and x

- Suppose w = [–3, 4, 2] and x = [1, 2, 0]
- What label will be selected if we classify deterministically?
  - wx = –3+8+0 = 5
  - 5 is positive, so the classifier guesses the positive label
- What are the probabilities of each label if we classify probabilistically?
  - 1 / (1 + e^{-5}) = 0.9933 probability of positive label
  - 1 – 0.9933 = 0.0067 probability of negative label

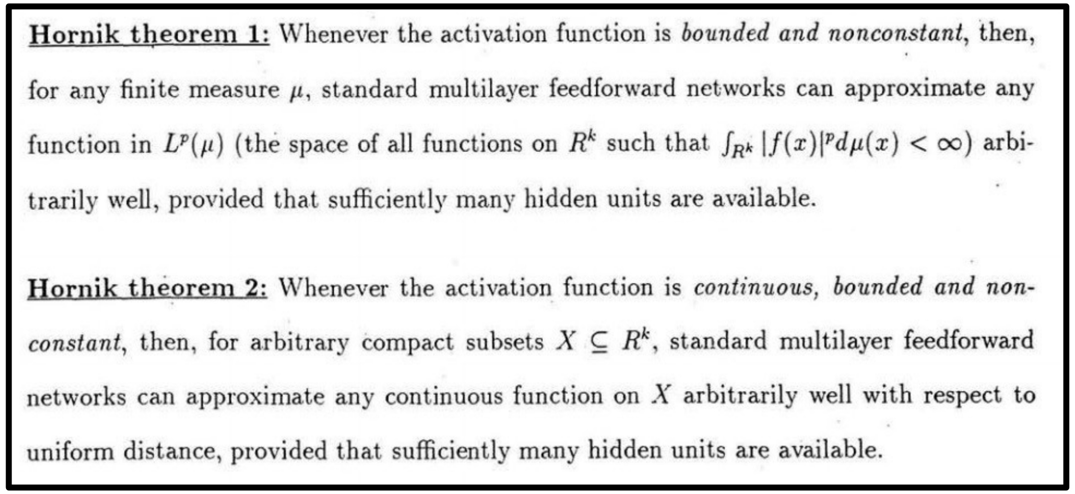
神经网络的表示能力

- 万能近似定理

只需要一个包含足够多神经元的隐层，多层前馈神经网络就能以任意精度逼近任意复杂度的连续函数 **[Hornik et al.， 1989]**

利用两个隐藏层，甚至能表示非连续的函数。

□□□□□□□□□□□□□□□□□□□□□□□□□□□□□、。□□□□□□□□□

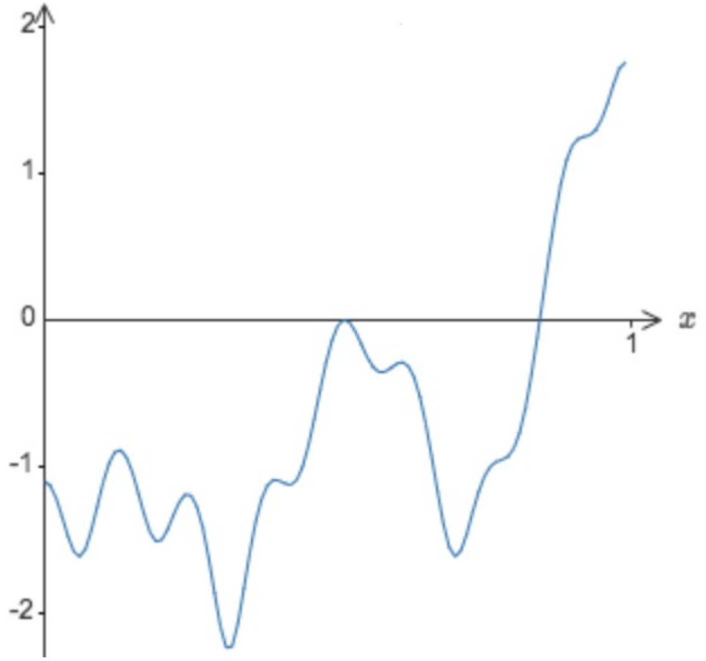

Hornik theorem 1: Whenever the activation function is *bounded and nonconstant*, then, for any finite measure μ, standard multilayer feedforward networks can approximate any function in $L^p(\mu)$ (the space of all functions on R^k such that $\int_{R^k} |f(x)|^p d\mu(x) < \infty$) arbitrarily well, provided that sufficiently many hidden units are available.

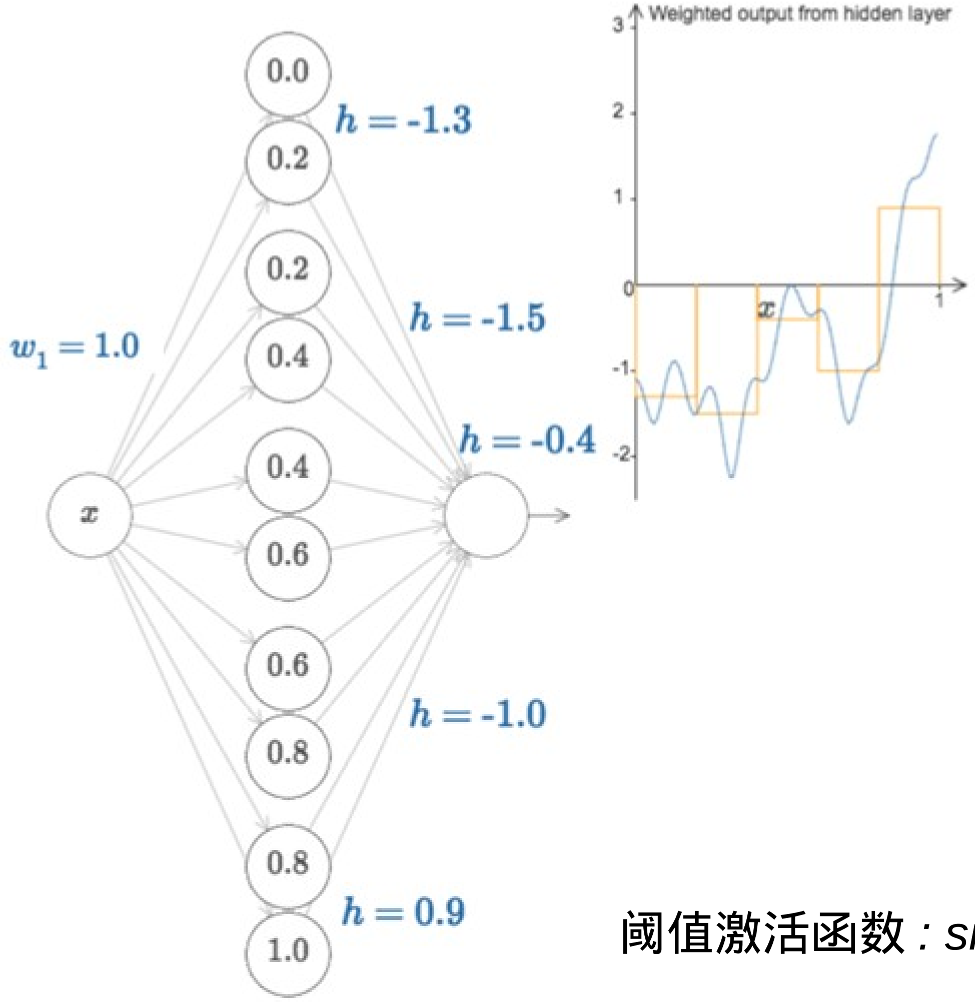
Hornik theorem 2: Whenever the activation function is *continuous, bounded and nonconstant*, then, for arbitrary compact subsets $X \subseteq R^k$, standard multilayer feedforward networks can approximate any continuous function on X arbitrarily well with respect to uniform distance, provided that sufficiently many hidden units are available.

Hornik, K., Stinchcombe, M. B., and White, H. (1989). Multilayer feedforward networks are universal approximators.
Cybenko (1989) “Approximation by superpositions of a sigmoidal function”

神经网络模拟函数

曲线函数：





阈值激活函数 : *sign(x)*

本质上，使用一个单层神经网络构建了一个 lookup 表，不同区间对应不同的值，区间分的越细小，就越准确。