



上海海事大学  
SHANGHAI MARITIME UNIVERSITY

# 自然语言处理

2024-2025 学年第 2 学期

信息工程学院 谢雨波



# 逻辑回归

# 逻辑回归

- 逻辑回归 (Logistic Regression)
  - 在自然科学与社会科学中，是一个重要的分析工具
  - 在监督式学习中，是分类任务的基准模型
  - 是神经网络的基础

# 生成模型和判别模型

- 朴素贝叶斯是一个生成模型 (**Generative Model**)
- 逻辑回归是一个判别模型 (**Discriminative Model**)

# 生成模型和判别模型

- 判断图片中是猫还是狗



(图片来自 ImageNet)

# 生成模型

- 对猫图中的内容进行建模
  - 胡须、眼睛、鼻子、耳朵.....
  - 为任意一张图片赋予概率值：
    - 这张图片是猫的概率有多少？
- 同样地，对狗图进行建模
- 给定一张新的图片：
  - 比较猫模型和狗模型哪一个更符合



# 判别模型

- 直接区分猫和狗



狗都有项圈，而猫没有

# 生成模型和判别模型

- 为输入文档  $d$  找到一个正确的类别标签  $c$
- 朴素贝叶斯（生成模型）：

$$\hat{c} = \operatorname{argmax}_{c \in \mathcal{C}} P(d | c) P(c)$$

似然

先验

- 逻辑回归（判别模型）：

$$\hat{c} = \operatorname{argmax}_{c \in \mathcal{C}} P(c | d)$$

后验

# 机器学习中的概率分类器

- 给定  $M$  个输入输出对  $(x^{(i)}, y^{(i)})$  (训练集) :
  1. 输入的特征表示 (**Feature Representation**) : 对每一个输入  $x^{(i)}$ , 有它的一个特征向量  $[x_1^{(i)}, x_2^{(i)}, \dots, x_n^{(i)}]$
  2. 分类函数 (**Classification Function**) : 通过  $P(y|x)$  得到预估的类别  $\hat{y}$ , 例如 sigmoid 和 softmax 函数
  3. 学习的目标函数 (**Objective Function**) : 例如交叉熵损失函数 (Cross-Entropy Loss Function)
  4. 目标函数的优化算法 (**Optimization Algorithm**) : 例如随机梯度下降 (Stochastic Gradient Descent)

# 逻辑回归的两个阶段

## 1. 训练 (Training)

- 使用随机梯度下降和交叉熵损失函数学习模型的  $w$  和  $b$

## 2. 测试 (Testing)

- 给定一个测试样本  $x$ , 使用学习到的参数  $w$  和  $b$  计算  $P(y|x)$ , 返回概率值最高的标签 ( $y = 0$  或  $y = 1$ )

# 二元逻辑回归

- 输入：向量  $x = [x_1, x_2, \dots, x_n]$
- 权重：向量  $w = [w_1, w_2, \dots, w_n]$   
或称：参数  $\theta = [\theta_1, \theta_2, \dots, \theta_n]$
- 输出：预测的类别  $\hat{y} \in \{0, 1\}$

# 如何进行分类？

- 对每一个输入特征  $x_i$ ，对应的权重  $w_i$  代表了  $x_i$  的重要性（另有一偏置  $b$ ）
- 例如，对于情感分析：
  - $x_i$  = “文本包含：太棒了”， $w_i = 10$
  - $x_j$  = “文本包含：真垃圾”， $w_j = -10$
  - $x_k$  = “文本包含：一般般”， $w_k = -2$

# 如何进行分类？

- 对每一个输入特征  $x_i$ ，对应的权重  $w_i$  代表了  $x_i$  的重要性（另有一偏置  $b$ ）
- 对所有输入特征进行加权求和，并加上偏置：

$$z = \left( \sum_{i=1}^n w_i x_i \right) + b$$

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

- 如果  $z$  值大，那么  $y = 1$ ；否则  $y = 0$

# 如何获得概率分类器？

- 需要对“ $z$  值大”进行量化
- 如同朴素贝叶斯一样，需要有一个输出概率值的分类器：

$$P(y = 1 | x; \theta)$$

$$P(y = 0 | x; \theta)$$

# 如何获得概率分类器？

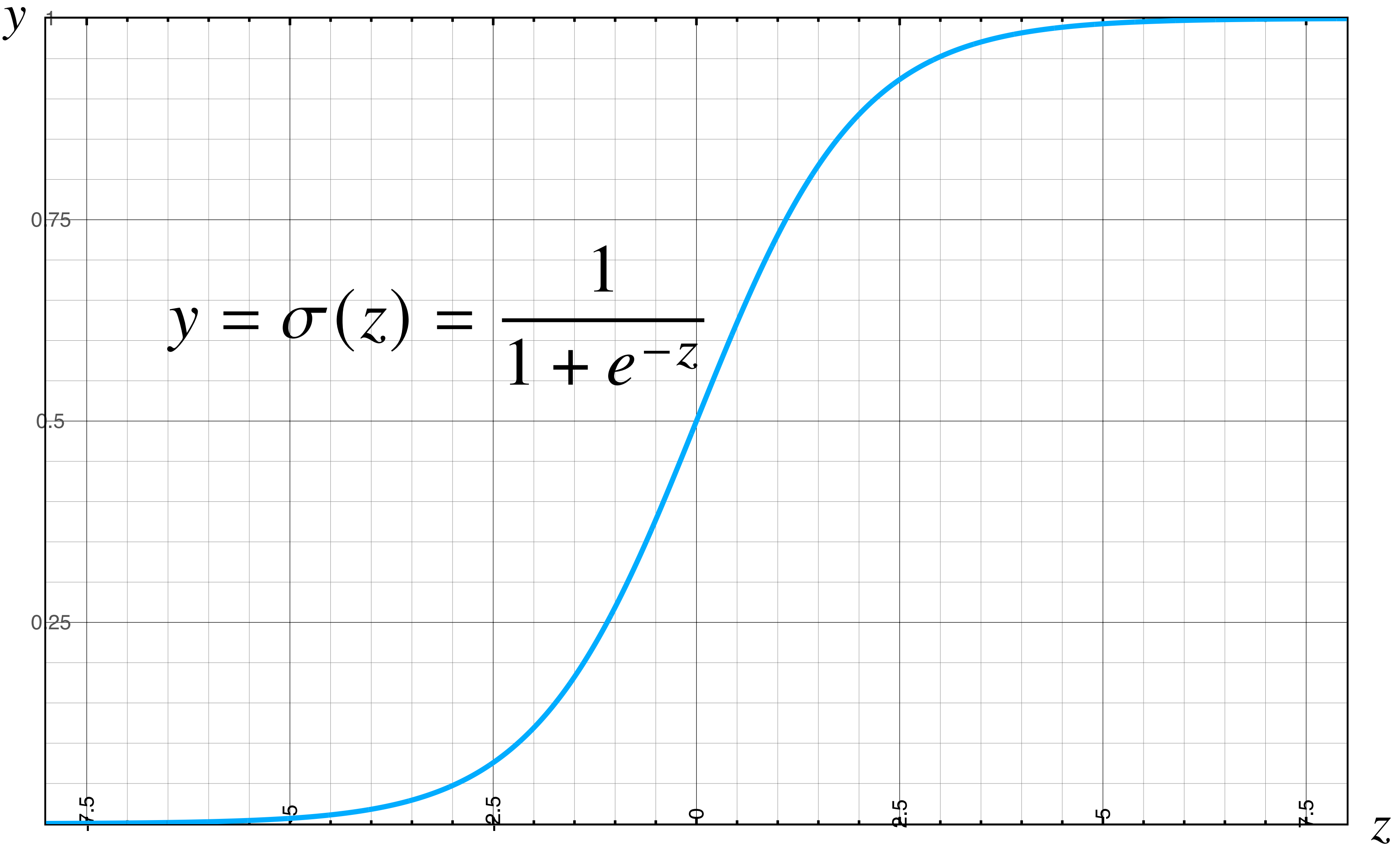
- $z$  只是一个数字，并不是一个概率值：

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

- 解决方法：使用一个函数将  $z$  映射至  $[0,1]$ ：

$$y = \sigma(z) = \frac{1}{1 + e^{-z}} = \frac{1}{1 + \exp(-z)}$$

# sigmoid 函数



# 使用 sigmoid 函数计算概率值

$$\begin{aligned} P(y = 1|x) &= \sigma(w \cdot x + b) \\ &= \frac{1}{1 + \exp(-(w \cdot x + b))} \end{aligned}$$

$$\begin{aligned} P(y = 0|x) &= 1 - \sigma(w \cdot x + b) \\ &= 1 - \frac{1}{1 + \exp(-(w \cdot x + b))} \end{aligned}$$

$$1 - \sigma(x) = \sigma(-x)$$

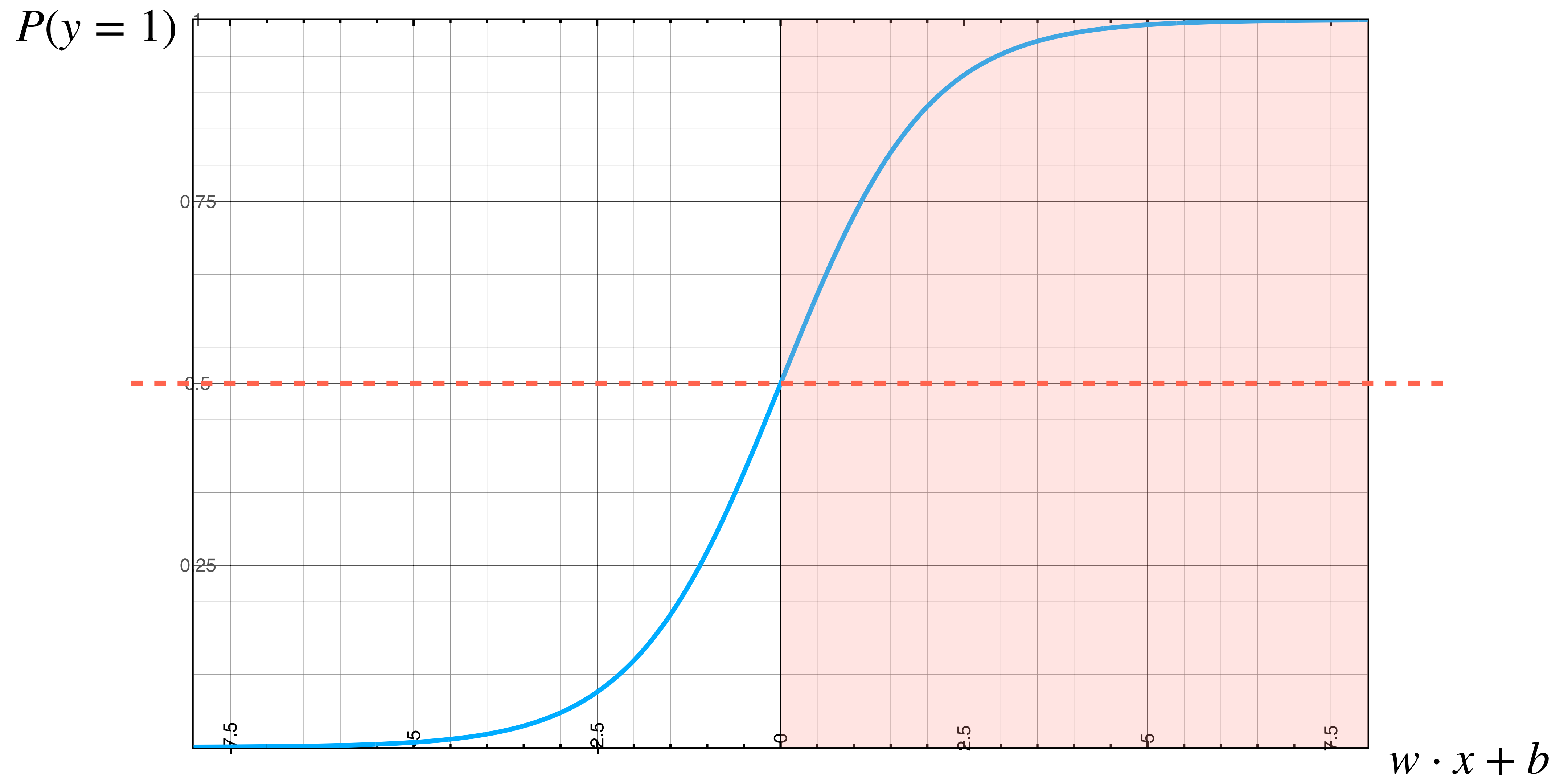
$$= \frac{\exp(-(w \cdot x + b))}{1 + \exp(-(w \cdot x + b))} = \frac{1}{1 + \exp(w \cdot x + b)} = \sigma(-(w \cdot x + b))$$

# 决策边界

$$\hat{y} = \begin{cases} 1 & \text{if } P(y = 1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases}$$

此处 0.5 即为分类器的**决策边界 (Decision Boundary)**

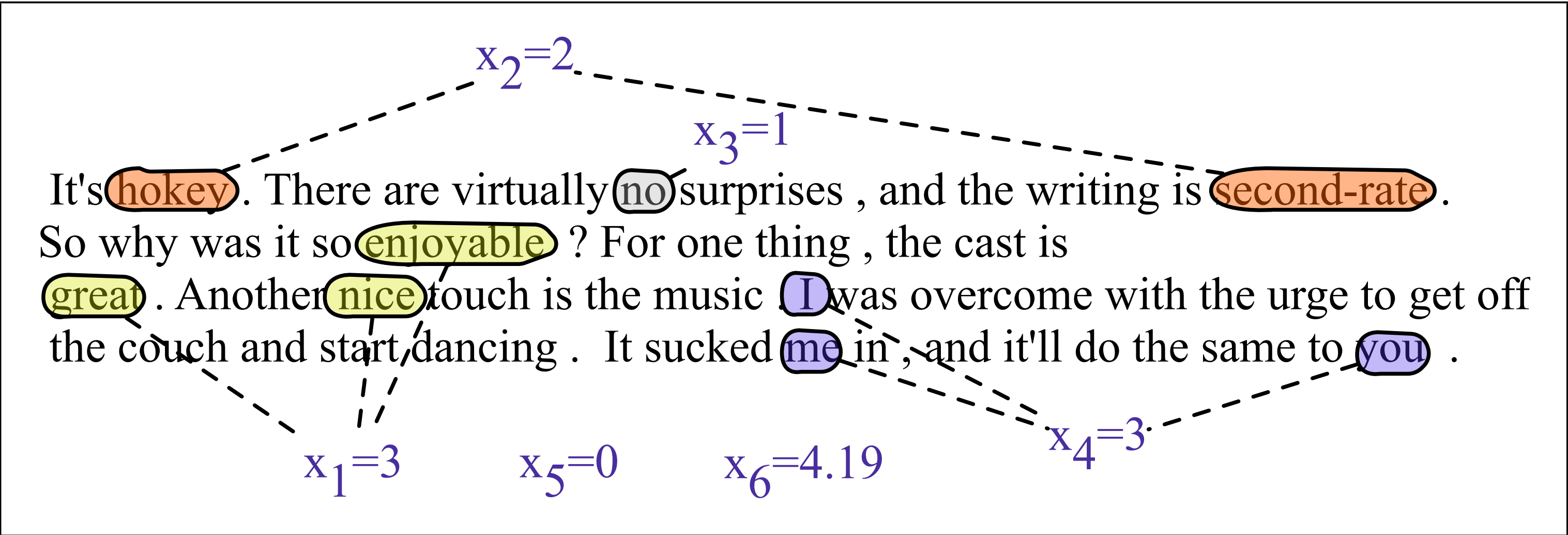
# 决策边界



# 决策边界

$$\hat{y} = \begin{cases} 1 & \text{if } P(y = 1|x) > 0.5 \\ 0 & \text{otherwise} \end{cases} \quad \begin{array}{l} \Leftrightarrow \quad \text{if } w \cdot x + b > 0 \\ \Leftrightarrow \quad \text{if } w \cdot x + b \leq 0 \end{array}$$

# 逻辑回归：以情感分析为例



$x_1$	count(positive lexicon) $\in$ doc)	3
$x_2$	count(negative lexicon) $\in$ doc)	2
$x_3$	$\begin{cases} 1 & \text{if "no" } \in \text{ doc} \\ 0 & \text{otherwise} \end{cases}$	1
$x_4$	count(1st and 2nd pronouns $\in$ doc)	3
$x_5$	$\begin{cases} 1 & \text{if "!" } \in \text{ doc} \\ 0 & \text{otherwise} \end{cases}$	0
$x_6$	log(word count of doc)	$\ln(66) = 4.19$

# 逻辑回归：以情感分析为例

- 假设：

- $w = [2.5, -5.0, -1.2, 0.5, 2.0, 0.7]$

- $b = 0.1$

$x_1$	count(positive lexicon) $\in$ doc)	3
$x_2$	count(negative lexicon) $\in$ doc)	2
$x_3$	$\begin{cases} 1 & \text{if "no"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	1
$x_4$	count(1st and 2nd pronouns $\in$ doc)	3
$x_5$	$\begin{cases} 1 & \text{if "!"} \in \text{doc} \\ 0 & \text{otherwise} \end{cases}$	0
$x_6$	log(word count of doc)	$\ln(66) = 4.19$

# 逻辑回归：以情感分析为例

✓ 
$$\begin{aligned} P(+|x) &= P(y = 1|x) = \sigma(w \cdot x + b) \\ &= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\ &= \sigma(0.833) \\ &= 0.70 \end{aligned}$$

$$\begin{aligned} P(-|x) &= P(y = 0|x) = 1 - \sigma(w \cdot x + b) \\ &= 0.30 \end{aligned}$$

# 逻辑回归模型的训练

# 模型参数 $w$ 和 $b$ 从何而来？

- 监督学习 (Supervised Learning) :
  - 对于每一个输入  $x^{(i)}$ , 有一个正确的标签  $y^{(i)}$
  - 同时, 模型的预测标签是  $\hat{y}^{(i)}$
- 找到  $w$  和  $b$ , 使得  $\hat{y}^{(i)}$  和  $y^{(i)}$  之间的距离最小:
  - 距离的衡量: **损失函数 (Loss Function, Cost Function)**
  - **优化算法**: 更新  $w$  和  $b$  使得损失函数最小

# 模型的训练

- 损失函数：
  - 交叉熵损失 (**Cross-Entropy Loss**)
- 优化算法：
  - 随机梯度下降 (**Stochastic Gradient Descent**)

# $\hat{y}$ 和 $y$ 之间的距离

- 模型的输出为：

$$\hat{y} = P(y = 1 | x) = \sigma(w \cdot x + b)$$

- 正确的输出应该为：

$$y \text{ (0 或 1)}$$

- $\hat{y}$  和  $y$  之间的距离记为

$$L(\hat{y}, y)$$

# 交叉熵损失

- 交叉熵损失 (Cross-Entropy Loss)
  - 即：负对数似然损失 (Negative Log Likelihood Loss)
- 条件最大似然估计：
  - 给定输入  $x$
  - 找到  $w$  和  $b$ ，使得正确标签  $y$  的对数概率值最大

# 交叉熵损失

- 目标：最大化正确标签  $y$  的条件概率  $P(y | x)$
- 分两种情况进行讨论：
  - 如果正确标签  $y = 1$ ，那么  $P(y | x) = \sigma(w \cdot x + b) = \hat{y}$
  - 如果正确标签  $y = 0$ ，那么  $P(y | x) = 1 - \sigma(w \cdot x + b) = 1 - \hat{y}$
- 写成一个表达式：

$$P(y | x) = \hat{y}^y (1 - \hat{y})^{1-y}$$

# 交叉熵损失

- 目标：最大化正确标签  $y$  的条件概率  $P(y | x)$

$$\text{最大化： } P(y | x) = \hat{y}^y (1 - \hat{y})^{1-y}$$

- 两边取  $\log$ （数学上更加简便）

$$\begin{aligned}\text{最大化： } \log P(y | x) &= \log [\hat{y}^y (1 - \hat{y})^{1-y}] \\ &= y \log \hat{y} + (1 - y) \log(1 - \hat{y})\end{aligned}$$

# 交叉熵损失

- 目标：最大化正确标签  $y$  的条件概率  $\log P(y|x)$

$$\text{最大化: } \log P(y|x) = y \log \hat{y} + (1 - y) \log(1 - \hat{y})$$

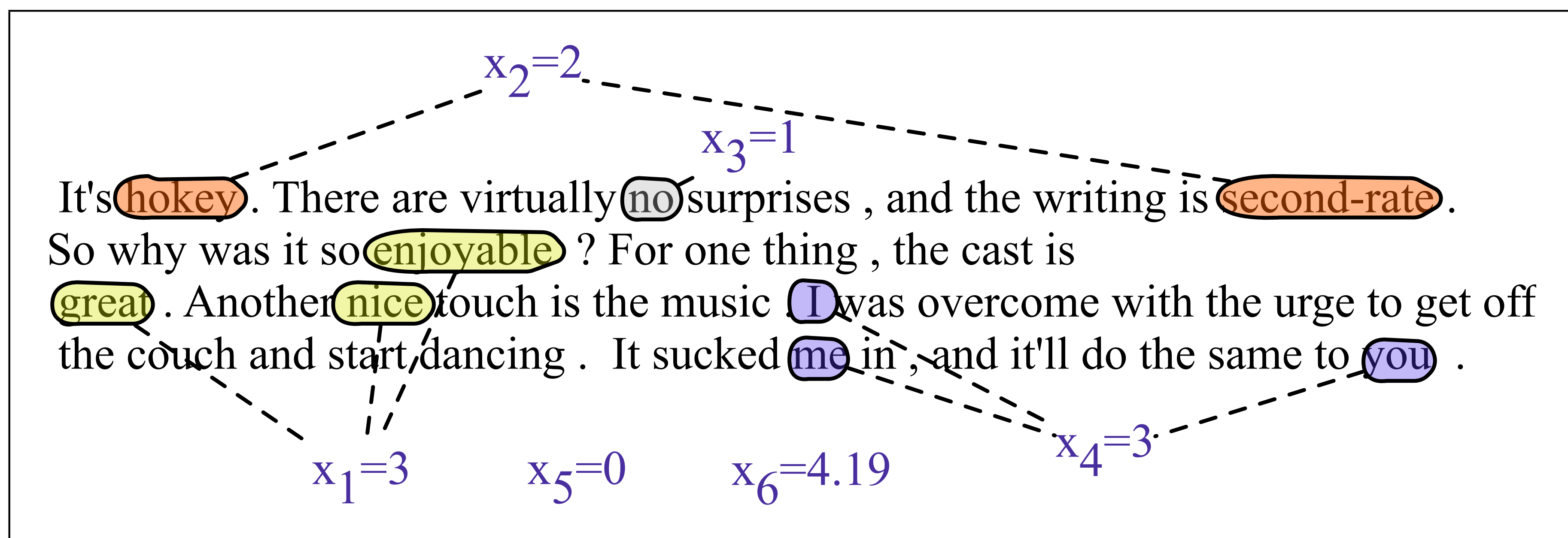
- 两边取负，得到交叉熵损失函数：

$$\text{最小化: } L_{\text{CE}}(\hat{y}, y) = -y \log \hat{y} - (1 - y) \log(1 - \hat{y})$$

$$L_{\text{CE}}(\hat{y}, y) = -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)]$$

# 交叉熵损失：以情感分析为例

- 模型预测越接近正确标签，损失函数值越小
- 反之，损失函数值越大



# 交叉熵损失：以情感分析为例

- 假设正确的标签为  $y = 1$

$$\begin{aligned} P(+|x) &= P(y = 1|x) = \sigma(w \cdot x + b) \\ &= \sigma([2.5, -5.0, -1.2, 0.5, 2.0, 0.7] \cdot [3, 2, 1, 3, 0, 4.19] + 0.1) \\ &= \sigma(0.833) \\ &= 0.70 \end{aligned}$$

- 模型的损失函数值为

$$\begin{aligned} L_{\text{CE}}(\hat{y}, y) &= -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)] \\ &= -\log \sigma(w \cdot x + b) \\ &= -\log 0.70 = 0.36 \end{aligned}$$

# 交叉熵损失：以情感分析为例

- 假设正确的标签为  $y = 0$

$$\begin{aligned} P(-|x) &= P(y = 0|x) = 1 - \sigma(w \cdot x + b) \\ &= 0.30 \end{aligned}$$

- 模型的损失函数值为

$$\begin{aligned} L_{\text{CE}}(\hat{y}, y) &= -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)] \\ &= -\log [1 - \sigma(w \cdot x + b)] \\ &= -\log 0.30 = 1.2 \end{aligned}$$

# 最小化损失函数

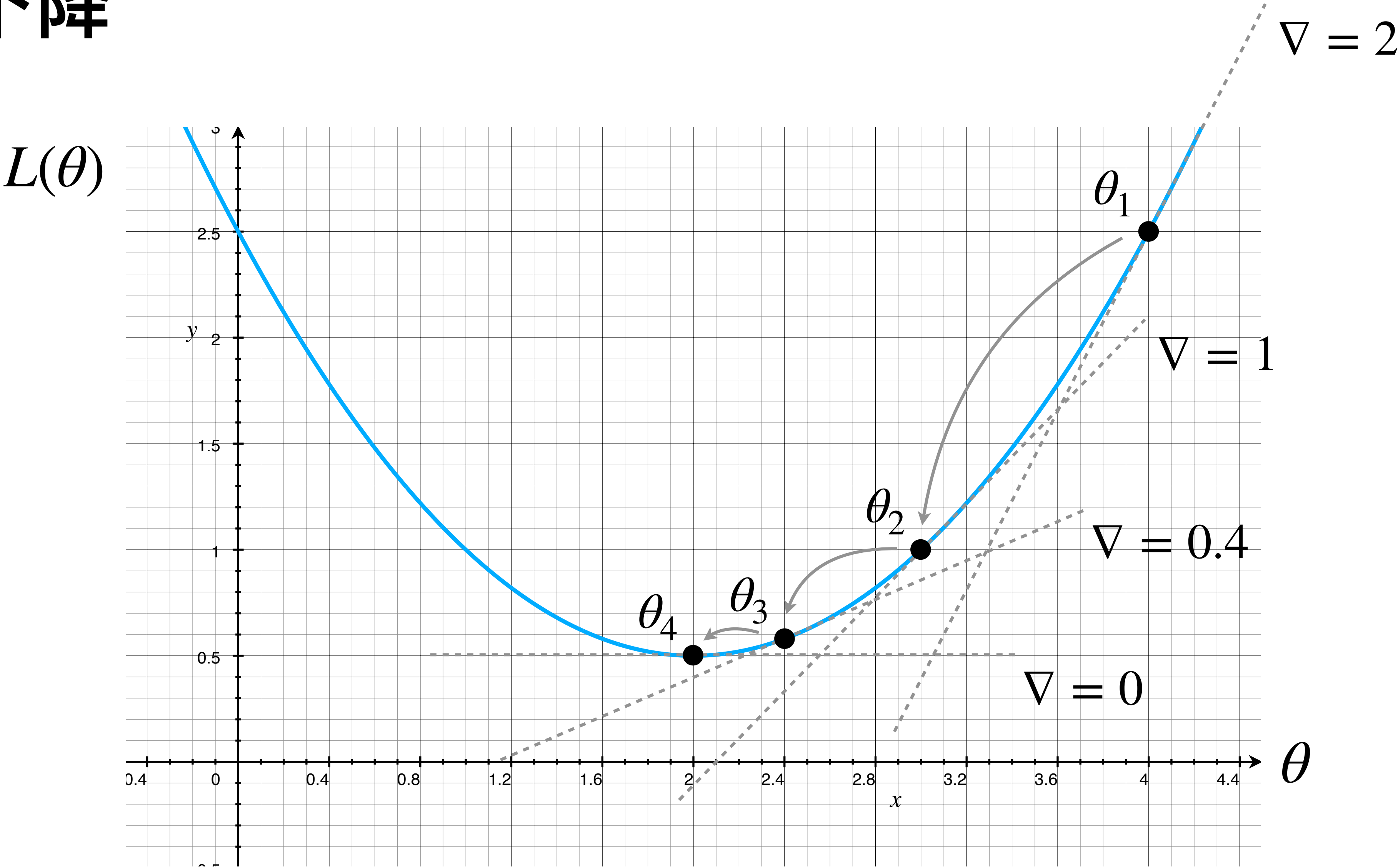
- 损失函数的参数为  $\theta = (w, b)$
- 将模型视作一个函数  $\hat{y} = f(x; \theta)$
- 找到  $\hat{\theta}$ , 使得数据集上的平均损失函数值最小:

$$\hat{\theta} = \operatorname{argmin}_{\theta} \frac{1}{M} \sum_{i=1}^M L_{\text{CE}}\left(f(x^{(i)}; \theta), y^{(i)}\right)$$

# 最小化损失函数

- 对于逻辑回归来说，损失函数为凸函数 (**Convex Function**)
- 凸函数只有一个最小值
- 从任意点开始梯度下降，总能到达最小值
- 神经网络的损失函数不是凸函数

# 梯度下降



# 梯度下降

- **梯度 (Gradient)**：一个向量，其方向指向函数增长最快的方向，而其大小（或长度）表示函数在这个方向上变化的速度

$$\nabla_{\theta} L(\theta) = \frac{d}{d\theta} L(\theta)$$

- **梯度下降 (Gradient Descent)**：在当前点计算损失函数的梯度，然后朝着梯度相反的方向移动

$$\theta^{t+1} = \theta^t - \nabla_{\theta} L(\theta)$$

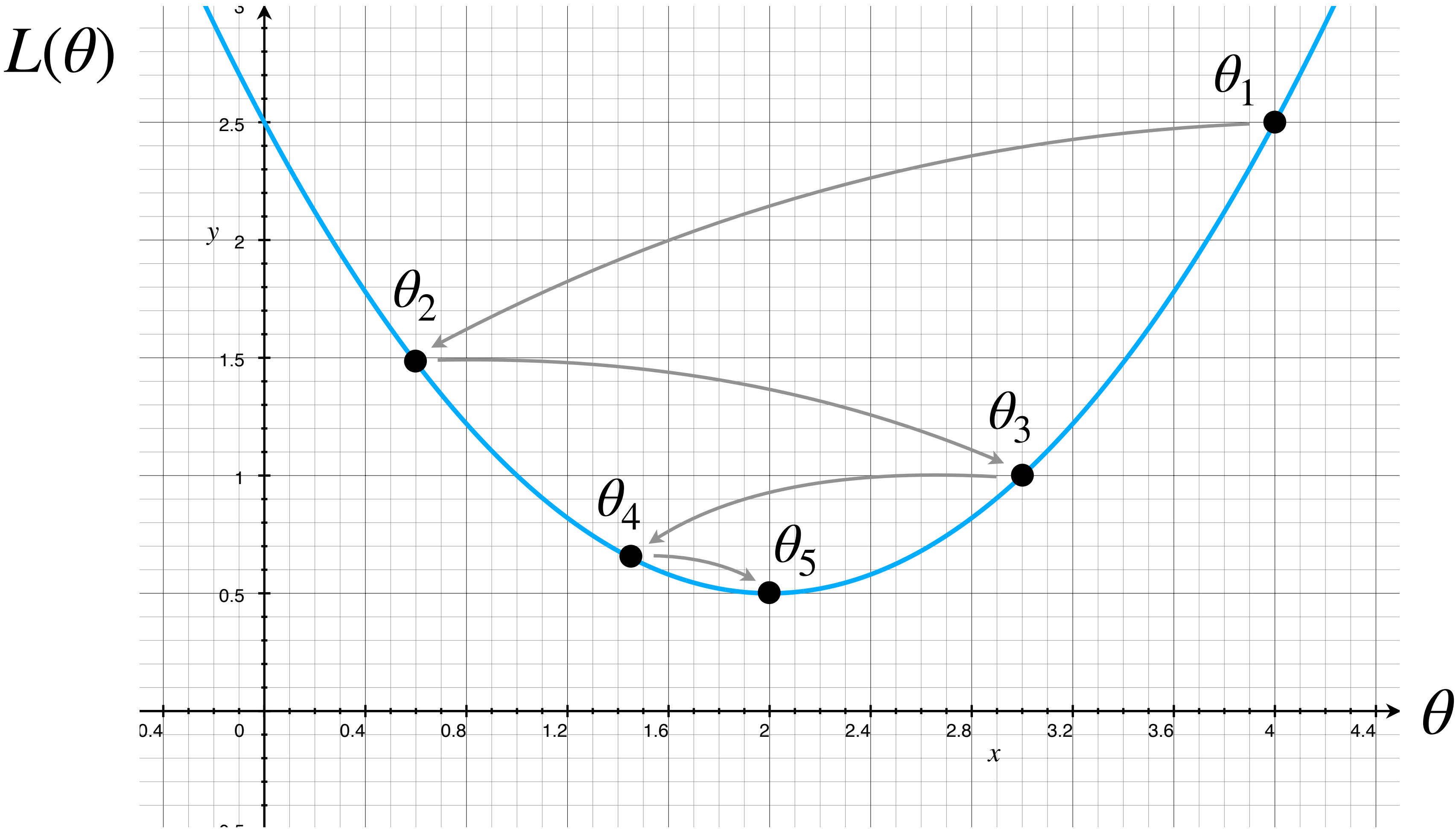
# 学习率

- 在梯度前乘上一个系数，即**学习率 (Learning Rate)**  $\eta > 0$

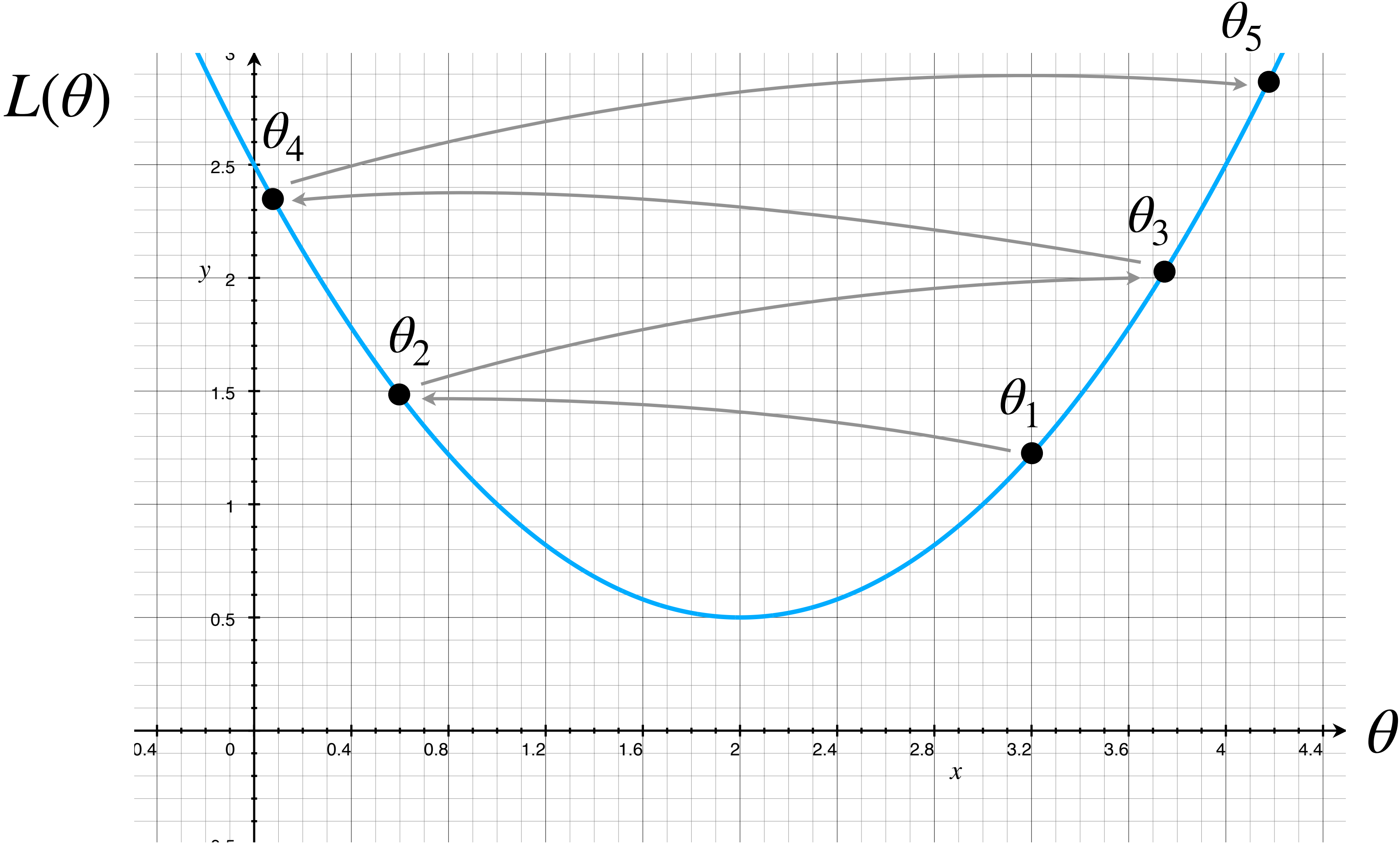
$$\begin{aligned}\theta^{t+1} &= \theta^t - \eta \nabla_{\theta} L(\theta) \\ &= \theta^t - \eta \frac{d}{d\theta} L(f(x; \theta), y)\end{aligned}$$

- 学习率越高，则  $\theta$  移动地越快

# 设置合理的学习率



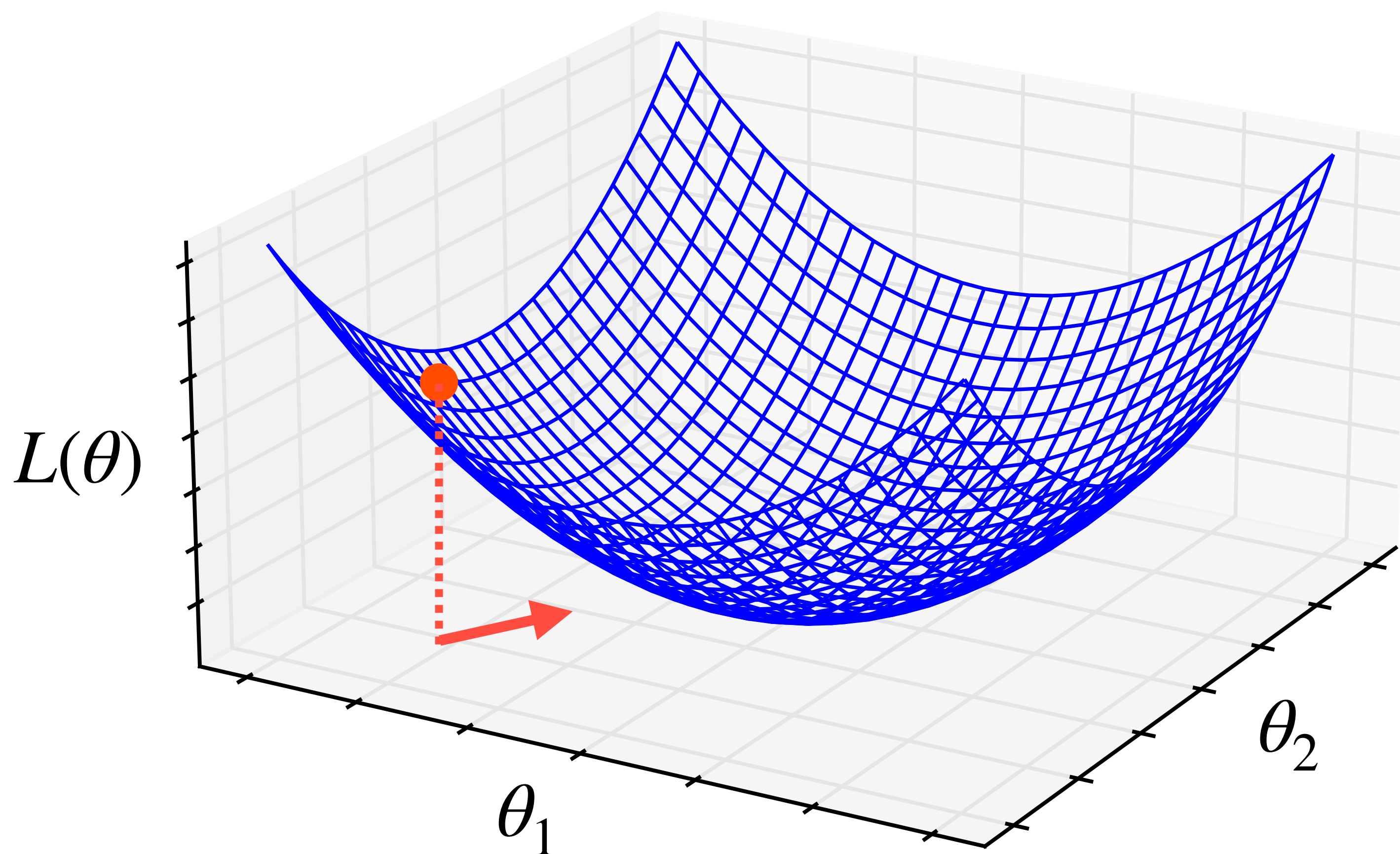
# 设置合理的学习率



# 超参数

- 学习率  $\eta$  是模型的一个超参数 (**Hyperparameter**)
- 超参数是机器学习模型的一种特殊参数
- 超参数不是通过监督学习得到的，而是模型设计者手动设定的
- 可以通过验证集 (Validation Set) 来设定

# 多元损失函数的梯度



$$\nabla_{\theta} L(\theta) = \begin{bmatrix} \frac{\partial}{\partial \theta_1} L(\theta) \\ \frac{\partial}{\partial \theta_2} L(\theta) \\ \vdots \\ \frac{\partial}{\partial \theta_n} L(\theta) \end{bmatrix}$$

$$\theta^{t+1} = \theta^t - \eta \nabla_{\theta} L(\theta)$$

# 逻辑回归损失函数的梯度

- 逻辑回归的损失函数：

$$L_{\text{CE}} = -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)]$$

$$\frac{\partial L_{\text{CE}}}{\partial w_j} = ?$$

$$\frac{\partial L_{\text{CE}}}{\partial b} = ?$$

# 逻辑回归损失函数的梯度

$$\begin{aligned}\frac{d\sigma(z)}{dz} &= \frac{d}{dz} \frac{1}{1 + e^{-z}} \\&= -\frac{1}{(1 + e^{-z})^2} \cdot \frac{d}{dz}(1 + e^{-z}) \\&= \frac{e^{-z}}{(1 + e^{-z})^2} \\&= \frac{1}{1 + e^{-z}} \cdot \frac{e^{-z}}{1 + e^{-z}} \\&= \sigma(z) \cdot (1 - \sigma(z))\end{aligned}$$

$$\frac{d\sigma(z)}{dz} = \sigma(z) \cdot (1 - \sigma(z))$$

# 逻辑回归损失函数的梯度

$$\begin{aligned}\frac{\partial L_{\text{CE}}}{\partial w_j} &= \frac{\partial}{\partial w_j} \left[ -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)] \right] \\ &= -\frac{\partial}{\partial w_j} y \log \sigma(w \cdot x + b) - \frac{\partial}{\partial w_j} (1 - y) \log [1 - \sigma(w \cdot x + b)] \\ &= -\frac{y}{\sigma(w \cdot x + b)} \frac{\partial}{\partial w_j} \sigma(w \cdot x + b) - \frac{(1 - y)}{1 - \sigma(w \cdot x + b)} \frac{\partial}{\partial w_j} [1 - \sigma(w \cdot x + b)] \\ &= -\left[ \frac{y}{\sigma(w \cdot x + b)} - \frac{(1 - y)}{1 - \sigma(w \cdot x + b)} \right] \frac{\partial}{\partial w_j} \sigma(w \cdot x + b)\end{aligned}$$

# 逻辑回归损失函数的梯度

$$\frac{d\sigma(z)}{dz} = \sigma(z) \cdot (1 - \sigma(z))$$

$$\begin{aligned}\frac{\partial L_{\text{CE}}}{\partial w_j} &= - \left[ \frac{y}{\sigma(w \cdot x + b)} - \frac{(1 - y)}{1 - \sigma(w \cdot x + b)} \right] \frac{\partial}{\partial w_j} \sigma(w \cdot x + b) \\&= - \frac{y - \sigma(w \cdot x + b)}{\sigma(w \cdot x + b) [1 - \sigma(w \cdot x + b)]} \frac{\partial}{\partial w_j} \sigma(w \cdot x + b) \\&= - \frac{y - \sigma(w \cdot x + b)}{\sigma(w \cdot x + b) [1 - \sigma(w \cdot x + b)]} \sigma(w \cdot x + b) [1 - \sigma(w \cdot x + b)] \frac{\partial(w \cdot x + b)}{\partial w_j} \\&= [\sigma(w \cdot x + b) - y] x_j\end{aligned}$$

# 逻辑回归损失函数的梯度

$$\begin{aligned}\frac{\partial L_{\text{CE}}}{\partial b} &= \frac{\partial}{\partial b} \left[ -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)] \right] \\ &= \dots \\ &= - \frac{y - \sigma(w \cdot x + b)}{\sigma(w \cdot x + b) [1 - \sigma(w \cdot x + b)]} \sigma(w \cdot x + b) [1 - \sigma(w \cdot x + b)] \frac{\partial (w \cdot x + b)}{\partial b} \\ &= \sigma(w \cdot x + b) - y\end{aligned}$$

# 逻辑回归损失函数的梯度

- 逻辑回归的损失函数：

$$L_{\text{CE}} = -y \log \sigma(w \cdot x + b) - (1 - y) \log [1 - \sigma(w \cdot x + b)]$$

$$\frac{\partial L_{\text{CE}}}{\partial w_j} = [\sigma(w \cdot x + b) - y] x_j \quad \frac{\partial L_{\text{CE}}}{\partial b} = \sigma(w \cdot x + b) - y$$

$$\text{令 } x_0 = 1, w_0 = b$$

$$\frac{\partial L_{\text{CE}}}{\partial w_j} = [\sigma(w \cdot x) - y] x_j \quad (j = 0, \dots, n)$$

# 逻辑回归的梯度下降

- 训练数据集上的平均损失函数：

$$L_{\text{CE}} = \frac{1}{M} \sum_{i=1}^M L_{\text{CE}}\left(f(x^{(i)}; \theta), y^{(i)}\right)$$

- 梯度下降：

$$w_j^{t+1} = w_j^t - \eta \frac{\partial L_{\text{CE}}}{\partial w_j}$$

$$w_j^{t+1} = w_j^t - \eta \frac{1}{M} \sum_{i=1}^M \left[ \sigma(w^t \cdot x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

# 随机梯度下降

- 随机梯度下降 (**Stochastic Gradient Descent**)
  - 每次更新  $w_j^t$  时, 随机挑选一个训练样本  $(x^{(i)}, y^{(i)})$ , 计算梯度

$$w_j^{t+1} = w_j^t - \eta \frac{1}{M} \sum_{i=1}^M \left[ \sigma(w^t \cdot x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

⇓

$$w_j^{t+1} = w_j^t - \eta \left[ \sigma(w^t \cdot x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

# 随机梯度下降

- 随机梯度下降的优点
  - 高效：每次更新只需要在一个训练样本上计算梯度
  - 在某些情况下，可以帮助逃离局部极值点（Local Minimum）
- 随机梯度下降的缺点：噪音较大，收敛可能较慢
- 小批量训练（**Mini-Batch Training**）：
  - 每次更新  $w_j^t$  时，随机挑选大小为  $m$  的批量训练样本计算梯度

$$w_j^{t+1} = w_j^t - \eta \frac{1}{m} \sum_{i=1}^m \left[ \sigma(w^t \cdot x^{(i)}) - y^{(i)} \right] x_j^{(i)}$$

# 梯度下降的向量化表示

- 令  $X \in \mathbb{R}^{m \times (n+1)}$  为小批量训练样本中的  $m$  个  $x^{(i)} \in \mathbb{R}^{(n+1) \times 1}$  ( $x_0^{(i)} = 1$ )
- 令  $y \in \mathbb{R}^{m \times 1}$  为小批量训练样本中的  $m$  个  $y^{(i)}$
- 令  $w \in \mathbb{R}^{(n+1) \times 1}$  为模型的参数 ( $w_0 = b$ )
- 则梯度可以表示为向量

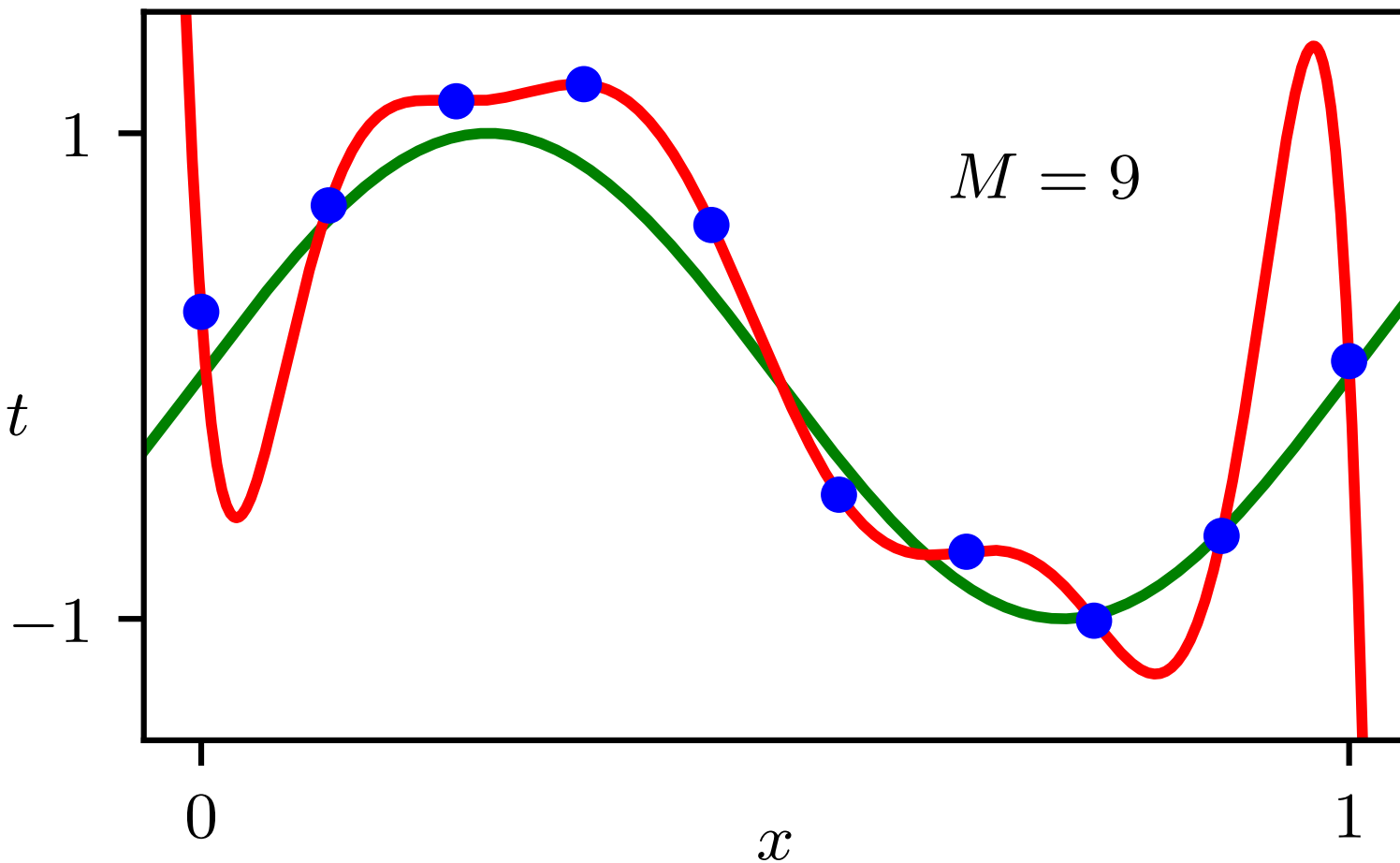
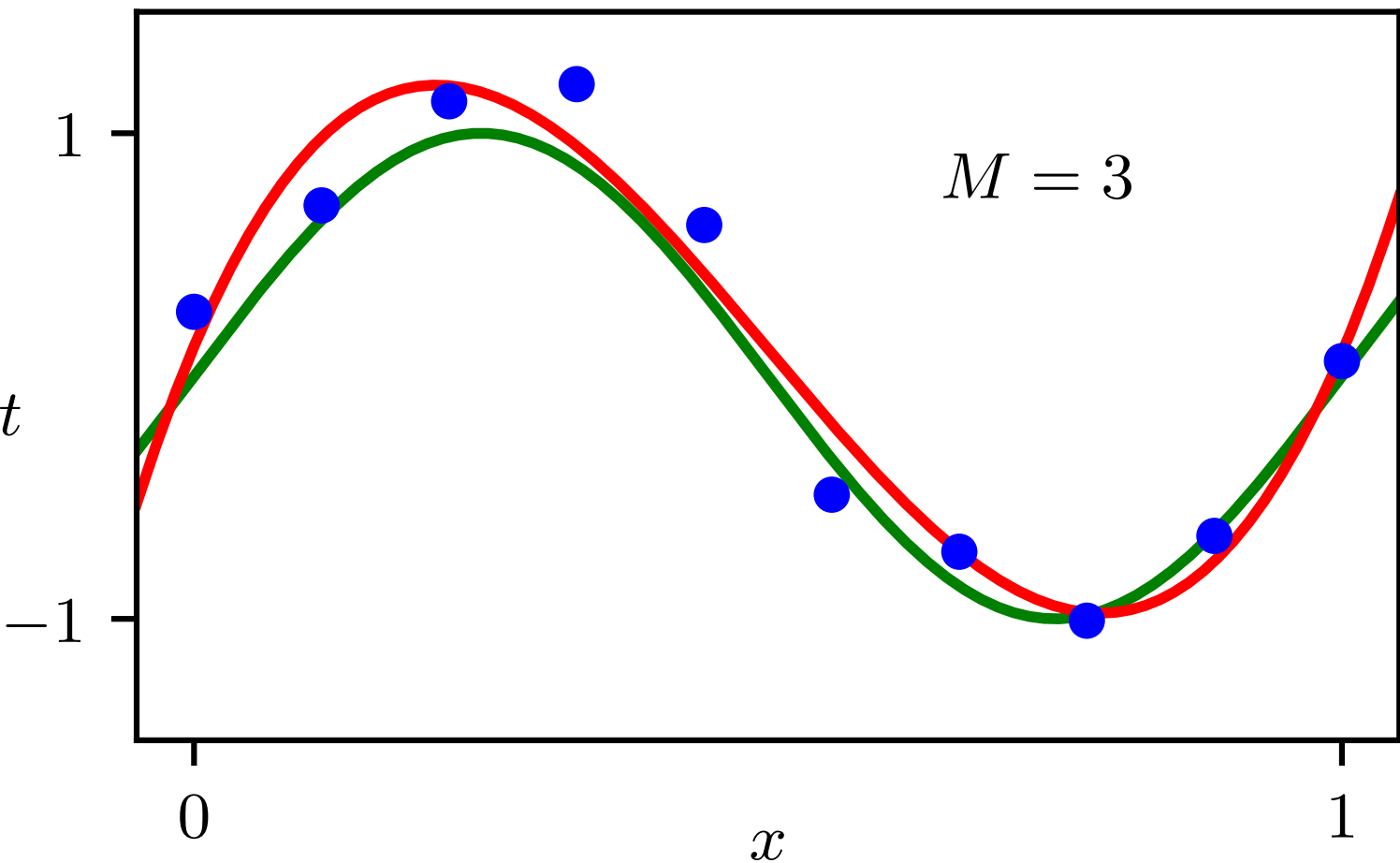
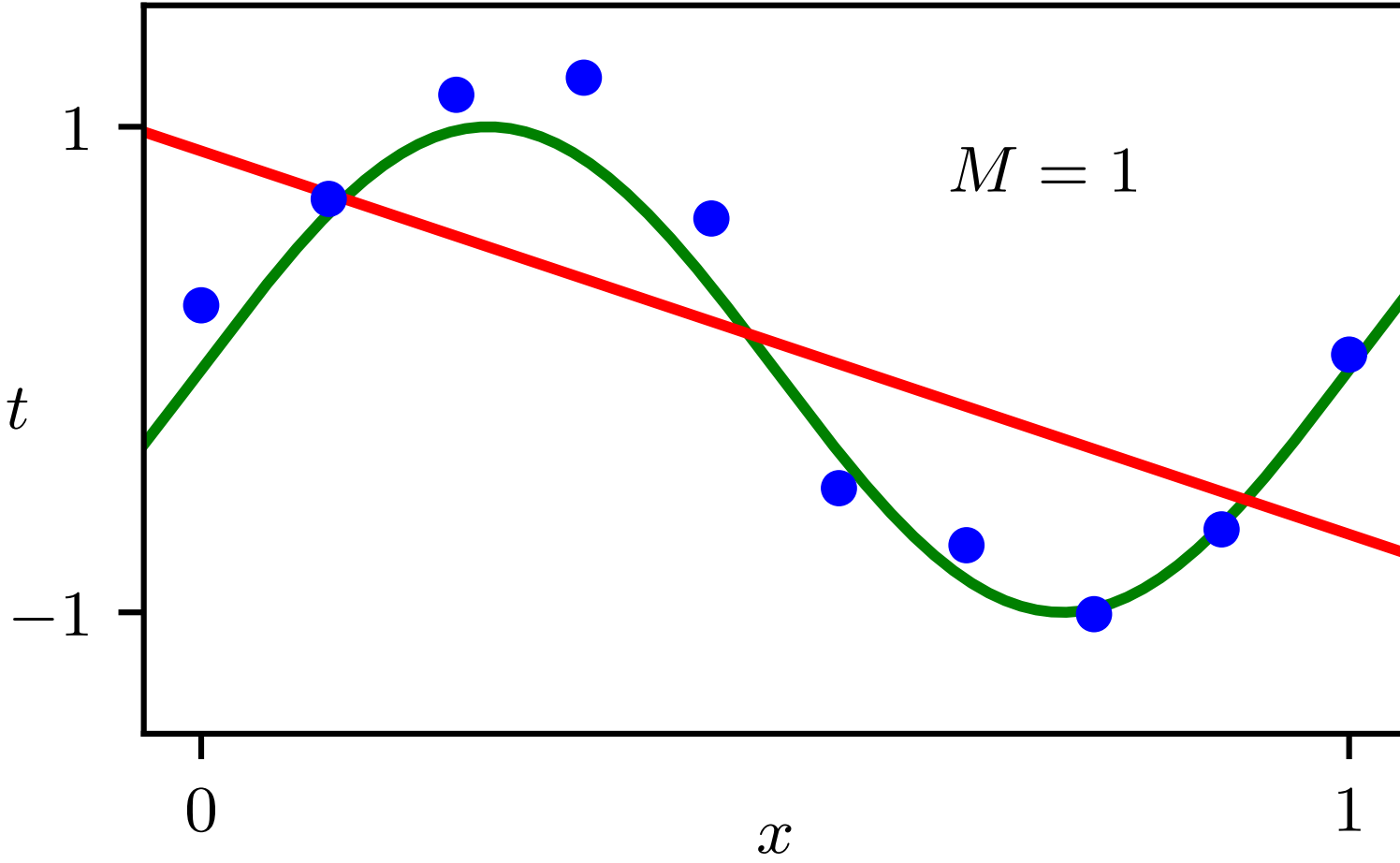
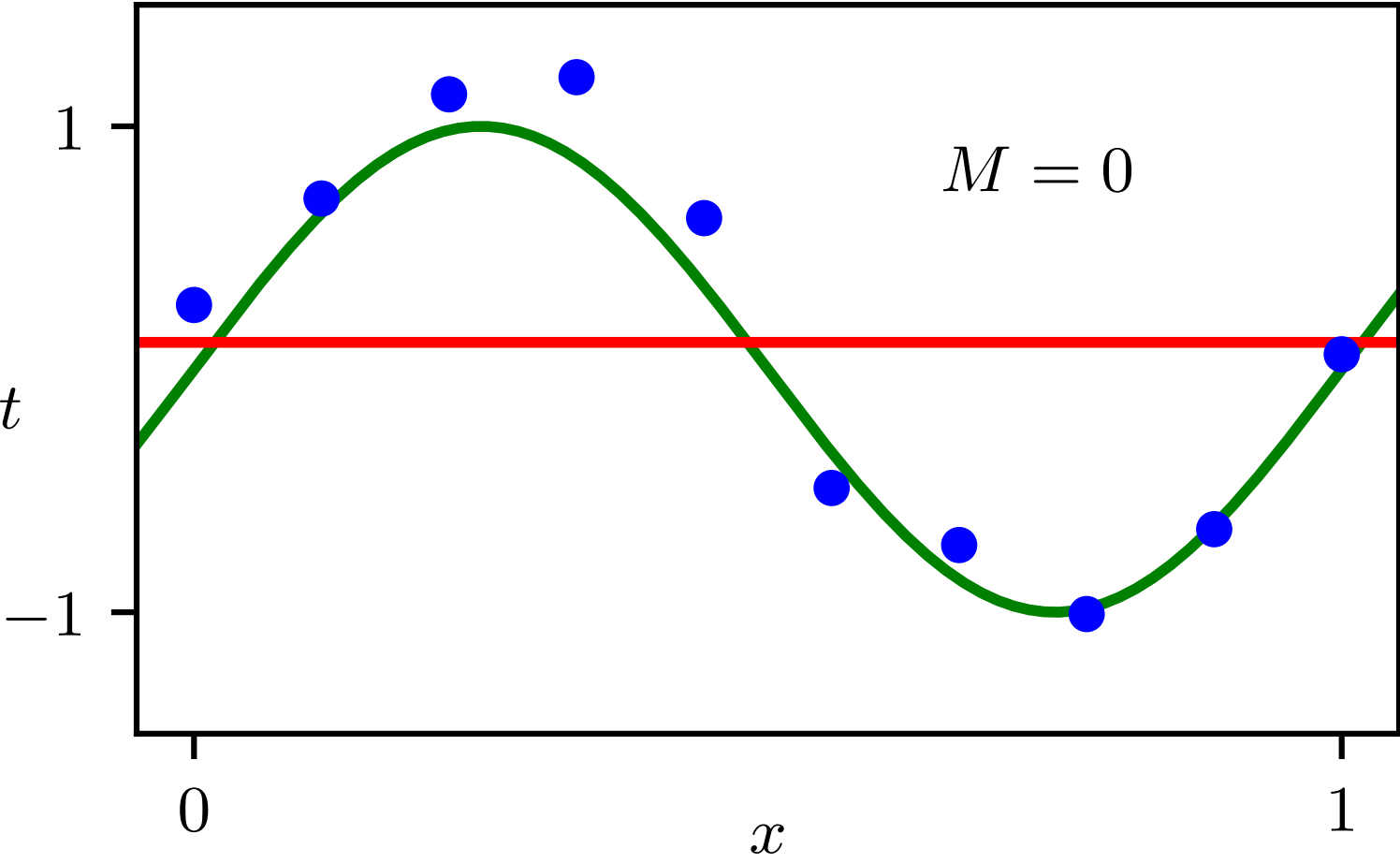
$$\frac{\partial L_{\text{CE}}}{\partial w} = \frac{1}{m} X^T [\sigma(Xw) - y]$$

# 过拟合与正则化

# 过拟合

- **过拟合 (Overfitting) :**
  - 模型在训练数据上表现得非常好，但是对于新的、未见过的数据却表现不佳
  - 模型过于复杂，学习到了训练数据中的“噪声”或随机波动，而不是真实的数据分布规律
- 过拟合的特征：
  - **在训练集上的误差非常小：** 模型几乎完美地预测了训练数据中的每个点
  - **在验证集或测试集上误差较大：** 当模型应用于未见过的数据时，性能明显下降

# 过拟合

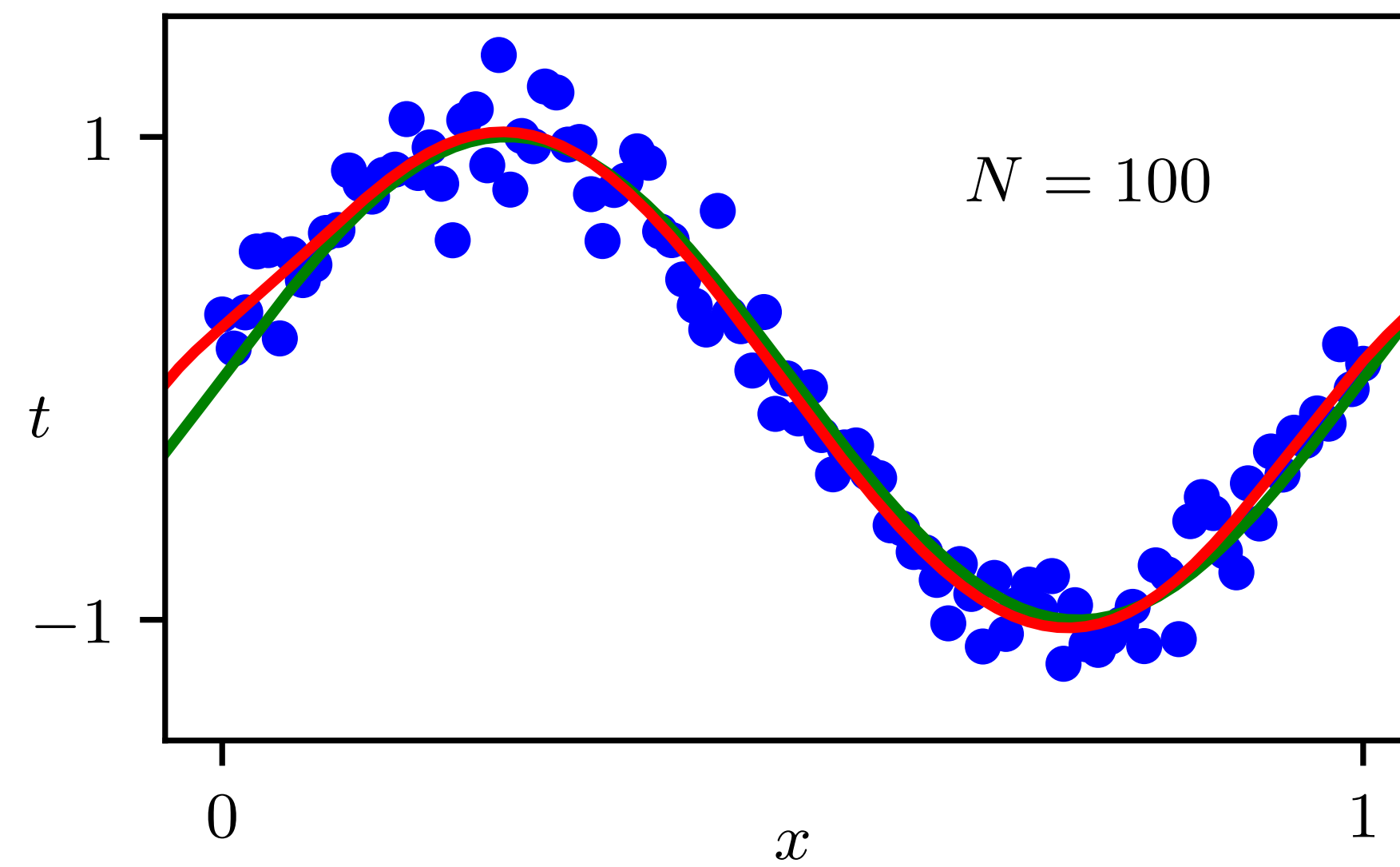
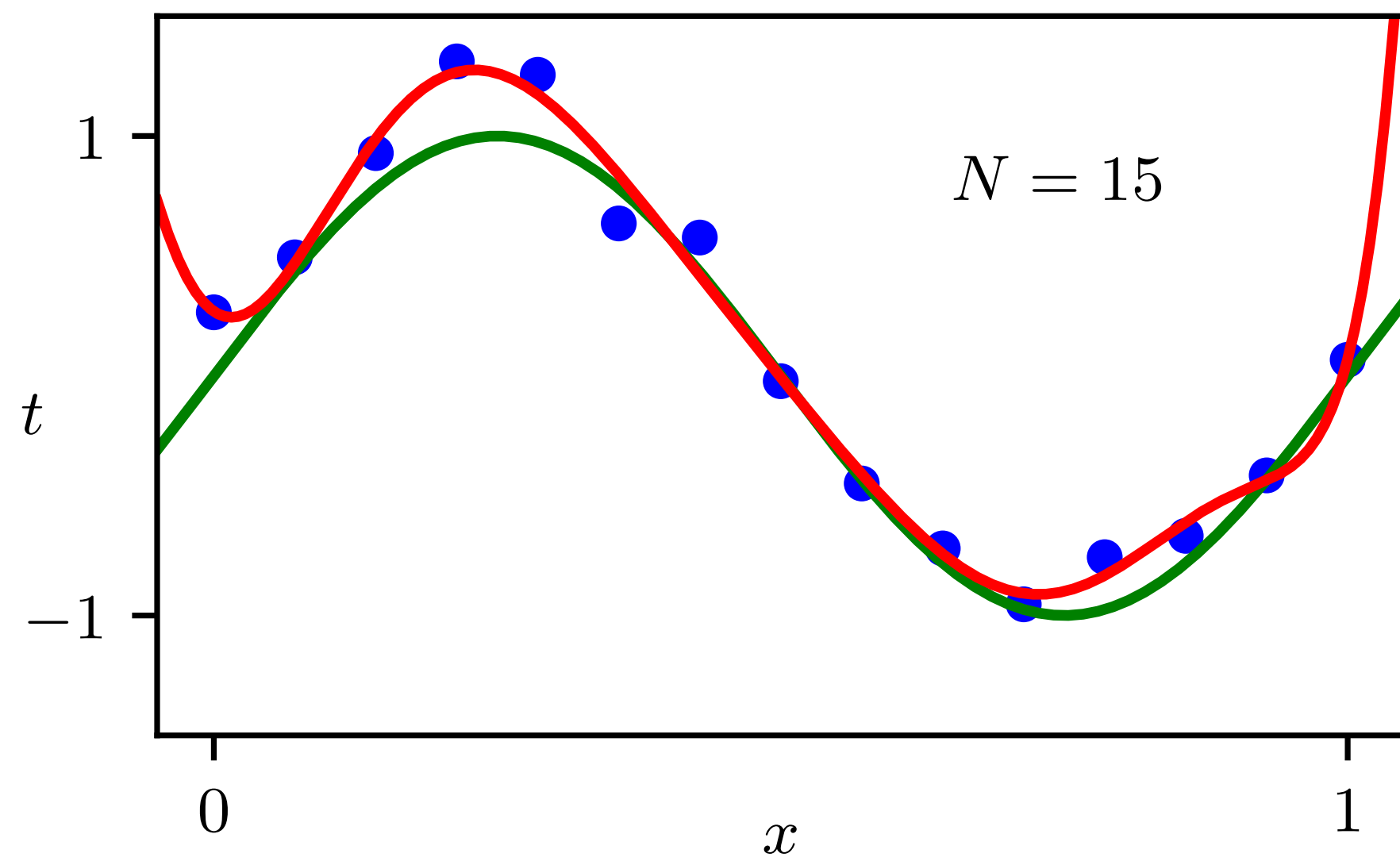


用多项式拟合  
10 个数据点

$M$  为多项式的次数

# 过拟合

- 避免过拟合的常用策略：
  - 增加训练数据量



$$M = 9$$

# 过拟合

- 避免过拟合的常用策略：
  - 增加训练数据量
  - 减少模型复杂度：选择更简单的模型或减少模型中的参数数量
  - 正则化（**Regularization**）：在模型的损失函数中添加一个惩罚项，限制模型的复杂度
  - 早停（**Early Stopping**）：在训练过程中，一旦在验证集上的性能开始下降，即停止训练，以防模型继续学习训练数据中的噪声

# 正则化

- 正则化 (Regularization) :

- 在模型的损失函数中添加一个惩罚项  $R(\theta)$ , 限制模型的复杂度:

$$L_R(\theta) = L(\theta) + \lambda R(\theta)$$

- 正则化通过  $R(\theta)$  阻止模型权重达到极大或极小的值, 而这些极值通常与训练数据上高度特定、非普遍性的模式相关
  - 正则化鼓励模型权重保持较小的值, 从而使模型的预测更加稳定, 不会对训练数据中的小波动做出过度反应

# L2 正则化

- 在 L2 正则化中,  $R(\theta)$  为  $\theta$  的 L2 范数 (L2 Norm) 的平方:

$$R(\theta) = \|\theta\|_2^2 = \sum_{j=1}^n \theta_j^2$$

- 使用 L2 正则化的逻辑回归损失函数:

$$L_R = \frac{1}{m} \left[ \sum_{i=1}^m L_{\text{CE}}(f(x^{(i)}; w), y^{(i)}) + \lambda \sum_{j=1}^n w_j^2 \right]$$

- 其中  $\lambda$  为模型的超参数

# L1 正则化

- 在 L1 正则化中,  $R(\theta)$  为  $\theta$  的 L1 范数 (L1 Norm) :

$$R(\theta) = \|\theta\|_1 = \sum_{j=1}^n |\theta_j|$$

- 使用 L1 正则化的逻辑回归损失函数:

$$L_R = \frac{1}{m} \left[ \sum_{i=1}^m L_{\text{CE}}(f(x^{(i)}; w), y^{(i)}) + \lambda \sum_{j=1}^n |w_j| \right]$$

- 其中  $\lambda$  为模型的超参数