1 课程简介

王中雷

厦门大学王亚南经济研究院和经济学院, 2025

内容摘要

1. 回顾已学内容

2. 问题

3. 学习目标

4. 交叉熵

回顾已学内容

1．数据集类型：

- 训练集（Training data）样本量n：$\{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}$
- 测试集（Test data）

2．回归类型：

- 线性回归：$y_i = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 + \epsilon_i \quad (i = 1, \ldots, n)$
- 逻辑回归：$\mathrm{logit} P(y_i = 1 \mid \boldsymbol{x}_i) = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 \quad (i = 1, \ldots, n)$
  - $\mathrm{logit} p = \log(p) - \log(1 - p), \quad \forall p \in (0, 1)$
- 多元逻辑回归 (Softmax regression)

符号

1. n：训练集样本量
2. d：特征向量维度
3. $\boldsymbol{X} \in \mathbb{R}^{n \times d}$：设计矩阵
   - $\boldsymbol{X} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n)^{\mathrm{T}}$
   - $\boldsymbol{x}_i \in \mathbb{R}^{d \times 1}$：第$i$个训练样本的特征向量
4. $\boldsymbol{Y} \in \mathbb{R}^{n \times 1}$：标签向量
   - $\boldsymbol{Y} = (y_1, \ldots, y_n)^{\mathrm{T}}$
   - $y_i \in \mathbb{R}$：第i个样本的标签

线性回归--模型设定

1. 训练集

- 特征向量，标签：$\boldsymbol{x}_i \in \mathbb{R}^{d\times 1}$, $y_i \in \mathbb{R}$ $\quad (i = 1, \ldots, n)$

2. 目标

- 学习 $\boldsymbol{x}$ 和 y 之间的关系

3. 真实模型（用于产生数据）

- $y_i = b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0 + \epsilon_i \quad (i = 1, \ldots, n)$
  - $\boldsymbol{\theta}_0 = (b_0, \boldsymbol{w}_0{}^{\mathrm{T}})^{\mathrm{T}}$：真实模型参数
  - ϵ_i：满足$E(\epsilon_i \mid \boldsymbol{x}_i) = 0$, 以及$\mathrm{var}(\epsilon_i \mid \boldsymbol{x}_i) = \sigma^2 > 0$ 的残差项

线性回归--模型设定

1．所提出模型（用于拟合数据）

- $E(y_i \mid \boldsymbol{x}_i) = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w} \quad (i = 1, \ldots, n)$
  - $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$：（提出）模型的参数
  - 与真实模型形式相同，但我们需要确定模型参数
  - 训练模型是指，估计所提出模型的参数

线性回归--参数估计

1．损失函数：对于第i 个训练样本，

- l_2-损失：$\mathcal{L}(y_i, \hat{y}_i) = (y_i - \hat{y}_i)^2$
- $\hat{y}_i = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}$
- $\hat{y}_i$ 是模型参数$\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$ 的函数，但为了简单起见，我们忽略模型参数
- $\mathcal{L}(y_i, \hat{y}_i)$ 实际上是模型参数$\boldsymbol{\theta}$ 的函数

2．代价函数

$$\mathcal{J}(\boldsymbol{\theta}) = n^{-1} \sum_{i=1}^{n} \mathcal{L}(y_i, \hat{y}_i) = n^{-1} \sum_{i=1}^{n} (y_i - \hat{y}_i)^2 = n^{-1} \sum_{i=1}^{n} (y_i - b - \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w})^2$$

3．我们通常不区分损失函数和代价函数

线性回归--参数估计

1．求解

$$\frac{\mathcal{J}}{\partial b} = 0, \quad \frac{\mathcal{J}}{\partial \boldsymbol{w}} = 0$$

2．问题：求和运算导致算法计算效率低下

向量化

1．考虑

$$\sum_{i=1}^{m} a_i b_i$$

- $a_i \in \mathbb{R}, \quad b_i \in \mathbb{R}$

2．对于 Python 或者其他软件，用 for 循环计算加和是非常低效的

3．然而，矩阵运算是高效的

4．因此，我们可对加进行向量化处理

向量化

1. 向量化结果

$$\sum_{i=1}^{m} a_i b_i = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{b}$$

- $\boldsymbol{a} = (a_1, \ldots, a_m)^{\mathrm{T}} \in \mathbb{R}^{m \times 1}$
- $\boldsymbol{b} = (b_1, \ldots, b_m)^{\mathrm{T}} \in \mathbb{R}^{m \times 1}$

向量化

1. 例 1：考虑

$$\sum_{i=1}^{m} a_i$$

- $a_i \in \mathbb{R}$

2. 向量化结果

$$\sum_{i=1}^{n} a_i = \sum_{i=1}^{n} a_i \times 1 = \boldsymbol{a}^{\mathrm{T}} \cdot \mathbf{1} (= \mathbf{1}^{\mathrm{T}} \cdot \boldsymbol{a})$$

- $\boldsymbol{a} = (a_1, \ldots, a_m)^{\mathrm{T}} \in \mathbb{R}^{m \times 1}$
- $\mathbf{1} = (1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^{m \times 1}$：由 1 组成的长度为 m 的列向量

向量化

1. 例 2：考虑

$$\sum_{i=1}^{m} a_i \boldsymbol{b}_i$$

- $a_i \in \mathbb{R}$：一个可能取值为 1 的标量
- $\boldsymbol{b}_i \in \mathbb{R}^{k\times 1}$

2. 向量化结果

$$\sum_{i=1}^{n} a_i \boldsymbol{b}_i = \boldsymbol{B} \cdot \boldsymbol{a}$$

- $\boldsymbol{a} = (a_1, \ldots, a_m)^{\mathrm{T}} \in \mathbb{R}^{m\times 1}$
- $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m) \in \mathbb{R}^{k\times m}$

向量化

1. 例 3：考虑

$$\sum_{i=1}^{m} a_i \boldsymbol{b}_i{}^{\mathrm{T}}$$

- $a_i \in \mathbb{R}$：一个可能取值为 1 的标量
- $\boldsymbol{b}_i \in \mathbb{R}^{k\times 1}$

2. 向量化结果

$$\sum_{i=1}^{n} a_i \boldsymbol{b}_i = \boldsymbol{a}^{\mathrm{T}} \cdot \boldsymbol{B}$$

- $\boldsymbol{a} = (a_1, \ldots, a_m)^{\mathrm{T}} \in \mathbb{R}^{m\times 1}$
- $\boldsymbol{B} = (\boldsymbol{b}_1, \ldots, \boldsymbol{b}_m)^{\mathrm{T}} \in \mathbb{R}^{m\times k}$

向量化

1. 例 4：考虑

$$\sum_{i=1}^{m} \boldsymbol{a}_i \boldsymbol{b}_i^{\mathrm{T}}$$

- $\boldsymbol{a}_i \in \mathbb{R}^{k\times 1}$
- $\boldsymbol{b}_i \in \mathbb{R}^{l\times 1}$

2. 向量化结果

$$\sum_{i=1}^{n} \boldsymbol{a}_i \boldsymbol{b}_i^{\mathrm{T}} = \boldsymbol{A}\cdot\boldsymbol{B}$$

- $\boldsymbol{A} = (\boldsymbol{a}_1, \ldots, \boldsymbol{a}_m) \in \mathbb{R}^{k\times m}$
- $\boldsymbol{B} = (\boldsymbol{b}_1, \cdots, \boldsymbol{b}_m)^{\mathrm{T}} \in \mathbb{R}^{m\times k}$

向量化

1. 记$\hat{\boldsymbol{Y}} = b\mathbf{1}_n + \boldsymbol{X}\boldsymbol{w}$

   - $\mathbf{1}_n = (1, \ldots, 1)^{\mathrm{T}} \in \mathbb{R}^{n\times 1}$：由 1 组成的长度为$n$ 的向量

2. 代价函数

$$\begin{aligned}\mathcal{J}(\boldsymbol{\theta}) &= n^{-1}\sum_{i=1}^{n}(y_i - \hat{y}_i)^2 = n^{-1}(\boldsymbol{Y} - \hat{\boldsymbol{Y}})^{\mathrm{T}}(\boldsymbol{Y} - \hat{\boldsymbol{Y}}) \\ &= n^{-1}(\boldsymbol{Y} - b\mathbf{1}_n - \boldsymbol{X}\boldsymbol{w})^{\mathrm{T}}(\boldsymbol{Y} - b\mathbf{1}_n - \boldsymbol{X}\boldsymbol{w}) \\ &= n^{-1}(\boldsymbol{Y} - \tilde{\boldsymbol{X}}\boldsymbol{\theta})^{\mathrm{T}}(\boldsymbol{Y} - \tilde{\boldsymbol{X}}\boldsymbol{\theta})\end{aligned}$$

   - $\tilde{\boldsymbol{X}} = (\tilde{\boldsymbol{x}}_1, \ldots, \tilde{\boldsymbol{x}}_n)^{\mathrm{T}}$ (将b 和$\boldsymbol{w}$ 放在一起考虑)
   - $\tilde{\boldsymbol{x}}_i = (1, \boldsymbol{x}_i^{\mathrm{T}})^{\mathrm{T}}$

向量化

1．考虑

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}} = 0$$

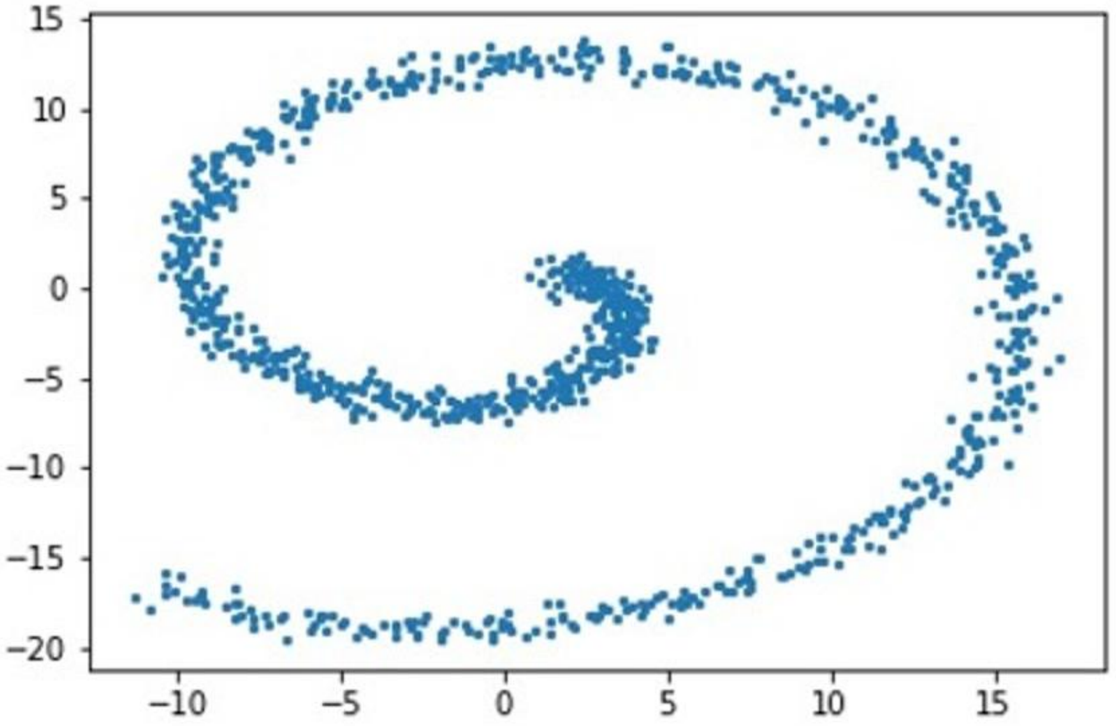
2．得到正则方程（normal equation）

$$\tilde{\boldsymbol{X}}^{\mathrm{T}} \tilde{\boldsymbol{X}} \boldsymbol{\theta} = \tilde{\boldsymbol{X}}^{\mathrm{T}} \boldsymbol{Y}$$

3．解 (显式解)

$$\hat{\boldsymbol{\theta}} = (\tilde{\boldsymbol{X}}^{\mathrm{T}} \tilde{\boldsymbol{X}})^{-1} \tilde{\boldsymbol{X}}^{\mathrm{T}} \boldsymbol{Y}$$

例子



逻辑回归--模型设定

1．训练集

- 特征向量，标签：$\boldsymbol{x}_i \in \mathbb{R}^{d\times 1}$，$y_i \in \{0,1\}$　$(i=1,\ldots,n)$

2．目标

- 学习 $\boldsymbol{x}$ 和 y 之间的关系

3．真实模型（用于产生数据）

$$E(y_i \mid \boldsymbol{x}_i) = P(y_i = 1 \mid \boldsymbol{x}_i) = \sigma(b_0 + \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w}_0) \quad (i=1,\ldots,n)$$

- $\sigma(z) = \{1+\exp(-z)\}^{-1}$ (我们为什么需要这步变换?)

逻辑回归--模型设定

1．所提出模型（用于拟合数据）

- $P(y_i = 1 \mid \boldsymbol{x}_i) = \sigma(b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}) \quad (i = 1, \ldots, n)$
  - $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$：（提出）模型的参数
  - 与真实模型形式相同，但我们需要确定模型参数
  - 训练模型是指，估计所提出模型的参数

逻辑回归--参数估计

1．损失函数：对于第i 个训练样本，

- 交叉熵（负对数似然函数）

$$\mathcal{L}(y_i, a_i) = -\{y_i \log a_i + (1 - y_i) \log(1 - a_i)\}$$

  - $a_i = \sigma(z_i)$ （估计概率）
  - $z_i = b + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}$

- a_i 是模型参数$\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$的函数，但简单起见，我们忽略该模型参数
- 交叉熵实际上是模型参数$\boldsymbol{\theta}$ 的函数

逻辑回归--参数估计

1．代价函数

$$\mathcal{J}(\boldsymbol{\theta}) = n^{-1}\sum_{i=1}^{n}\mathcal{L}\{y_i, a_i\}$$

$$= -n^{-1}\sum_{i=1}^{n}\{y_i \log a_i + (1-y_i)\log\{1-a_i\}\}$$

- $\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$
- $a_i = \sigma(b + \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w})$

逻辑回归--参数估计

1．求解

$$\frac{\mathcal{J}}{\partial \boldsymbol{\theta}} = 0$$

2．没有显式解

牛顿-拉裴森（Newton-Raphson）算法

步骤1. 随机初始化$\boldsymbol{\theta}^{(0)}$

步骤2. 基于$\boldsymbol{\theta}^{(t)}$ 得到

$$\nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)=\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}(\boldsymbol{\theta}^{(t)})$$

$$H(\mathcal{J})\left(\boldsymbol{\theta}^{(t)}\right)=\frac{\partial^2 \mathcal{J}}{\partial \boldsymbol{\theta} \partial \boldsymbol{\theta}^{\mathrm{T}}}(\boldsymbol{\theta}^{(t)})$$

步骤3. 更新模型参数

$$\boldsymbol{\theta}^{(t+1)}=\boldsymbol{\theta}^{(t)}-\left[H(\mathcal{J})\left(\boldsymbol{\theta}^{(t)}\right)\right]^{-1} \nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)$$

步骤4. 回到步骤 2 直至收敛

讨论

1. 收敛速度快（得益于 Hessian 矩阵的使用）
2. 计算效率低下（涉及到对 Hessian 矩阵求逆）
3. 适用于当模型参数规模较小的情况
4. 不适用于深度学习模型参数的训练
5. 什么算法适用于深度学习模型的训练呢？

批量梯度下降（Batch gradient descent）法

步骤1. 随机初始化$\boldsymbol{\theta}^{(0)}$

步骤2. 基于$\boldsymbol{\theta}^{(t)}$ 计算

$$\nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)=\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}\left(\boldsymbol{\theta}^{(t)}\right)$$

步骤3. 更新模型参数

$$\boldsymbol{\theta}^{(t+1)}=\boldsymbol{\theta}^{(t)}-\alpha \nabla \mathcal{J}\left(\boldsymbol{\theta}^{(t)}\right)$$

步骤4. 回到步骤 2 直至收敛

批量梯度下降（Batch gradient descent）法

1．α：学习率（learning rate）

- 控制算法收敛速度
- 影响训练模型的好坏
- 将在后续学习中展开讨论

算法比较

1. 牛顿-拉裴森算法

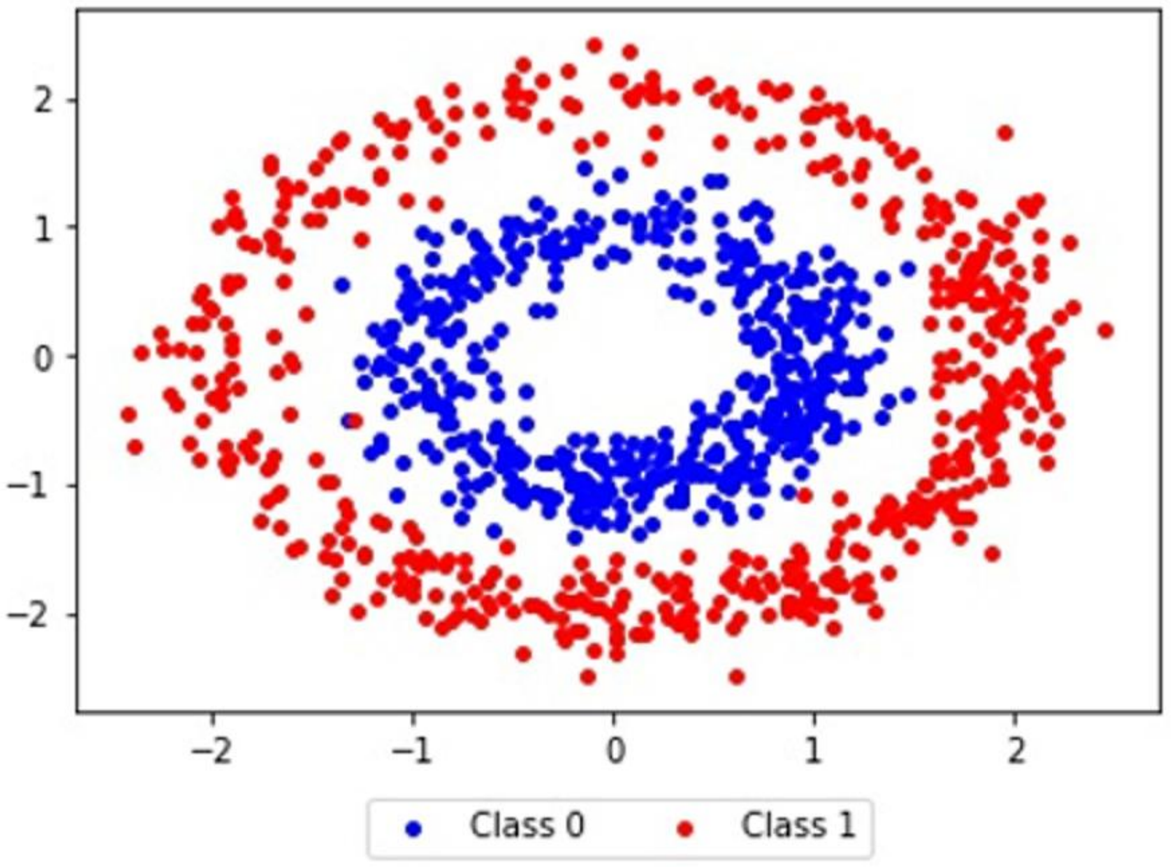
- 优势:
  - 收敛速度快
  - 算法精度高
- 缺点:
  - 计算效率低（Hessian 矩阵求逆）
  - 稳定性差（当 Hessian 矩阵出现病态时）

算法比较

1．批量梯度下降法

- 优势：
  - 计算效率高
  - 便于操作
- 缺点：
  - 收敛速度较慢
  - 对学习率较为敏感

例子



总结

1．我们通常利用如下步骤进行数据建模

问题	模型	损失	算法
$y \in \mathbb{R}$	$E(y \mid \boldsymbol{x}) = b + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{w}$	$n^{-1}\sum_{i=1}^{n}(y_i - b - \boldsymbol{x}_i^{\mathrm{T}}\boldsymbol{w})^2$	正则方程
$y \in \{0, 1\}$	$E(y \mid \boldsymbol{x}) = \sigma(b + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{w})$	$-n^{-1}\sum_{i=1}^{n}\{y_i \log a_i + (1 - y_i)\log(1 - a_i)\}$	梯度下降

2．梯度下降法广泛应用于人工智能模型中

3．梯度下降法的关键步骤是计算

$$\frac{\partial \mathcal{J}}{\partial \boldsymbol{\theta}}$$

问题

1. 目前为止，我们假设真实模型和所提出模型具有相同的形式

2. 然而，这种假设在现实生活中通常不成立

3. 模型错误设定

   - 被提出的（统计）模型过于简单
   - 然而，真实模型往往很复杂

学习目标

1. 全连接神经网络（多层感知机）
2. 卷积神经网络（CNN, ...）
3. 序列模型（RNN, LSTM, transformers, ...）
4. （时间允许情况下的）其他话题（GNN, GCN, ...）

学习目标

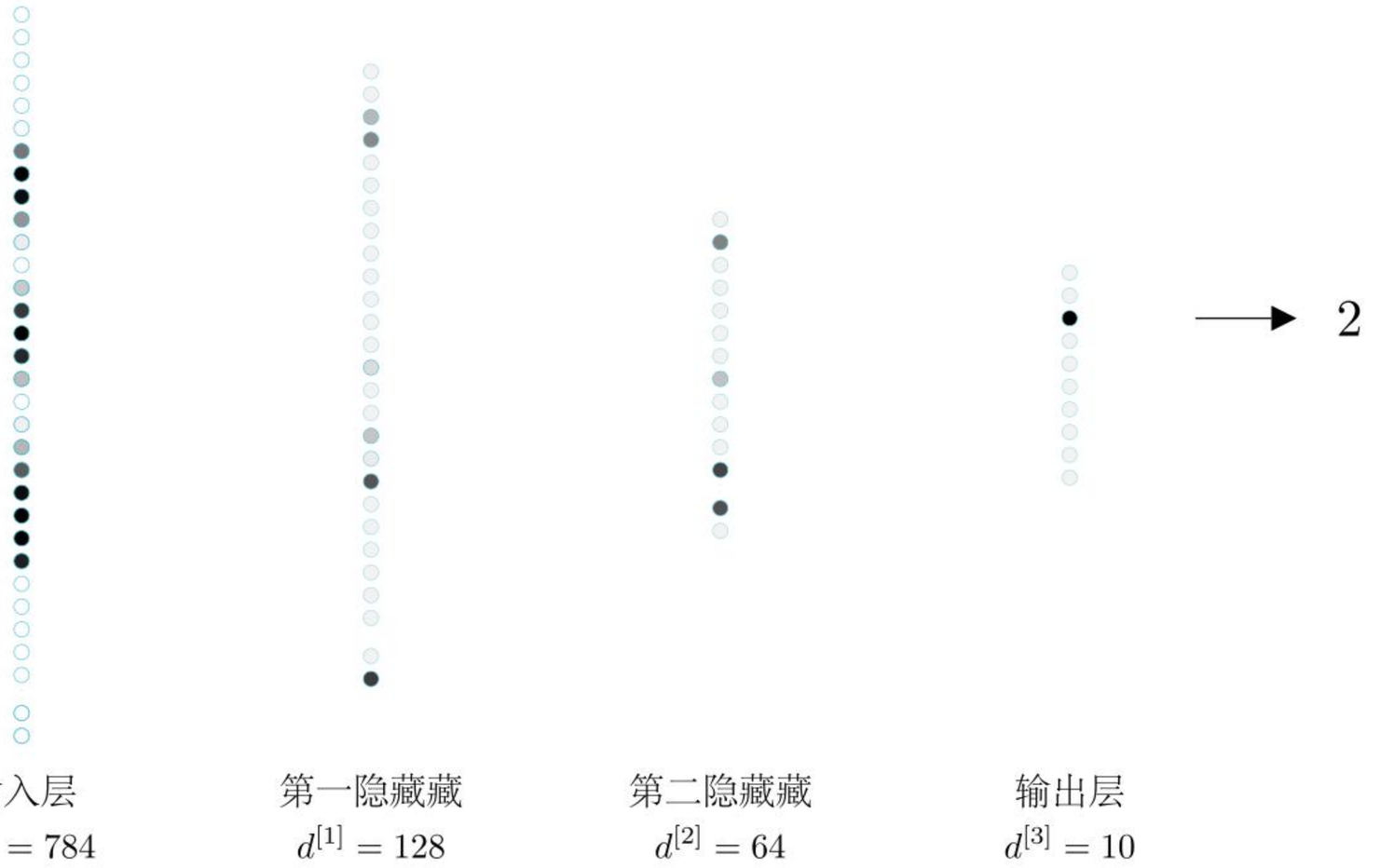
1. 全连接神经网络（多层感知机）
2. 卷积神经网络（CNN, ...）
3. 序列模型（RNN, LSTM, transformers, ...）
4. （时间允许情况下的）其他话题（GNN, GCN, ...）

全连接神经网络例子

测试图片

模型 (具有两个隐藏藏的全连接神经网络)

估计结果

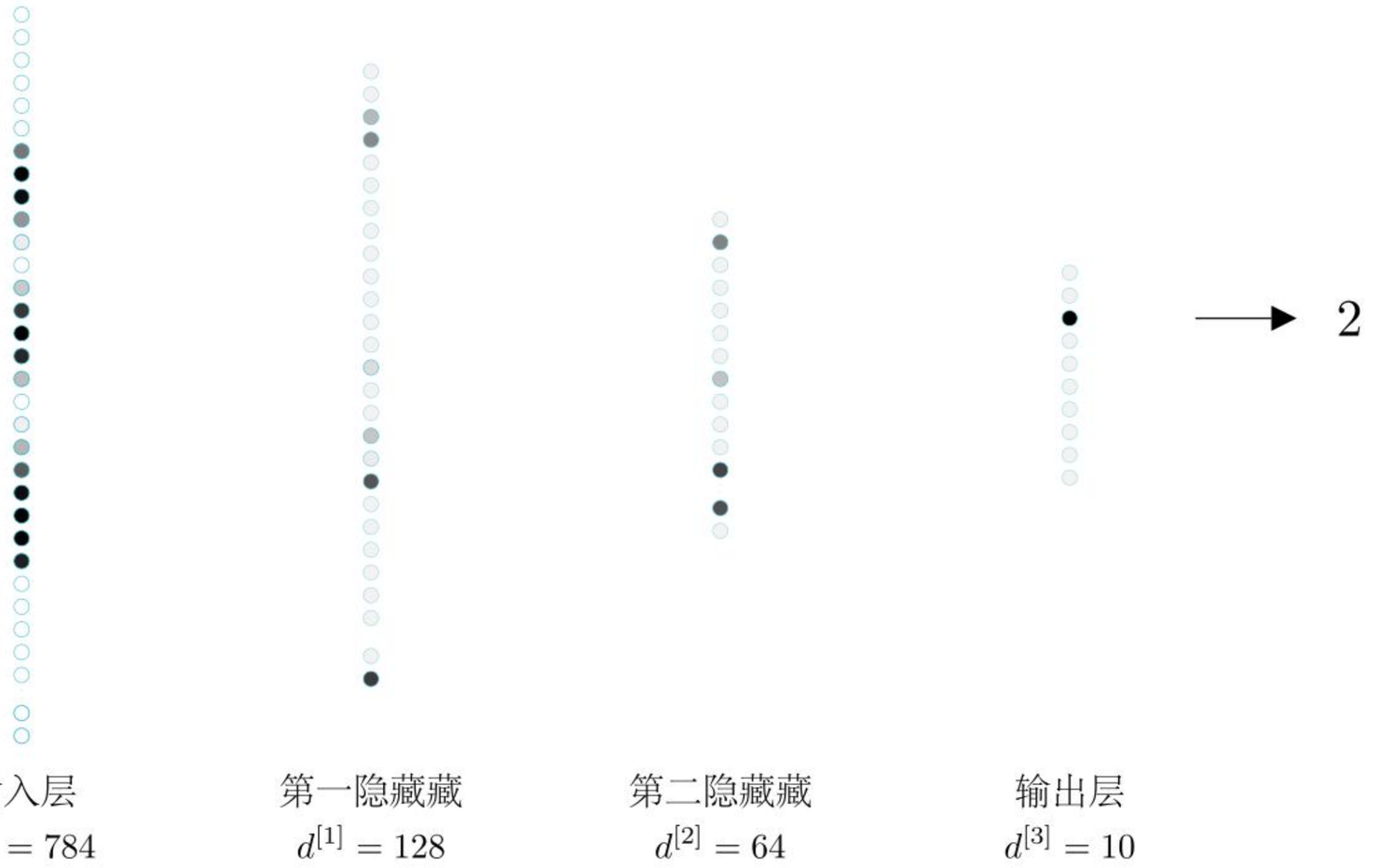


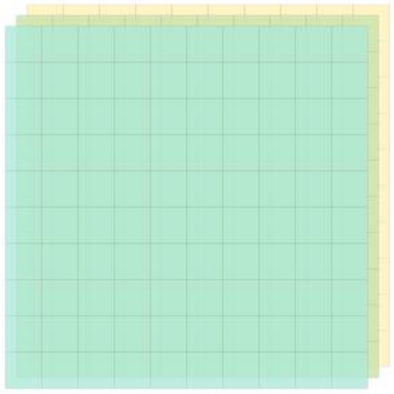
全连接神经网络例子

测试图片

模型 (具有两个隐藏藏的全连接神经网络)

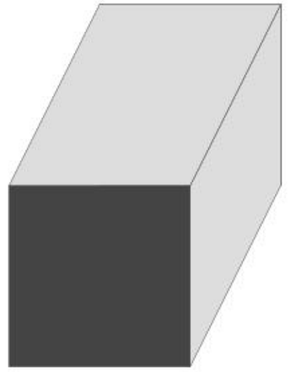
估计结果





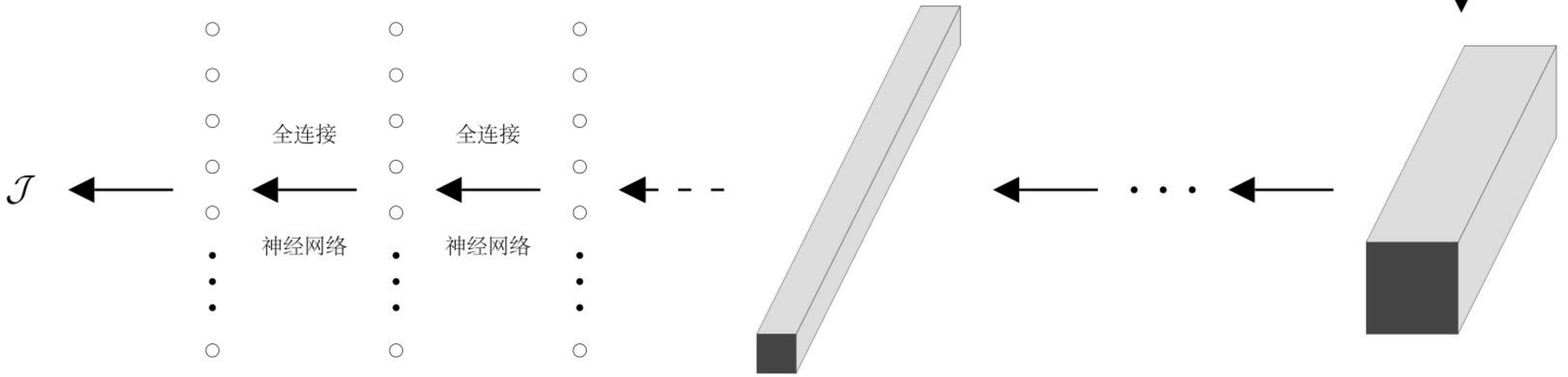
卷积

池化



输入($\mathbb{R}^{d_H \times d_W \times d_C}$)

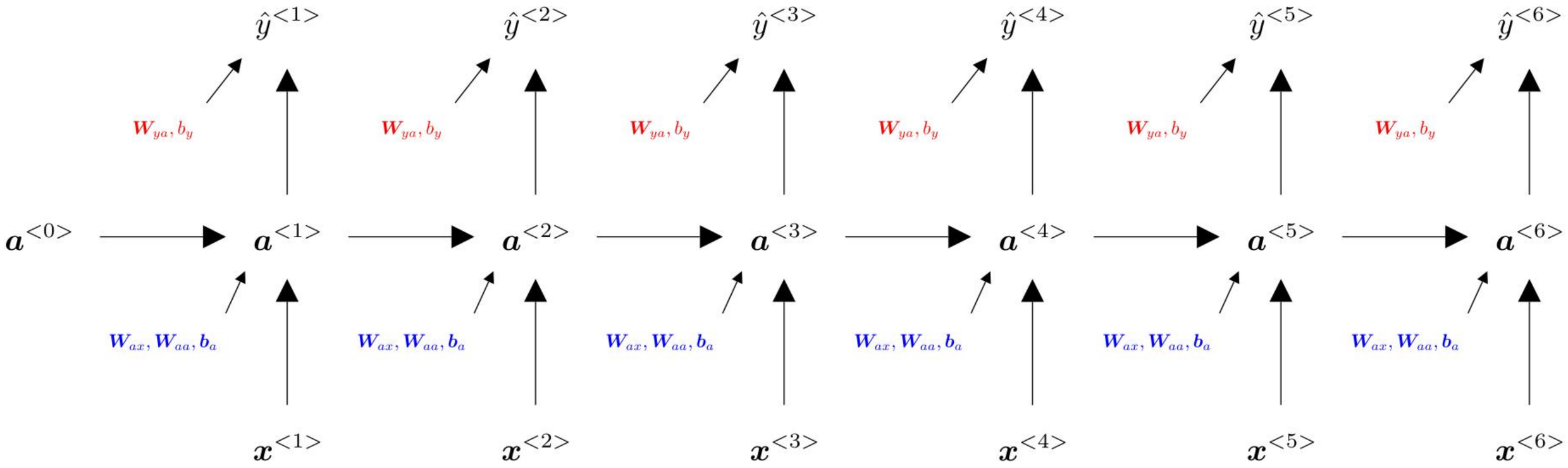
后续讨论



$\boldsymbol{A}^{[L]} \in \mathbb{R}^{d^{[L]} \times 1 \times 1}$

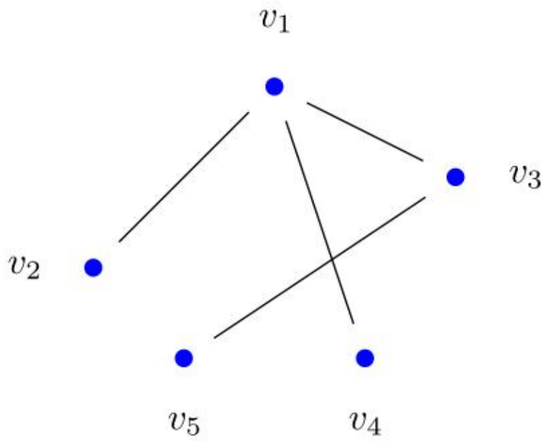
循环神经网络模块

$$\hat{y}^{<i>} = \sigma_{ya}(\boldsymbol{W}_{ya}\boldsymbol{a}^{<i>} + b_y) \quad (i = 1, \ldots, 6)$$



$$\boldsymbol{a}^{<i>} = \sigma_{ax}(\boldsymbol{W}_{ax}\boldsymbol{x}^{<i>} + \boldsymbol{W}_{aa}\boldsymbol{a}^{<i-1>} + \boldsymbol{b}_a) \quad (i = 1, \ldots, 6)$$

无向图



顶点集$\mathcal{V} = \{v_i : i = 1, \ldots, n\}$

边集: $\mathcal{E}$

无向图: $\mathcal{G} = \mathcal{V} \cup \mathcal{E}$

邻接矩阵（{0,1}; [0,1]）

$$\begin{pmatrix} 0 & 1 & 1 & 1 & 0 \\ 1 & 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 & 1 \\ 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 1 & 0 & 0 \end{pmatrix}$$

度矩阵（对角阵, 邻接矩阵A的每行求和）

$$D = \begin{pmatrix} 3 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 2 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$$

交叉熵

1. 一个事实：在一般条件下，

$$E_{X\sim P}\{\log p(X)\} \geq E_{X\sim P}\{\log q(X)\}$$

- $X \sim P$: 随机变量X 产生于分布P
- $p(x)$: 分布P 的概率密度函数
- $q(x)$: 其他任意概率密度函数

2. 对于两个密度函数的p 和q 的交叉熵

$$H(p,q) = -E_{X\sim P}\{\log q(X)\} = -E\{\log q(X)\}$$

- 衡量概率密度函数q 对随机变量X 分布的近似程度

交叉熵

1. 只观测到随机样本$\{x_1, \ldots, x_n\}$

2. 交叉熵$H(p, q)$ 可被如下公式近似

$$\hat{H}(p, q) = -\frac{1}{n} \sum_{i=1}^{n} \log q(x_i)$$

   - 以上是经验密度函数$P_n(x) = n^{-1} \sum_{i=1}^{n} \delta(x = x_i)$ 和q 的交叉熵

3. 目标: 寻找密度函数q 用来最小化$\hat{H}(p, q)$
   - 即为了近似随机变量的密度函数

回顾逻辑回归

1. 假设随机变量$\{(\boldsymbol{x}_i, y_i) : i = 1, \ldots, n\}$ 是独立同分布的，对应的概率密度函数为

$$p(\boldsymbol{x}, y) = p(\boldsymbol{x})p(y \mid \boldsymbol{x})$$

- $p(\boldsymbol{x})$：（不是很重要的）边际概率密度函数$\boldsymbol{x}$
- $p(y \mid \boldsymbol{x}) = \sigma(b_0 + \boldsymbol{x}_i^{\mathrm{T}} \boldsymbol{w}_0)$：关于$y$ 的（我们感兴趣的）参数化的条件密度函数

2. 目标: 寻找概率密度函数$q(\boldsymbol{x}, y)$ 以最小化

$$\hat{H}(p, q) = -\frac{1}{n} \sum_{i=1}^{n} \log q(\boldsymbol{x}_i, y_i)$$

回顾逻辑回归

1. 一些数学推到

$$-\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i, y_i) = -\frac{1}{n}\sum_{i=1}^{n}\{\log q(\boldsymbol{x}_i) + \log q(y_i \mid \boldsymbol{x}_i)\}$$

$$= -\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

- $q(\boldsymbol{x}, y) = q(\boldsymbol{x})q(y \mid \boldsymbol{x})$
- $q(\boldsymbol{x})$：不假定具体形式
- $q(y \mid \boldsymbol{x}) = a(\boldsymbol{\theta})^y\{1-a(\boldsymbol{\theta})\}^{1-y}$
- $a(\boldsymbol{\theta}) = \sigma(b + \boldsymbol{x}^{\mathrm{T}}\boldsymbol{w})$：模型参数为$\boldsymbol{\theta} = (b, \boldsymbol{w}^{\mathrm{T}})^{\mathrm{T}}$

回顾逻辑回归

1．因此，我们有

$$-\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i, y_i) = -\frac{1}{n}\sum_{i=1}^{n}\log q(\boldsymbol{x}_i) - \frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

- 其中，我们对第一部分不感兴趣
- 我们对第二部分中的模型参数感兴趣

2．因此，等价于我们需要最小化下面的式子

$$-\frac{1}{n}\sum_{i=1}^{n}[y_i \log a_i(\boldsymbol{\theta}) + (1-y_i)\log\{1-a_i(\boldsymbol{\theta})\}]$$

- 以上便是从交叉熵出发，推导出的逻辑回归模型的代价函数