逻辑回归
（logistic regression）

程煦

xcheng8@njust.edu.cn

计算机科学与工程学院

Outline

- 逻辑回归模型
- 交叉熵
- 最大似然估计
- 随机梯度下降
- 多分类问题

监督学习（supervised learning）

- 回归

输出是数值
（连续值）

Share Price
“$ 24.50”

Continuous Labels
Regression

- 分类

Feature Space $\mathcal{X}$

Label Space $\mathcal{Y}$

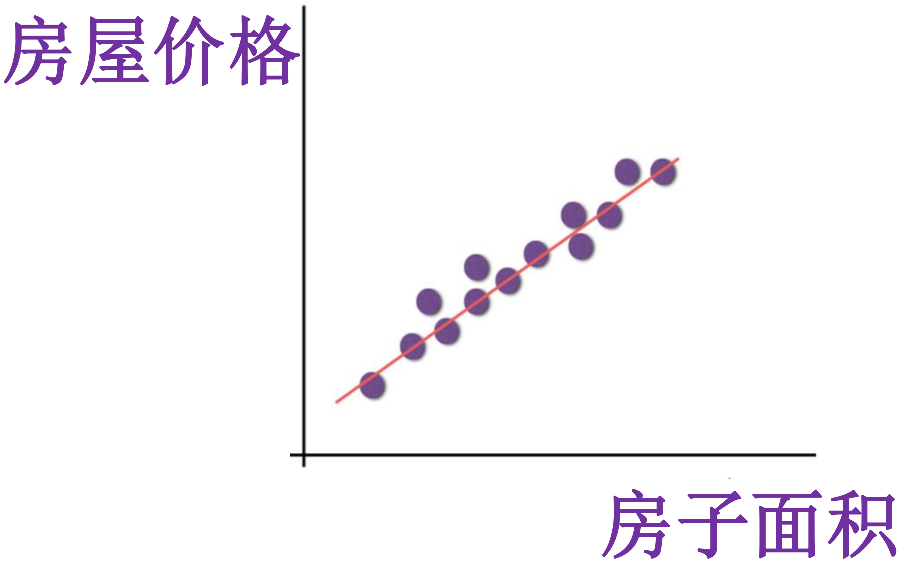
“Sports”
“News”
“Science”
…

输出是类别
（离散值）

Discrete Labels
Classification

线性回归（linear regression）

面积 (m^2)	房屋价格（万元）
123	250
150	320
87	160
102	220
…	…



$$y = \mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \mathrm{w}_3 x_3 + \cdots + \mathrm{w}_d x_d + b$$

$$y = \boldsymbol{w}^T \boldsymbol{x}$$

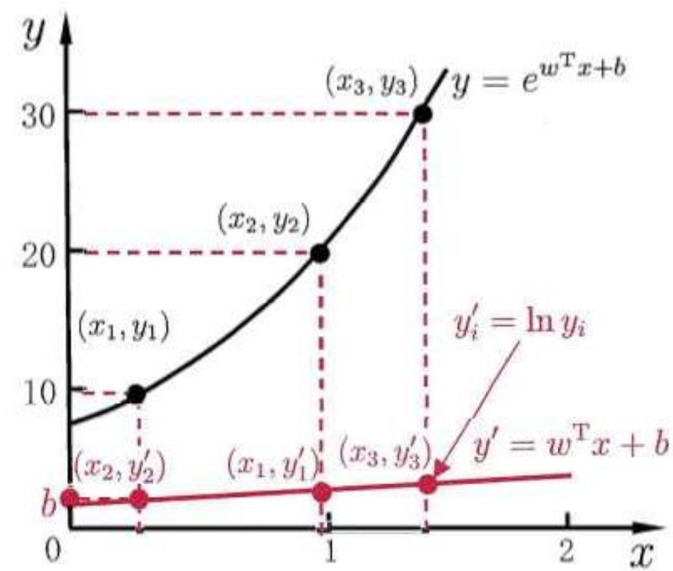
对数线性回归

- 形式：

$$\hat{y} = e^{\mathrm{w}_1 x_1 + \mathrm{w}_2 x_2 + \mathrm{w}_3 x_3 + \cdots + \mathrm{w}_d x_d + b} = e^{\boldsymbol{w}^T \boldsymbol{x}}$$

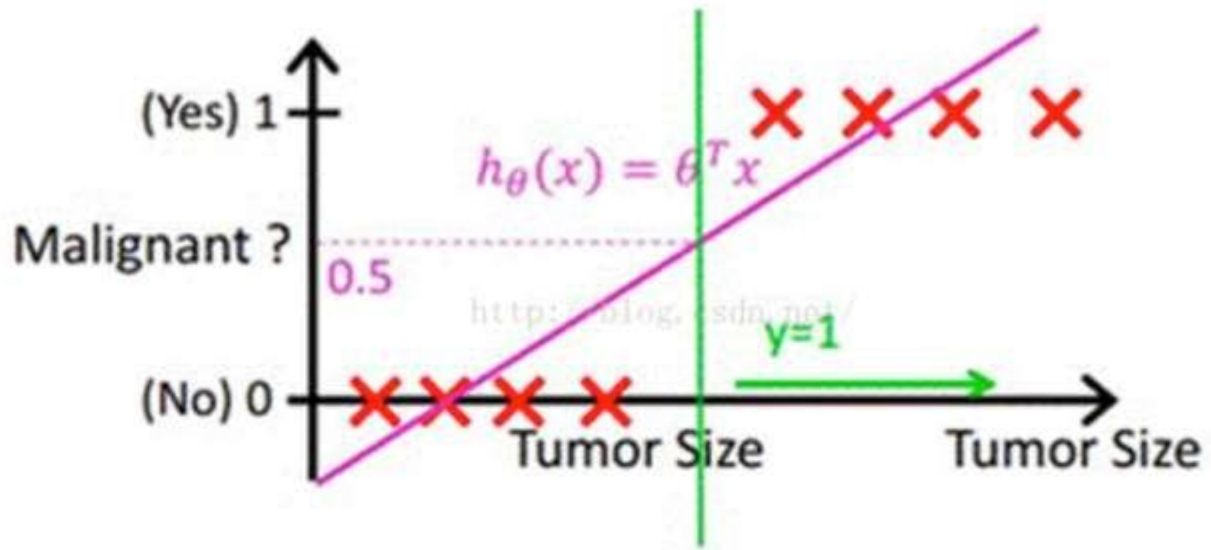
$$\Rightarrow \quad \ln \hat{y} = \boldsymbol{w}^T \boldsymbol{x}$$

- 将输出y的对数作为线性模型逼近的目标，实质上求取输入空间到输出空间的非线性映射。



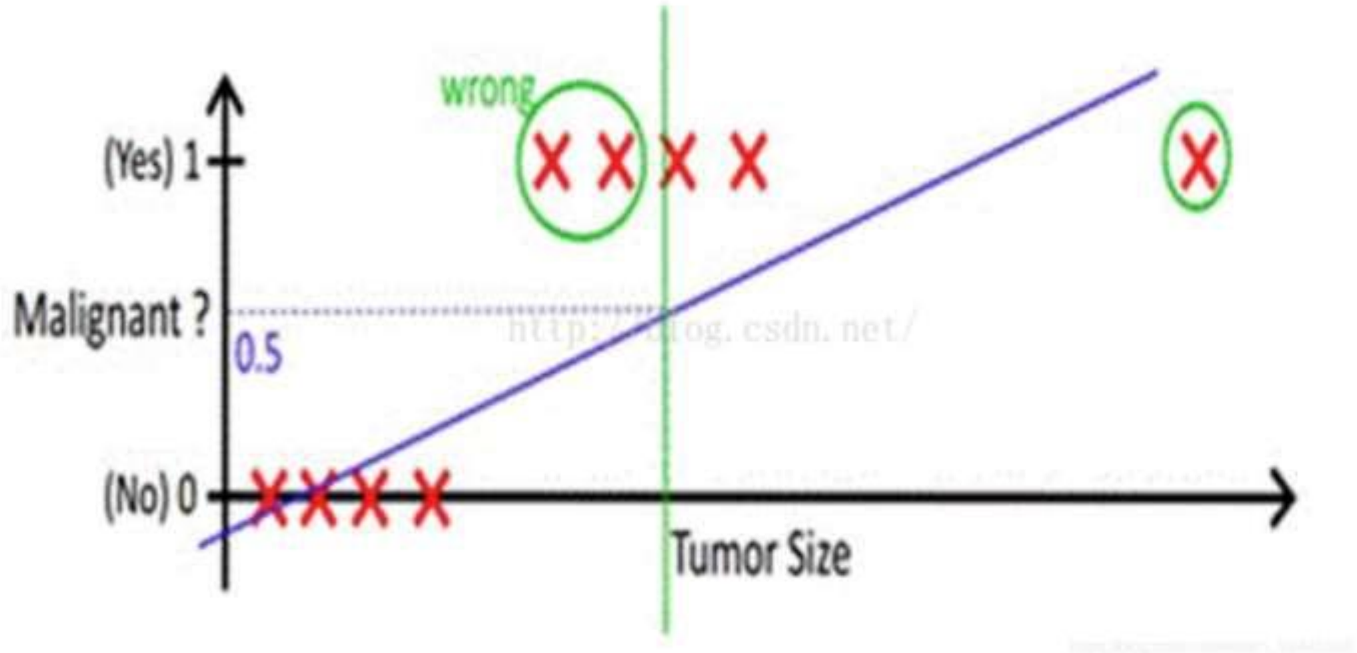
线性回归→二分类

- 通过回归任务来预测人体内肿瘤的大小，取一个平均值作为阈值，假如平均值为y，肿瘤大小超过y为恶性肿瘤，无肿瘤或大小小于y的，为非恶性。这样通过线性回归加设定阈值的办法，就可以完成一个简单的二分类任务。



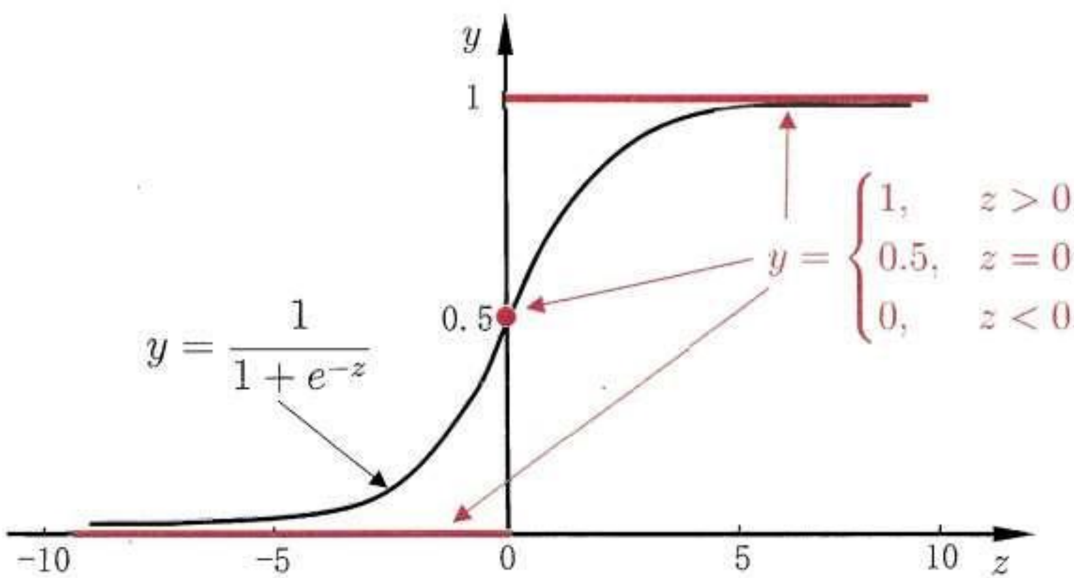
线性回归→二分类

- 如果有一个超大的肿瘤在我们的例子中，阈值就很难设定。若还是取平均值为阈值（阈值变大），则会出现下图的情况：漏诊！



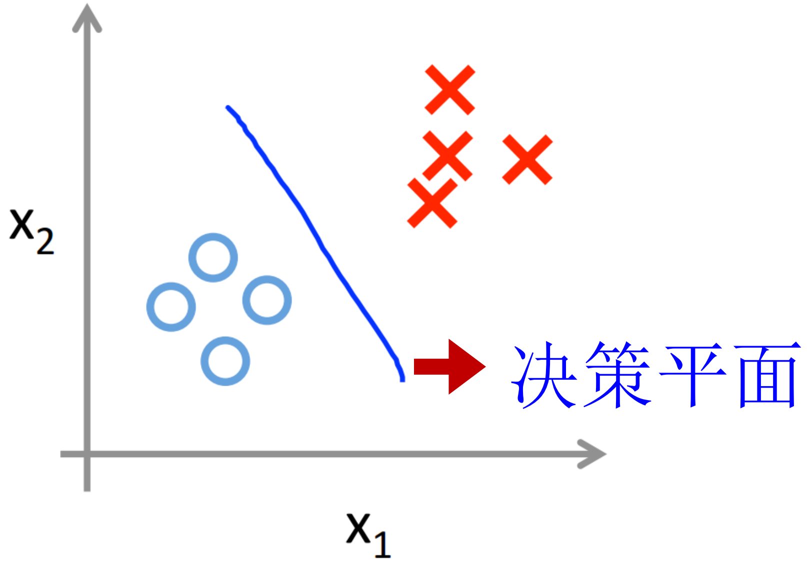
线性回归→二分类

- 1-利用线性回归的办法来拟合然后设置阈值的办法容易受到离群值的影响。
- 2-解决离群点问题：需要将线性回归产生的预测值从整个实数域压缩到 (0,1)，变成概率的形式（模型有多大的可能性将样本分到类别为1），可通过sigmoid函数实现



逻辑斯蒂回归（Logistic Regression）

- 是一种分类模型，尽管它被称作“回归”。
- 是一种二分类模型。
- 是一种线性分类模型，有一个线性决策面（超平面），但用一个非线性激活函数（Sigmoid函数）来计算分类概率

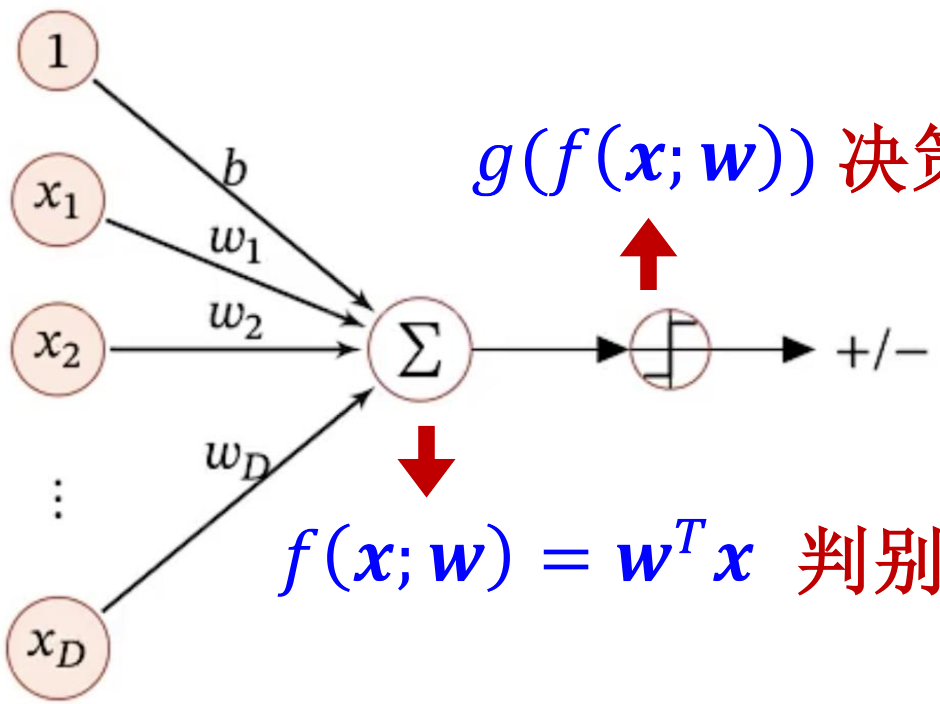


逻辑斯蒂回归（Logistic Regression）

■ 基本结构

$$g(f(\boldsymbol{x};\boldsymbol{w})) = \begin{cases} 1 & \text{if} \quad f(\boldsymbol{x};\boldsymbol{w}) > 0 \\ 0 & \text{if} \quad f(\boldsymbol{x};\boldsymbol{w}) < 0 \end{cases}$$

输出是0和1
两个类别值！



$g(f(\boldsymbol{x};\boldsymbol{w}))$ 决策函数

$f(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x}$ 判别函数，提取特征

如何构建损失函数？

逻辑斯蒂回归（Logistic Regression）

- 将分类问题看作条件概率估计问题
  - 引入非线性函数g来预测类别标签的条件概率$p(y=c|x)$
  - 以二分类为例：

$$p(y=1|\boldsymbol{x}) = g(f(\boldsymbol{x};\boldsymbol{w}))$$

$$p(y=0|\boldsymbol{x}) = 1 - p(y=1|\boldsymbol{x})$$

  - 判别函数f：线性函数$f(\boldsymbol{x};\boldsymbol{w}) = \boldsymbol{w}^T\boldsymbol{x} \in \mathbb{R}$
  - 决策函数g：把线性函数的值域从实数区间“挤压”到[0,1]之间来表示概率

如何构建函数 g ？

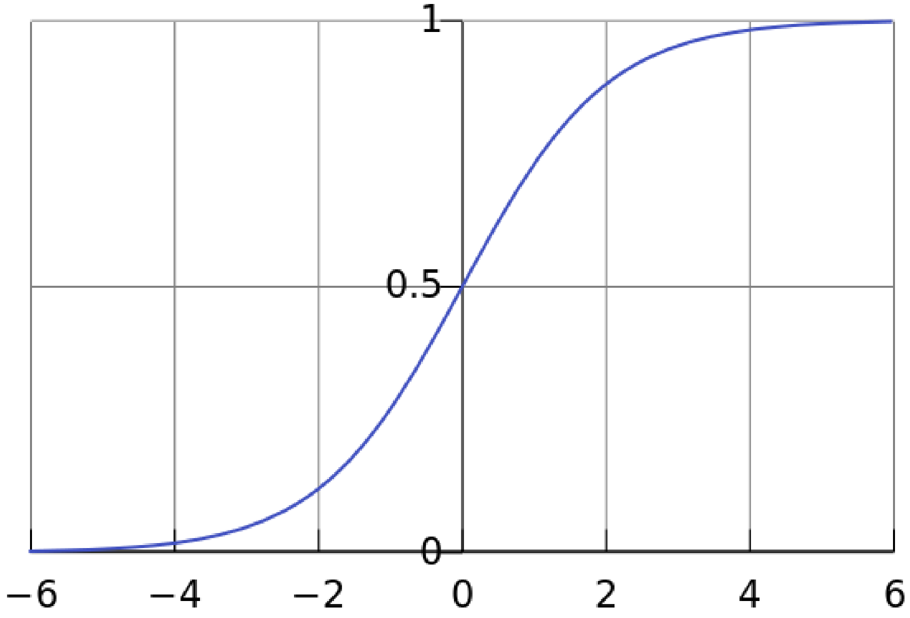
逻辑斯蒂回归（Logistic Regression）

逻辑斯蒂回归模型

除了sigmoid函数，还有什么函数$g: \mathbb{R} \rightarrow [0,1]$?

- Sigmoid 函数

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



- Logistic 回归

$$p(y = 1|\boldsymbol{x}) = g\big(f(\boldsymbol{x};\boldsymbol{w})\big) = \sigma(\boldsymbol{w}^T\boldsymbol{x}) = \frac{1}{1 + \exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

$$p(y = 0|\boldsymbol{x}) = 1 - p(y = 1|\boldsymbol{x})$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 模型预测的条件概率$p(y|\boldsymbol{x})$

$$p(y=1|\boldsymbol{x})=\sigma(\boldsymbol{w}^T\boldsymbol{x})=\frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

  - 真实条件概率$p_r(y|x)$
    - 对于一个样本(x,y)，其真实条件概率为

$y=0$	$p_r(y=1\|\boldsymbol{x})=0$ $p_r(y=0\|\boldsymbol{x})=1$
$y=1$	$p_r(y=1\|\boldsymbol{x})=1$ $p_r(y=0\|\boldsymbol{x})=0$

$$p_r(y=1|\boldsymbol{x})=y$$
$$p_r(y=0|\boldsymbol{x})=1-y$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 如何衡量两个条件分布的差异？

- 交叉熵能够衡量同一个随机变量中的两个不同概率分布的差异程度。
- 在机器学习，交叉熵衡量真实概率分布p与预测概率分布之间的q差异，即用预测分布q编码真实数据p所需的平均比特数

$$H(p,q) = -\sum_i p(y_i)\log q(y_i).$$

- 交叉熵值越小，说明预测分布q越接近真实分布p

逻辑斯蒂回归（Logistic Regression）

■ 交叉熵损失（cross-entropy loss）

$$H(p,q) = -\sum_i p(y_i)\log q(y_i).$$

- 给定一张输入图片，真实类别是“猫”，即$p(y|x) = [1,0,0]$，模型的预测分布$q(y|x) = [0.7,0.2,0.1]$

$$H(p,q) = -(1\cdot\log 0.7 + 0\cdot\log 0.2 + 0\cdot\log 0.1) = -\log 0.7 = 0.154.$$

- 如果模型对正确类别的预测概率$q(y = 猫|x)$很高，交叉熵损失就低，例如$q(y|x) = [1,0,0] \rightarrow H(p,q) = 0$。
- 如果模型错得离谱，交叉熵损失会变大，

$$q(y) = [0.1, 0.3, 0.6] \qquad H(p,q) = -\log 0.1 = 1.0$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 如何衡量两个条件分布的差异？

交叉熵：$H(p_r, p) = -(\boxed{y} log \boxed{\hat{y}} + (1-y)\log(1-\hat{y}))$

$$p_r(y=1|\boldsymbol{x}) = y \qquad p(y=1|\boldsymbol{x}) = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

- 利用梯度下降算法学习模型参数$\boldsymbol{w}$
  - 给定训练集：$\{(\boldsymbol{x}_i, y_i)\}$，$i \in \{1,2,\cdots N\}$
  - 交叉熵损失函数，模型在训练集的风险函数为

$$R(\boldsymbol{w}) = -\frac{1}{N}\sum_{i=1}^{N}(y_i log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i))$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 利用梯度下降（Gradient Descent）学习模型参数
    - 梯度为

$$\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\frac{1}{N}\sum_{i=1}^{N}[\frac{\partial}{\partial \boldsymbol{w}}(y_i log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i))]$$

$$\hat{y}_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

✓ 第一项: $\frac{\partial}{\partial \boldsymbol{w}} y_i log\hat{y}_i = y_i \cdot \frac{1}{\hat{y}_i} \cdot \frac{\partial \hat{y}_i}{\partial \boldsymbol{w}}$

$$= y_i \cdot \frac{1}{\hat{y}_i} \cdot \hat{y}_i(1-\hat{y}_i)\boldsymbol{x}_i$$

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

$$= y_i \cdot (1-\hat{y}_i) \cdot \boldsymbol{x}_i$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 利用梯度下降（Gradient Descent）学习模型参数
    - 梯度为

$$\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\frac{1}{N}\sum_{i=1}^{N}[\frac{\partial}{\partial \boldsymbol{w}}(y_i log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i))]$$

✓ 第二项: $\frac{\partial}{\partial \boldsymbol{w}}(1-y_i)\log(1-\hat{y}_i) = (1-y_i)\cdot\frac{\partial log(1-\hat{y}_i)}{\partial \boldsymbol{w}}$

$$= (1-y_i)\cdot\frac{1}{1-\hat{y}_i}\cdot -\hat{y}_i(1-\hat{y}_i)\boldsymbol{x}_i$$

$$= -(1-y_i)\cdot\hat{y}_i\cdot\boldsymbol{x}_i$$

$$\hat{y}_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失（cross-entropy loss）
  - 利用梯度下降（Gradient Descent）学习模型参数
    - 梯度为

$$\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\frac{1}{N}\sum_{n=1}^{N}\left(y_i \cdot \left(1-\hat{y}_i\right)\cdot \boldsymbol{x}_i - \left(1-y_i\right)\cdot \hat{y}_i \cdot \boldsymbol{x}_i\right)$$

$$= -\frac{1}{N}\sum_{\mathrm{i}=1}^{N}\boxed{\left(y_i-\hat{y}_i\right)\boldsymbol{x}_i}$$

误差× 特征

    - 迭代更新直至收敛

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \frac{1}{N}\sum_{i=1}^{N}\left(y_i-\hat{y}_i\right)\boldsymbol{x}_i$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计（ maximum likelihood estimation, MLE ）

- 最大似然通过**最大化真实数据的概率**，使得预测分布$p(y|\boldsymbol{x})$更接近真实分布$p_r(y|\boldsymbol{x})$。
- **最大化真实数据的概率：**在已知训练数据的情况下，找到最优的模型参数w，使得模型生成这些数据的概率最大。
- **似然(likelihood)和概率(probability) 的区别：**概率 $P(X \mid \theta)$ 表达了给定参数 θ下，观察到某个数据 X发生的可能性。似然L$(\theta \mid X)$表示在已知数据 X 的情况下，某个参数θ产生这些数据的可能性。

概率 $P(X|\theta)$ **已知参数，计算数据的概率。**

似然 $L(\theta|X)$ **已知数据，计算参数的可能性。**

逻辑斯蒂回归（Logistic Regression）

- 最大似然估计（ maximum likelihood estimation, MLE ）
  - 假定数据集：$\{(\boldsymbol{x}_i, y_i) | i \in \{1,2,\cdots N\}\}$ 是独立同分布的**(i.i.d.)** 生成的，给定参数$\boldsymbol{w}$的情况下，logistic 回归预测概率为

$$p(y_i = 1|\boldsymbol{x}_i) = \sigma(\boldsymbol{w}^T\boldsymbol{x}_i) = \frac{1}{1 + \exp(-\boldsymbol{w}^T\boldsymbol{x}_i)}$$

$$p(y_i = 0|\boldsymbol{x}_i) = 1 - p(\hat{y}_i = 1|\boldsymbol{x}_i)$$

简写为

$$p(y_i|\boldsymbol{x}_i;\boldsymbol{w}) = (p(y_i = 1|\boldsymbol{x}_i))^{y_i}(1 - p(y_i = 1|\boldsymbol{x}_i))^{(1-y_i)}$$

$$= (\frac{1}{1 + e^{-\boldsymbol{w}^T\boldsymbol{x}_i}})^{y_i}(1 - \frac{1}{1 + e^{-\boldsymbol{w}^T\boldsymbol{x}_i}})^{(1-y_i)}$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计（ maximum likelihood estimation, MLE ）

- MLE 的目标是最大化**所有样本的联合概率**
- （条件）似然函数

$$L(\boldsymbol{w}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i;\boldsymbol{w})$$

$$= \prod_{i=1}^{N} \left(\frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}_i}}\right)^{y_i} \left(1-\frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}_i}}\right)^{(1-y_i)}$$

- 最大化似然函数：找到最符合数据的参数，使得观测到的数据最有可能发生

$$\max_{\boldsymbol{w}} L(\boldsymbol{w})$$

逻辑斯蒂回归（Logistic Regression）

- 最大似然估计（ maximum likelihood estimation, MLE ）

最大似然：
$$L(\boldsymbol{w})=\prod_{i=1}^{N}\left(\frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}_i}}\right)^{y_i}\left(1-\frac{1}{1+e^{-\boldsymbol{w}^T\boldsymbol{x}_i}}\right)^{(1-y_i)}$$

最小交叉熵损失：
$$-\frac{1}{N}\sum_{i=1}^{N}(y_i log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i))$$

负对数似然函数与交叉熵损失(cross entropy loss)函数等价

$$\max_{\boldsymbol{w}} L(\boldsymbol{w}) \xLeftrightarrow{\text{取对数加负号}} \min_{\boldsymbol{w}} -\sum_{i=1}^{n} y_i\log\hat{y}_i+(1-y_i)\log(1-\hat{y}_i)$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计（ maximum likelihood estimation, MLE ）

- 无约束最优化问题

$$\max_{\boldsymbol{w}} \sum_{i=1}^{n} y_i \log \hat{y}_i + (1-y_i)\log(1-\hat{y}_i)$$

- 优化方法
  - 梯度下降
  - 随机梯度下降
  - 牛顿法
  - 拟牛顿法
  - 共轭梯度法
  - …

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

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

$$\hat{y}_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

- 计算梯度

$$\frac{\partial \log L(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\sum_{i=1}^{n} y_i \log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i)$$

$$= \sum_{i=1}^{n}\left(y_i\frac{1}{\hat{y}_i} - (1-y_i)\frac{1}{1-\hat{y}_i}\right)\frac{\partial}{\partial \boldsymbol{w}}\hat{y}_i$$

$$= \sum_{i=1}^{n}\left(y_i\frac{1}{\hat{y}_i} - (1-y_i)\frac{1}{1-\hat{y}_i}\right)\hat{y}_i(1-\hat{y}_i)\frac{\partial}{\partial \boldsymbol{w}}\boldsymbol{w}^T\boldsymbol{x}_i$$

$$= \sum_{i=1}^{n}(y_i(1-\hat{y}_i) - (1-y_i)\hat{y}_i)\,\boldsymbol{x}_i \quad = \sum_{i=1}^{N}\boxed{(y_i-\hat{y}_i)\boldsymbol{x}_i}$$

误差× 特征

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 计算梯度

$$\frac{\partial \log L(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}} \sum_{i=1}^{N} y_i \log \hat{y}_i + (1 - y_i) \log(1 - \hat{y}_i)$$

$$= \sum_{i=1}^{N} (y_i - \hat{y}_i) \boldsymbol{x}_i$$

- 梯度上升，迭代更新直至收敛

$$\boldsymbol{w} \leftarrow \boldsymbol{w} + \alpha \sum_{i=1}^{N} (y_i - \hat{y}_i) \boldsymbol{x}_i$$

逻辑斯蒂回归（Logistic Regression）

- 随机梯度下降（Stochastic Gradient Descend, SGD）

  - Motivation: 梯度下降需要计算整个数据集的梯度

    →计算量大、效率低

  - 随机梯度下降算法：基于**单个样本**

    ①随机选择**一个**训练样本$(\boldsymbol{x}_i, y_i)$

    ②计算**该样本上**的梯度: $\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}}\Big|_{\boldsymbol{w}=\boldsymbol{w}_t} = -(y_i - \hat{y}_i)\boldsymbol{x}_i$

    ③沿梯度反方向移动到下一个位置 $t+1$:

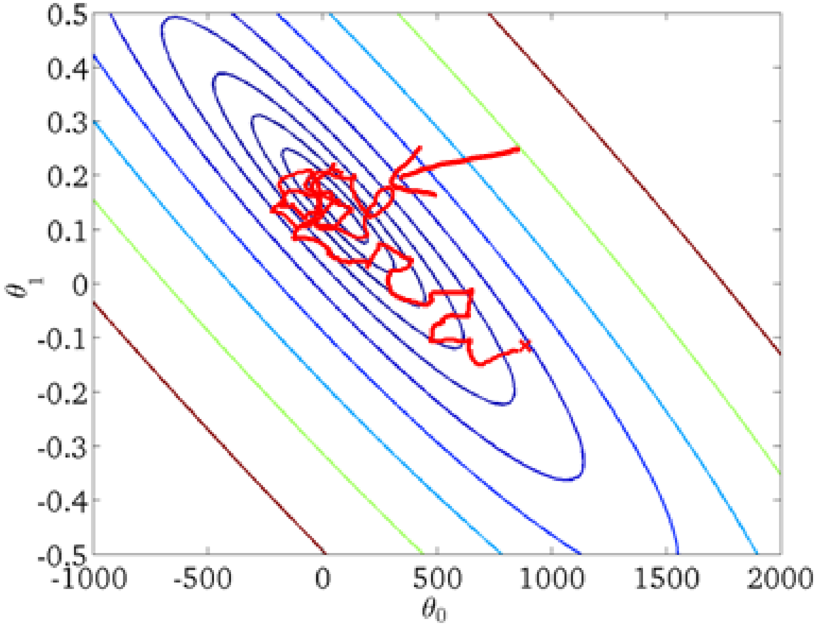
    $$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha(y_i - \hat{y}_i)\boldsymbol{x}_i, \alpha \text{ 是步长}$$

    ④ 重复步骤①②③直到收敛，得到最优$\boldsymbol{w}^*$

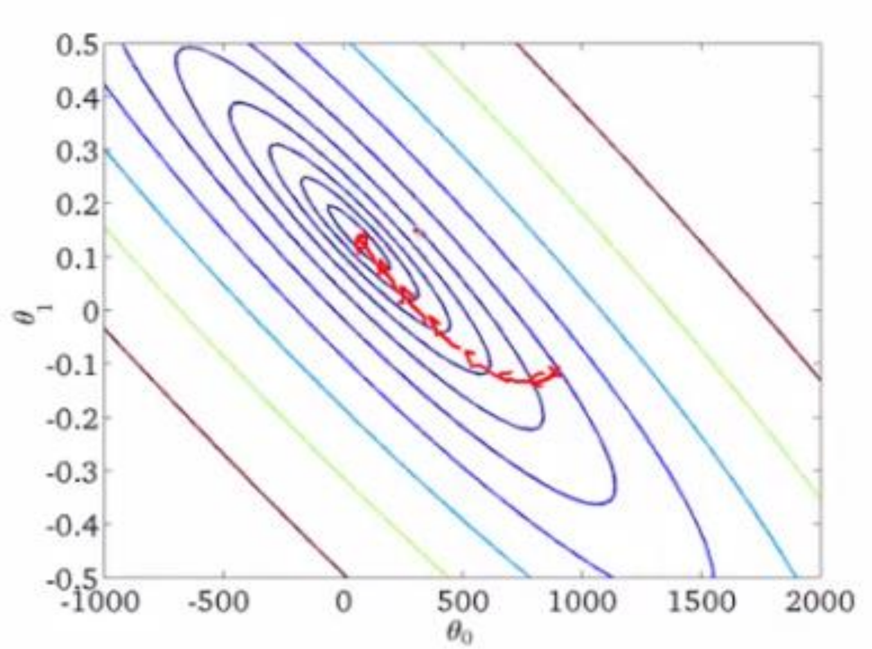
逻辑斯蒂回归（Logistic Regression）

随机梯度下降（ SGD） v.s. 梯度下降（GD）

基于随机挑选的单个样本更新参数w　　基于全部样本更新参数w



SGD



GD

逻辑斯蒂回归（Logistic Regression）

- 随机梯度下降（Stochastic Gradient Descend, SGD）
  - **小批量**随机梯度下降（mini-batch SGD ）

①随机选择**一批**训练样本$\{(\boldsymbol{x}_i, y_i)\}, i \in \{1,2,\cdots N'\}$

②计算**该批样本上**的梯度:

$$\frac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}}|_{\boldsymbol{w}=\boldsymbol{w}_t} = \frac{1}{N'}\sum_{i=1}^{N'} -(y_i - \hat{y}_i)\boldsymbol{x}_i$$

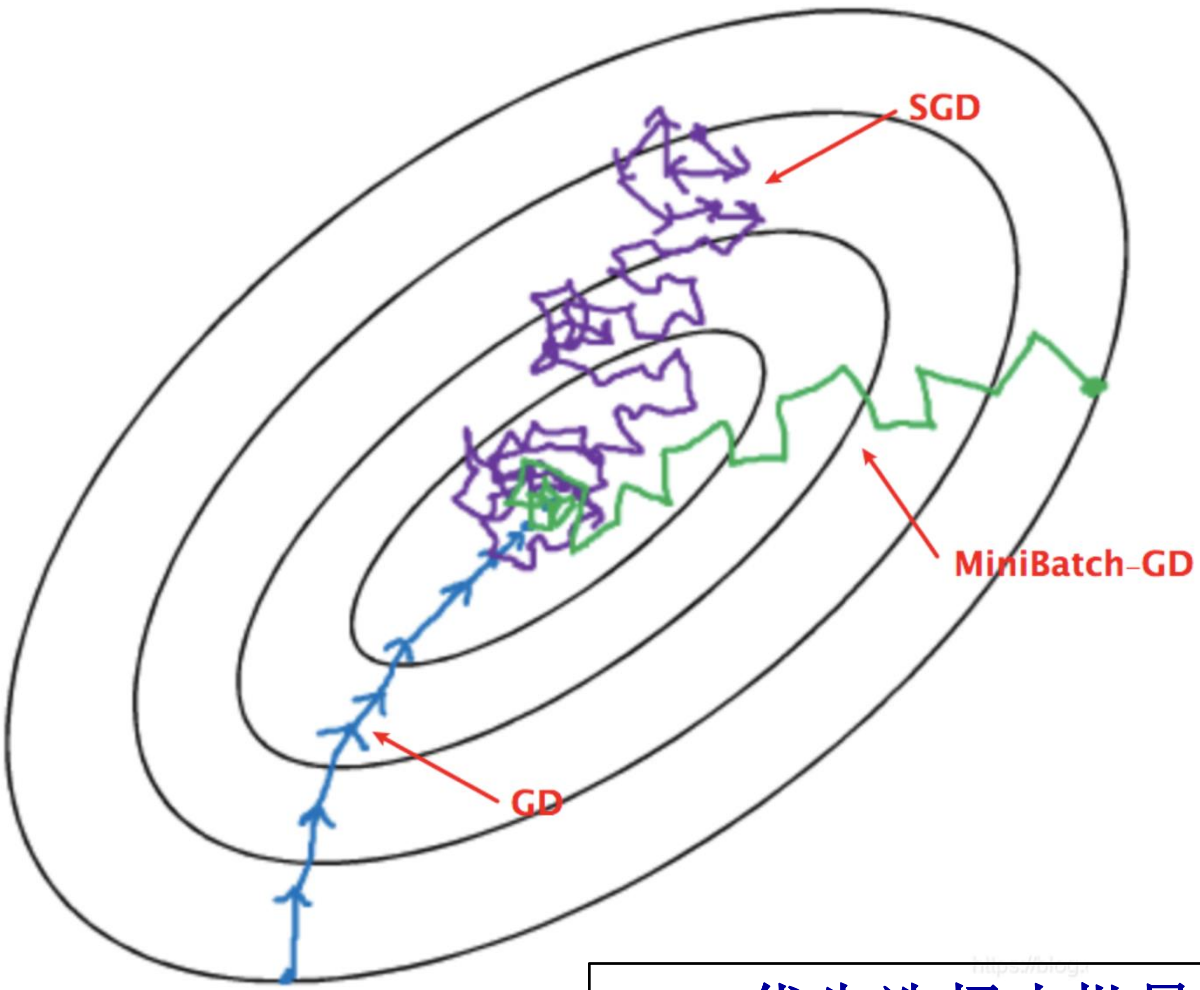
③ 沿梯度反方向移动到下一个位置 $t+1$:

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t + \alpha\frac{1}{N'}\sum_{i=1}^{N'}(y_i - \hat{y}_i)\boldsymbol{x}_i,\ \alpha \text{ 是步长}$$

④ 重复步骤①②③直到收敛，得到最优$\boldsymbol{w}^*$

逻辑斯蒂回归（Logistic Regression）

GD v.s. SGD v.s. mini-batch SGD



Tips: 优先选择小批量随机梯度下降法

牛顿法

- 牛顿法定义：一种**二阶优化算法**，用于求解无约束优化问题$\min\limits_{\boldsymbol{\theta}} f(\boldsymbol{\theta})$或非线性方程$\nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})=0$

$$\min_{\boldsymbol{\theta}} f(\boldsymbol{\theta}) \Leftrightarrow solve : \nabla_{\boldsymbol{\theta}} f(\boldsymbol{\theta})=0$$

- **求解**：首先，让$f(\theta)$在θ_t处进行二阶泰勒展开

$$f(\theta) \approx f(\theta_t) + \nabla f(\theta_t)^\top (\theta - \theta_t) + \frac{1}{2}(\theta - \theta_t)^\top \nabla^2 f(\theta_t)(\theta - \theta_t).$$

牛顿法

- 二阶泰勒展开：用于近似函数 $f(\theta)$ 在某点 θ_0 处的值，对于二阶展开，我们利用函数的梯度和 Hessian 矩阵来逼近。在 θ_0 处对 $f(\theta)$ 展开：

$$f(\theta) \approx f(\theta_0) + \nabla f(\theta_0)^\top(\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0).$$

符号说明

- $\theta \in \mathbb{R}^n$ 是 n 维变量。
- $\nabla f(\theta_0) \in \mathbb{R}^n$ 是 **梯度向量**（一阶偏导数）。
- $\nabla^2 f(\theta_0) \in \mathbb{R}^{n\times n}$ 是 **Hessian 矩阵**（二阶偏导数）
- $(\theta - \theta_0)^\top$ 代表行向量形式。是偏移量。

$$H = \nabla^2 f(\theta) = \begin{bmatrix} \frac{\partial^2 f}{\partial \theta_1^2} & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_2} & \cdots & \frac{\partial^2 f}{\partial \theta_1 \partial \theta_n} \\ \frac{\partial^2 f}{\partial \theta_2 \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_2^2} & \cdots & \frac{\partial^2 f}{\partial \theta_2 \partial \theta_n} \\ \vdots & \vdots & \ddots & \vdots \\ \frac{\partial^2 f}{\partial \theta_n \partial \theta_1} & \frac{\partial^2 f}{\partial \theta_n \partial \theta_2} & \cdots & \frac{\partial^2 f}{\partial \theta_n^2} \end{bmatrix}$$

牛顿法

- 二阶泰勒展开：用于近似函数 $f(\theta)$在某点θ_0处的值，其 **多变量形式**公式如下

$$f(\theta) \approx f(\theta_0) + \nabla f(\theta_0)^\top (\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0).$$

- 第一项$f(\boldsymbol{\theta_0})$：函数在 θ_0处的值，提供了一个基准，使近似从θ_0开始。
- 第二项$\nabla f(\boldsymbol{\theta_0})^T(\boldsymbol{\theta} - \boldsymbol{\theta_0})$：计算$f(\theta)$在$\theta_0$处的线性变化，提供对$f(\theta)$的线性近似。
- 第三项：衡量函数在不同方向上的变化，提供了一个曲率信息，使近似更精准(Hessian矩阵描述的是函数$f(\theta)$的曲率)。

牛顿法

- **目标：**

$$\min_{\theta} f(\theta) \Leftrightarrow solve: \ \nabla_{\theta} f(\theta) = 0$$

- $f(\theta)$在θ_t处进行二阶泰勒展开

$$f(\theta) \approx f(\theta_t) + \nabla f(\theta_t)^\top (\theta - \theta_t) + \frac{1}{2}(\theta - \theta_t)^\top \nabla^2 f(\theta_t)(\theta - \theta_t).$$

- 令梯度$\nabla_{\theta}\phi(\theta) = 0$

$$\nabla_{\theta}\phi(\theta) = \frac{\partial}{\partial\theta}\left[f(\theta_0) + \nabla f(\theta_0)^\top (\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0)\right]$$

牛顿法

- 令梯度$\nabla_\theta \phi(\theta) = 0$

$$\nabla_\theta \phi(\theta) = \frac{\partial}{\partial \theta}\left[f(\theta_0) + \nabla f(\theta_0)^\top (\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0)\right]$$

- 第一项：$\frac{\partial}{\partial \boldsymbol{\theta}} f(\boldsymbol{\theta_0}) = 0$

- 第二项：$\frac{\partial}{\partial \boldsymbol{\theta}} \nabla f(\boldsymbol{\theta_0})^T(\boldsymbol{\theta} - \boldsymbol{\theta_0}) = \nabla f(\boldsymbol{\theta_0})$

- 第三项：$\frac{\partial}{\partial \boldsymbol{\theta}} \frac{1}{2}(\boldsymbol{\theta} - \boldsymbol{\theta_0})^T \nabla^2 f(\boldsymbol{\theta_0})(\boldsymbol{\theta} - \boldsymbol{\theta_0})$

$$\boxed{\frac{\partial \mathbf{x}^T \mathbf{B} \mathbf{x}}{\partial \mathbf{x}} = (\mathbf{B} + \mathbf{B}^T)\mathbf{x}}$$

$$= \frac{1}{2}\left(\nabla^2 f(\boldsymbol{\theta_0}) + \left(\nabla^2 f(\boldsymbol{\theta_0})\right)^T\right)(\boldsymbol{\theta} - \boldsymbol{\theta_0})$$

$$= \nabla^2 f(\boldsymbol{\theta_0})(\boldsymbol{\theta} - \boldsymbol{\theta_0})$$

Hessian 矩阵是对称矩阵

$(H)^T$ =H

牛顿法

- 令梯度$\nabla_\theta \phi(\theta) = 0$

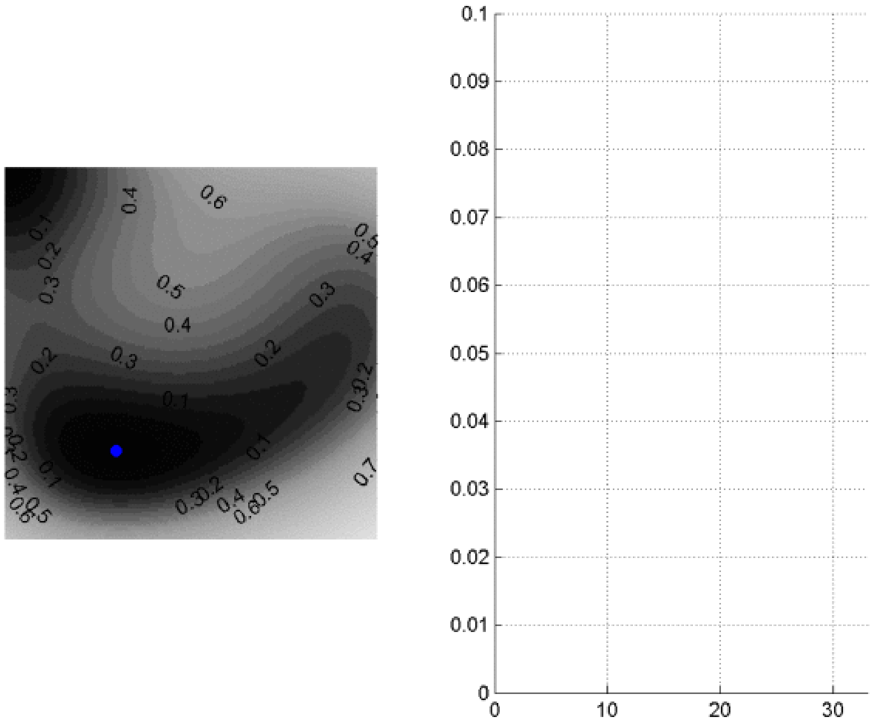
$$\nabla_\theta \phi(\theta) = \frac{\partial}{\partial \theta}\left[f(\theta_0) + \nabla f(\theta_0)^\top (\theta - \theta_0) + \frac{1}{2}(\theta - \theta_0)^\top \nabla^2 f(\theta_0)(\theta - \theta_0) \right]$$

$$\Rightarrow \nabla_\theta \phi(\theta) = \nabla f(\boldsymbol{\theta_0}) + \nabla^2 f(\boldsymbol{\theta_0})(\boldsymbol{\theta} - \boldsymbol{\theta_0}) = \mathbf{0}$$

$$\Rightarrow \boldsymbol{\theta}^* = \boldsymbol{\theta_0} - \nabla^2 f(\boldsymbol{\theta_0})^{-1} \nabla f(\boldsymbol{\theta_0})$$

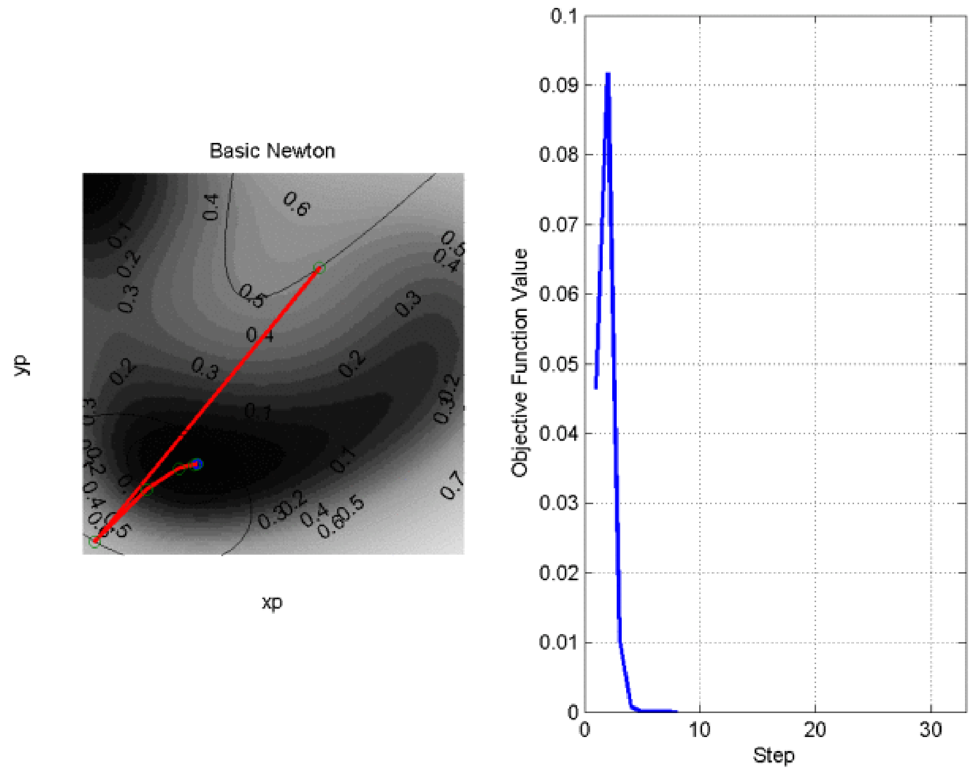
- 由于牛顿法是**近似解法**，一次更新通常**不会**直接达到最优点，因此需要不断迭代 **直到收敛**（即$\nabla_\theta f(\theta) = 0$）。

$$\theta_{t+1} = \theta_t - \left[\nabla^2 f(\theta_t)\right]^{-1} \nabla f(\theta_t)$$



梯度下降法：收敛速度相较于牛顿法慢

- 牛顿法：收敛快，但每步计算成本较高（计算hessian矩阵代价大）

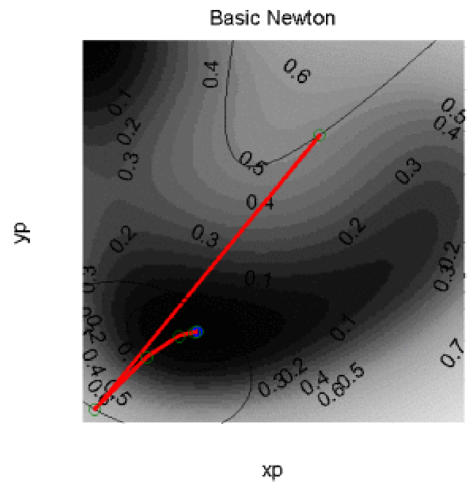


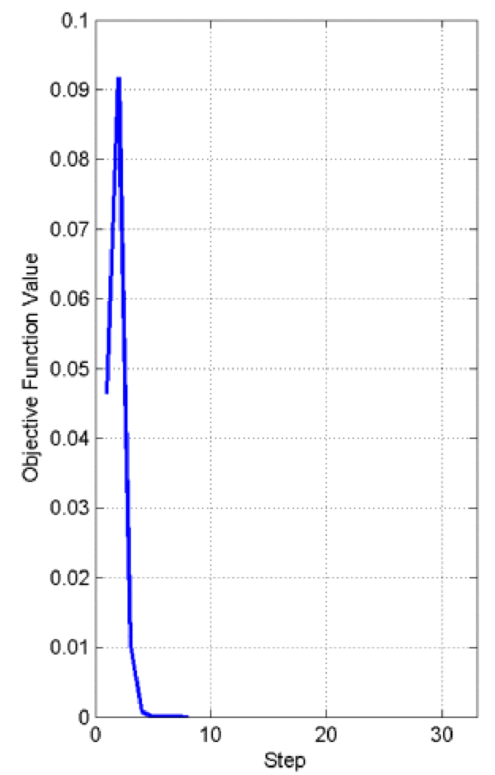
牛顿法收敛快的原因：

二阶信息的利用

- 牛顿法不仅使用梯度（即一阶导数）来引导搜索方向，还使用了Hessian矩阵（二阶导数矩阵），它反映了目标函数的曲率信息。
- 通过利用Hessian矩阵，牛顿法能够精确地调整每一步的步长，使得更新步骤不仅考虑了当前点的斜率，还考虑了目标函数在该点的形状（曲率）。这使得更新方向更加精确，避免了在某些方向上步长过大或过小的问题。

- **牛顿法：收敛快，但每步计算成本较高（计算hessian矩阵代价大）**





基于牛顿法的logistic regression

- 给定训练集： $\{(\boldsymbol{x}_i, y_i)\}$，$i \in \{1,2,\cdots N\}$
- 优化目标（交叉熵损失）：

$$\min_{\boldsymbol{w}} -\frac{1}{N}\sum_{i=1}^{N}(y_i log\hat{y}_i + (1-y_i)\log(1-\hat{y}_i))$$

- 问题求解：

牛顿法更新公式： $\boxed{\theta_{t+1} = \theta_t - [\nabla^2 f(\theta_t)]^{-1}\nabla f(\theta_t)}$

1.计算$\nabla R(\boldsymbol{w})$：$\dfrac{\partial R(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\dfrac{1}{N}\sum_{i=1}^{N}(y_i \cdot (1-\hat{y}_i)\cdot \boldsymbol{x}_i - (1-y_i)\cdot \hat{y}_i \cdot \boldsymbol{x}_i)$

$$= -\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)\boldsymbol{x}_i = -\frac{1}{N}\boldsymbol{X}^T(Y-\hat{Y})$$

基于牛顿法的logistic regression

- 问题求解：

$$\hat{y}_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

2.计算$\nabla^2 R(\boldsymbol{w})$：

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

$$\frac{\partial \nabla R(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\left(-\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)\boldsymbol{x}_i\right)$$

$$= -\frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{\partial \boldsymbol{w}}(-\hat{y}_i\boldsymbol{x}_i)$$

向量对向量求导是一个矩阵！

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i(1-\hat{y}_i)\frac{\partial(\boldsymbol{w}^T\boldsymbol{x}_i)}{\partial \boldsymbol{w}}(\boldsymbol{x}_i)^{\mathrm{T}}$$

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i(1-\hat{y}_i)\boldsymbol{x}_i(\boldsymbol{x}_i)^{\mathrm{T}}$$

基于牛顿法的logistic regression

- 问题求解：

2.计算$\nabla^2 R(\boldsymbol{w})$：$\dfrac{\partial \nabla R(\boldsymbol{w})}{\partial \boldsymbol{w}} = -\dfrac{1}{N}\sum_{i=1}^{N}\dfrac{\partial}{\partial \boldsymbol{w}}(-\hat{y}_i \boldsymbol{x}_i)$

向量对向量求导是一个矩阵！

如果我们有：

$$y = g(w)x$$

其中：

- $g(w)$ 是一个 **标量函数**，比如 $g(w) = \sigma(w^\top x)$。
- x 是一个 **列向量**。

求导时，利用 **外积规则（Outer Product Rule）**：

$$\frac{\partial (g(w)x)}{\partial w} = \left(\frac{\partial g(w)}{\partial w}\right) \otimes x$$

因为：

$$a \otimes b = ab^\top$$

- $\frac{\partial g(w)}{\partial w}$ 是一个 **列向量**。
- x 也是 **列向量**。

基于牛顿法的logistic regression

- 问题求解：

$$\hat{y}_i = \frac{1}{1+\exp(-\boldsymbol{w}^T\boldsymbol{x})}$$

2.计算$\nabla^2 R(\boldsymbol{w})$：

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

$$\frac{\partial \nabla R(\boldsymbol{w})}{\partial \boldsymbol{w}} = \frac{\partial}{\partial \boldsymbol{w}}\left(-\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)\boldsymbol{x}_i\right)$$

$$= -\frac{1}{N}\sum_{i=1}^{N}\frac{\partial}{\partial \boldsymbol{w}}(-\hat{y}_i\boldsymbol{x}_i)$$

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i(1-\hat{y}_i)\frac{\partial(\boldsymbol{w}^T\boldsymbol{x}_i)}{\partial \boldsymbol{w}}(\boldsymbol{x}_i)^{\mathrm{T}}$$

$$= -\frac{1}{N}\sum_{i=1}^{N} y_i(1-\hat{y}_i)\boldsymbol{x}_i(\boldsymbol{x}_i)^{\mathrm{T}}$$

基于牛顿法的logistic regression

- 使用牛顿法更新权重，直至收敛：

$$\boldsymbol{w}_{t+1} = \boldsymbol{w}_t - \nabla^2 R(\boldsymbol{w}_t)^{-1} \nabla R(\boldsymbol{w}_t)$$

其中：$\nabla^2 R(\boldsymbol{w}_t) = -\frac{1}{N}\sum_{i=1}^{N} y_i(1-\hat{y}_i)\boldsymbol{x}_i(\boldsymbol{x}_i)^{\mathrm{T}}$

$$\nabla R(\boldsymbol{w}_t) = -\frac{1}{N}\sum_{i=1}^{N}(y_i - \hat{y}_i)\boldsymbol{x}_i$$

多分类问题

- 将二分类的学习方法直接推广到多分类
- 基本思想：拆解法，将多分类任务拆为若干个二分类任务求解
  - 一对多 (one vs rest)：
    1. 对于 K类，每次选取一个类别作为正类，其余所有类别作为 负类，训练 K 个二分类器。
    2. 预测时，对每个测试样本计算 K个分类器的得分，选择得分最高的类别作为最终类别。
    3. 计算量小，仅需要K个二分类器。

多分类问题

- 将二分类的学习方法直接推广到多分类
- 基本思想：拆解法，将多分类任务拆为若干个二分类任务求解

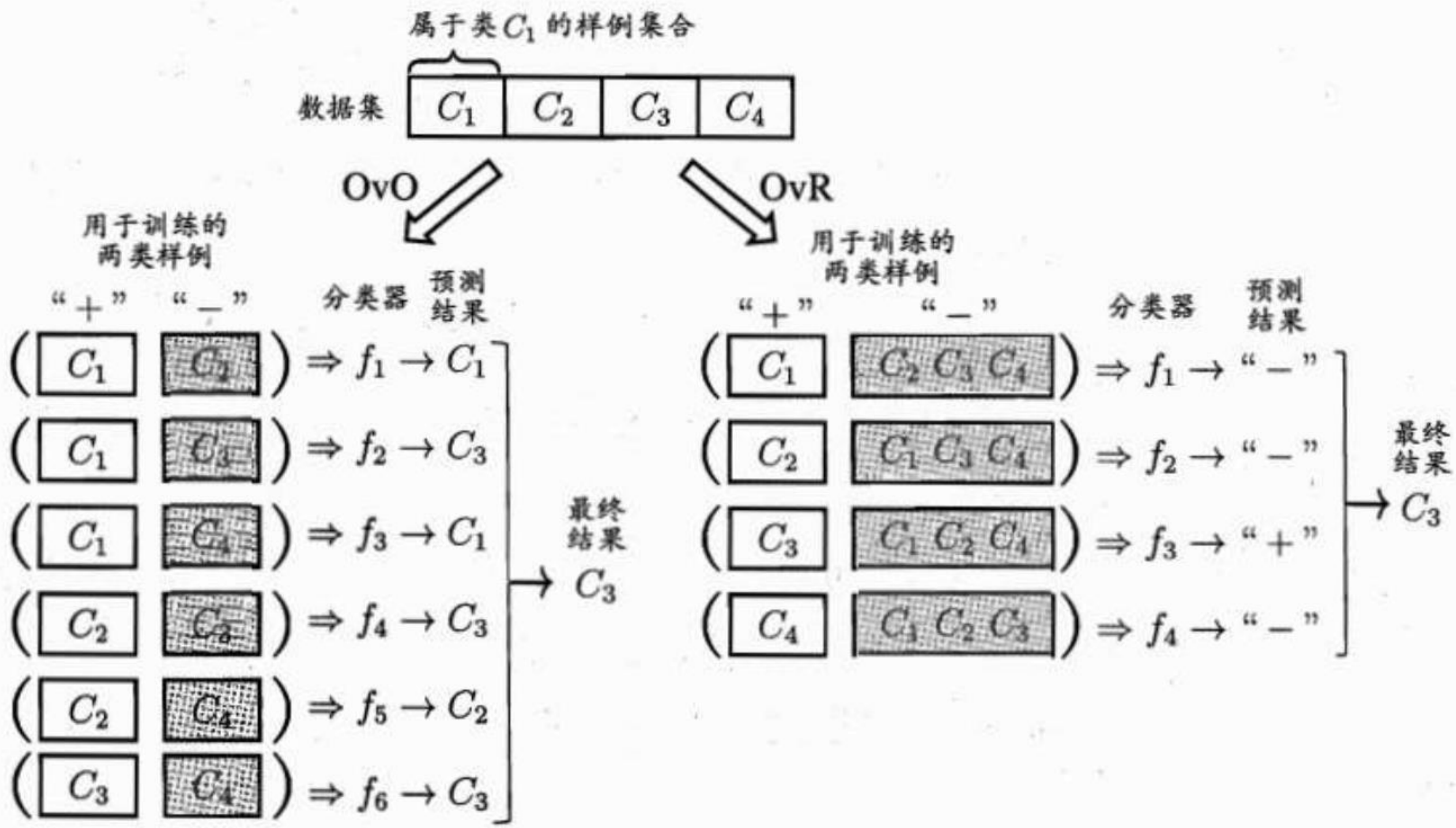
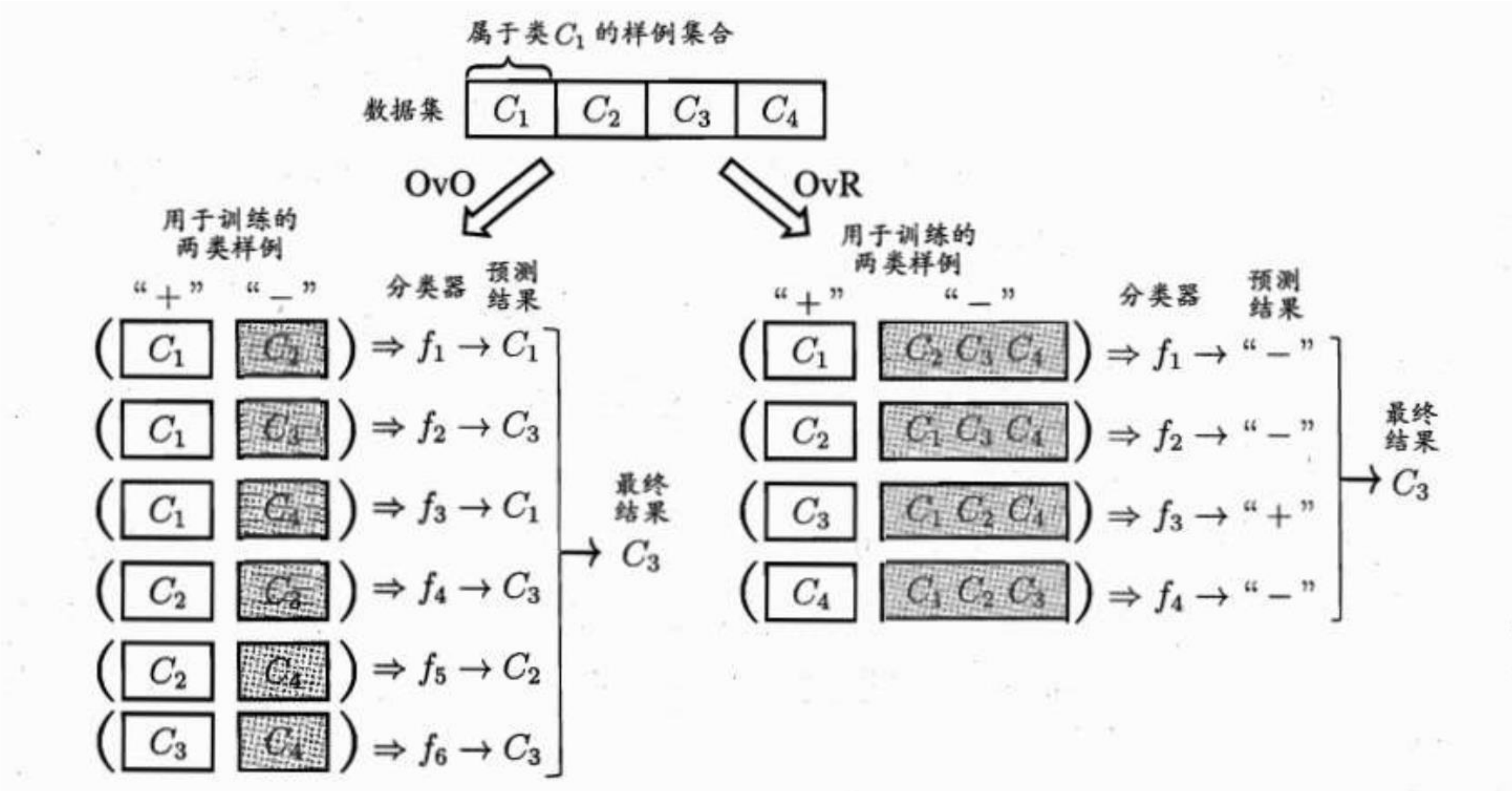


图 3.4 OvO 与 OvR 示意图

多分类问题

- 一对一(one vs one)：

1. 对于 K类，每次从 K 类中选取 两个类别，构建一个二分类任务，共训练 $\frac{K(K-1)}{2}$ 个分类器。



多分类问题

- 将二分类的学习方法直接推广到多分类
- 基本思想：拆解法，将多分类任务拆为若干个二分类任务求解
  - 一对一 (one vs one)：
    1. 对于 K类，每次选取一个类别作为正类，另一个类别作为负类，训练 K(K-1)/2 个二分类器。
    2. 预测时，对每个测试样本计算 K(K-1)/2个分类器的得分，投票选出最终类别。

多分类问题

投票法：

- **每个分类器输出一个类别**，最终选择**票数最多**的类别作为预测结果。
- 适用于 **一对一（OvO）方法**，因为 OvO 训练了 $K(K-1)/2$ 个分类器，每个分类器只能区分两类，最终让所有分类器"投票"决定样本属于哪个类别。

示例

假设有 4 类（A、B、C、D），我们训练了 6 个 OvO 分类器：

- $f(A, B) \to$ A
- $f(A, C) \to$ C
- $f(A, D) \to$ A
- $f(B, C) \to$ C
- $f(B, D) \to$ B
- $f(C, D) \to$ C

可能出现**平票，**
需要引入加权投票

投票统计：

类别	A	B	C	D
票数	2	1	3	0

最终预测类别：**C（票数最多）**。

多分类问题

加权投票法：

- 每个分类器不仅输出类别，还输出**置信度（得分或概率）**。
- 通过累加每个类别的得分，选择**总得分最高**的类别作为最终类别。

示例

假设有 4 类（A、B、C、D），6 个 OvO 分类器的输出置信度如下：

- $f(A, B)$ → A (0.8), B (0.2)
- $f(A, C)$ → A (0.3), C (0.7)
- $f(A, D)$ → A (0.6), D (0.4)
- $f(B, C)$ → B (0.4), C (0.6)
- $f(B, D)$ → B (0.7), D (0.3)
- $f(C, D)$ → C (0.5), D (0.5)

计算**累加得分**：

类别	A	B	C	D
票数（置信度之和）	1.7	1.3	1.8	1.2

最终预测类别：**C（得分最高）**。

多分类逻辑回归

- 逻辑回归解决多分类问题?

  - **方法一：**将**多分类问题拆解为多个二分类问题**，每个类别都训练一个逻辑回归分类器来区分“该类别 vs. 其他类别”，即通过**一对多**实现

具体做法：

- 对于 K 个类别，训练 K 个**二分类逻辑回归模型**。
- **第 k 个分类器：**
  - **正类：** 类别为 k 的样本
  - **负类：** 所有其他类别的样本
- 预测时：

  预测为每个类别k的概率

  - 计算所有 K 个分类器的**概率输出** $p_k(x) = \sigma(w_k^T x)$。
  - 选择**概率最高的类别**作为最终预测类别：

$$\hat{y} = \arg\max_k p_k(x)$$

多分类逻辑回归

- 逻辑回归解决多分类问题？
  - **方法一：**将**多分类问题拆解为多个二分类问题**，每个类别都训练一个逻辑回归分类器来区分“该类别 vs. 其他类别”，即通过**一对多**实现

1. 问题设定：

假设我们有一个包含 3 个类别的分类问题，类别为 A、B 和 C。我们有 5 个训练样本，每个样本的特征向量 $\mathbf{x}_i$ 和标签 y_i 如下：

样本	特征 $\mathbf{x}_i$	标签 y_i
1	$[1.0, 2.0]$	A
2	$[2.0, 3.0]$	B
3	$[3.0, 1.0]$	C
4	$[2.5, 3.5]$	A
5	$[1.5, 2.5]$	B

多分类逻辑回归

我们将这个三分类问题拆解为三个二分类问题，每个分类器训练数据都将该类别作为正类，其他类别作为负类。

分类器 1: 类别 A vs. 非 A (B 和 C)

训练数据集：

- 正类：所有标签为 A 的样本（样本 1 和样本 4）
- 负类：所有标签为 B 和 C 的样本（样本 2、3 和 5）

训练目标：

- 对于类别 A，我们的逻辑回归模型将学习如何区分类别 A 和其他类别（B、C）。

分类器 2: 类别 B vs. 非 B (A 和 C)

训练数据集：

- 正类：所有标签为 B 的样本（样本 2 和样本 5）
- 负类：所有标签为 A 和 C 的样本（样本 1、3 和 4）

训练目标：

- 对于类别 B，我们的逻辑回归模型将学习如何区分类别 B 和其他类别（A、C）。

分类器 3: 类别 C vs. 非 C (A 和 B)

训练数据集：

- 正类：所有标签为 C 的样本（样本 3）
- 负类：所有标签为 A 和 B 的样本（样本 1、2、4 和 5）

训练目标：

- 对于类别 C，我们的逻辑回归模型将学习如何区分类别 C 和其他类别（A、B）。

多分类逻辑回归

在训练阶段，我们分别训练三个二分类逻辑回归模型：

- 分类器 1: 学习如何区分类别 A 与非 A。
- 分类器 2: 学习如何区分类别 B 与非 B。
- 分类器 3: 学习如何区分类别 C 与非 C。

在预测阶段，假设我们有一个新的样本 $\mathbf{x} = [2.2, 3.1]$，我们将计算每个分类器的输出概率：

- 对于分类器 1（类别 A vs. 非 A），得到预测概率 p_1。
- 对于分类器 2（类别 B vs. 非 B），得到预测概率 p_2。
- 对于分类器 3（类别 C vs. 非 C），得到预测概率 p_3。

根据每个分类器的输出，我们选择具有最大概率的类别作为最终的预测类别：

$$\hat{y} = \arg\max\{p_1, p_2, p_3\}$$

例如：

- 假设分类器 1 输出 $p_1 = 0.8$，表示输入样本 $\mathbf{x}$ 属于类别 A 的概率为 0.8。
- 假设分类器 2 输出 $p_2 = 0.3$，表示输入样本 $\mathbf{x}$ 属于类别 B 的概率为 0.3。
- 假设分类器 3 输出 $p_3 = 0.4$，表示输入样本 $\mathbf{x}$ 属于类别 C 的概率为 0.4。

根据以上结果，最终预测的类别是类别 A，因为 $p_1 = 0.8$ 是最大的。

多分类逻辑回归

- 逻辑回归解决多分类问题?
  - 方法一：将**多分类问题拆解为多个二分类问题**，每个类别都训练一个逻辑回归分类器来区分“该类别 vs. 其他类别”，即通过一对多实现

具体做法:

- 对于 K 个类别，训练 K 个**二分类逻辑回归模型**。
- **第 k 个分类器**:
  - **正类**：类别为 k 的样本
  - **负类**：所有其他类别的样本
- 预测时:
  - 计算所有 K 个分类器的**概率输出** $p_k(x) = \sigma(w_k^T x)$。
  - 选择**概率最高的类别**作为最终预测类别:

$$\hat{y} = \arg\max_k p_k(x)$$

预测为每个类别k的概率

可能存在决策边界不一致的问题（多个分类器可能都预测为正类）、类别间关系难以建模、不适合复杂的多类之间的关系等问题。

多分类逻辑回归

- 方法二：**softmax回归**
  - 核心思想：基于softmax函数将**多个类别的概率建模在同一个公式中**，输出所有类别的**概率分布（所有类别概率和为1）**，然后选择最大概率的类别作为最终预测结果。
  - Softmax 回归是一种多分类模型，也称做多类Logistic回归。
  - 它是一种广泛使用的分类算法，常常作为深度学习分类模型的最后一层执行分类预测。

多分类逻辑回归

- **为什么使用Softmax函数？**

  - Sigmoid 函数的值域是(0,1)，其只适用于二分类，即只能给出模型预测为类别1或类别0的概率。

  - 对于多分类任务， Sigmoid不能保证概率总和为 1，导致分类冲突，即

如果直接对每个类别使用 Sigmoid：

$$p_k = \frac{1}{1 + e^{-w_k^T x}}, \quad k = 1, 2, \ldots, K$$

这意味着 **每个类别的概率是独立计算的**，但：

$$\sum_{k=1}^{K} p_k \neq 1$$

不同类别的概率不会归一化，无法保证它们是一个有效的**概率分布**。

多分类逻辑回归

- **Softmax函数：**一种**归一化的指数函数**
  - ➢ 可以将任意实数向量转换为**概率分布**，即将任意一个包含K维的实数向量“压缩”到另一个K维实向量中，使得每一个元素的范围都在(0,1)之间，并且所有元素的和为1。

$$(\boldsymbol{z})^{\mathrm{T}} = [z_1, z_2, z_3, \cdots, z_K]$$

$$\Big\downarrow \ \textbf{Softmax}$$

$$\textbf{Softmax}\ (\boldsymbol{z})^{\mathrm{T}} = [\frac{e^{z_1}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \cdots, \frac{e^{z_K}}{\sum_{i=1}^{K} e^{z_i}}]$$

多分类逻辑回归

- **Softmax函数：** 一种归一化的指数函数

$$(\boldsymbol{z})^{\mathrm{T}} = [z_1, z_2, z_3, \cdots, z_K]$$

$$\Downarrow \text{Softmax}$$

$$\text{Softmax}\ (\boldsymbol{z})^{\mathrm{T}} = [\frac{e^{z_1}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \cdots, \frac{e^{z_K}}{\sum_{i=1}^{K} e^{z_i}}]$$

e^{z_i} 计算每个类别得分的指数值。

分母 $\sum_{j=1}^{K} e^{z_j}$ 作用是归一化，确保所有元素的和为 1：

$$\sum_{i=1}^{K} \text{Softmax}(z)_i = 1$$

每个 $\text{Softmax}(z)_i$ 都在 (0,1) 之间，使其可以解释为**概率**。

多分类逻辑回归

- **Softmax函数：**一种归一化的指数函数

$$\text{Softmax}\ (\boldsymbol{z})^{\mathrm{T}} = [\frac{e^{z_1}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \frac{e^{z_2}}{\sum_{i=1}^{K} e^{z_i}}, \cdots, \frac{e^{z_K}}{\sum_{i=1}^{K} e^{z_i}}]$$

假设有一个 3 维向量：

$$z = [2.0, 1.0, 0.1]$$

计算 Softmax：

1. 计算指数：

$$e^{2.0} = 7.389, \quad e^{1.0} = 2.718, \quad e^{0.1} = 1.105$$

2. 计算分母（归一化因子）：

$$Z = 7.389 + 2.718 + 1.105 = 11.212$$

3. 计算 Softmax 输出：

$$p_1 = \frac{7.389}{11.212} = 0.659, \quad p_2 = \frac{2.718}{11.212} = 0.242, \quad p_3 = \frac{1.105}{11.212} = 0.099$$

最终得到概率分布：

$$p = [0.659, 0.242, 0.099]$$

多分类逻辑回归

模型描述

- 给定一个样本 $(\boldsymbol{x})^T = [x_1, x_2, \cdots, x_d]$，对每个类别$c$计算样本在该类别上的得分（logits）

$$\boldsymbol{z}_c = \boldsymbol{w}_c{}^T \boldsymbol{x}$$

$\boldsymbol{w}_c{}^T = [w_{c1,} w_{c2,} w_{c3,} \cdots, w_{cd}]$表示该类别上对应的模型参数

- Softmax归一化

$$p(y = c|\boldsymbol{x}) = \frac{e^{\boldsymbol{z}_c}}{\sum_{c'=1}^{C} e^{\boldsymbol{z}_{c'}}}, \qquad c \in \{1,2, \ldots, C\}$$

多分类逻辑回归

模型描述

- 矩阵形式

模型参数矩阵 $\quad \boldsymbol{w} = \left[\boldsymbol{w}_{1,}\boldsymbol{w}_{2,}\boldsymbol{w}_{3,}\cdots,\boldsymbol{w}_{C}\right]^{T} \in \mathbb{R}^{C\times d}$

logits $\quad \boldsymbol{z} = [\boldsymbol{w}_1{}^T\boldsymbol{x}, \boldsymbol{w}_2{}^T\boldsymbol{x}, \boldsymbol{w}_3{}^T\boldsymbol{x}\cdots, \boldsymbol{w}_C{}^T\boldsymbol{x}]^T$

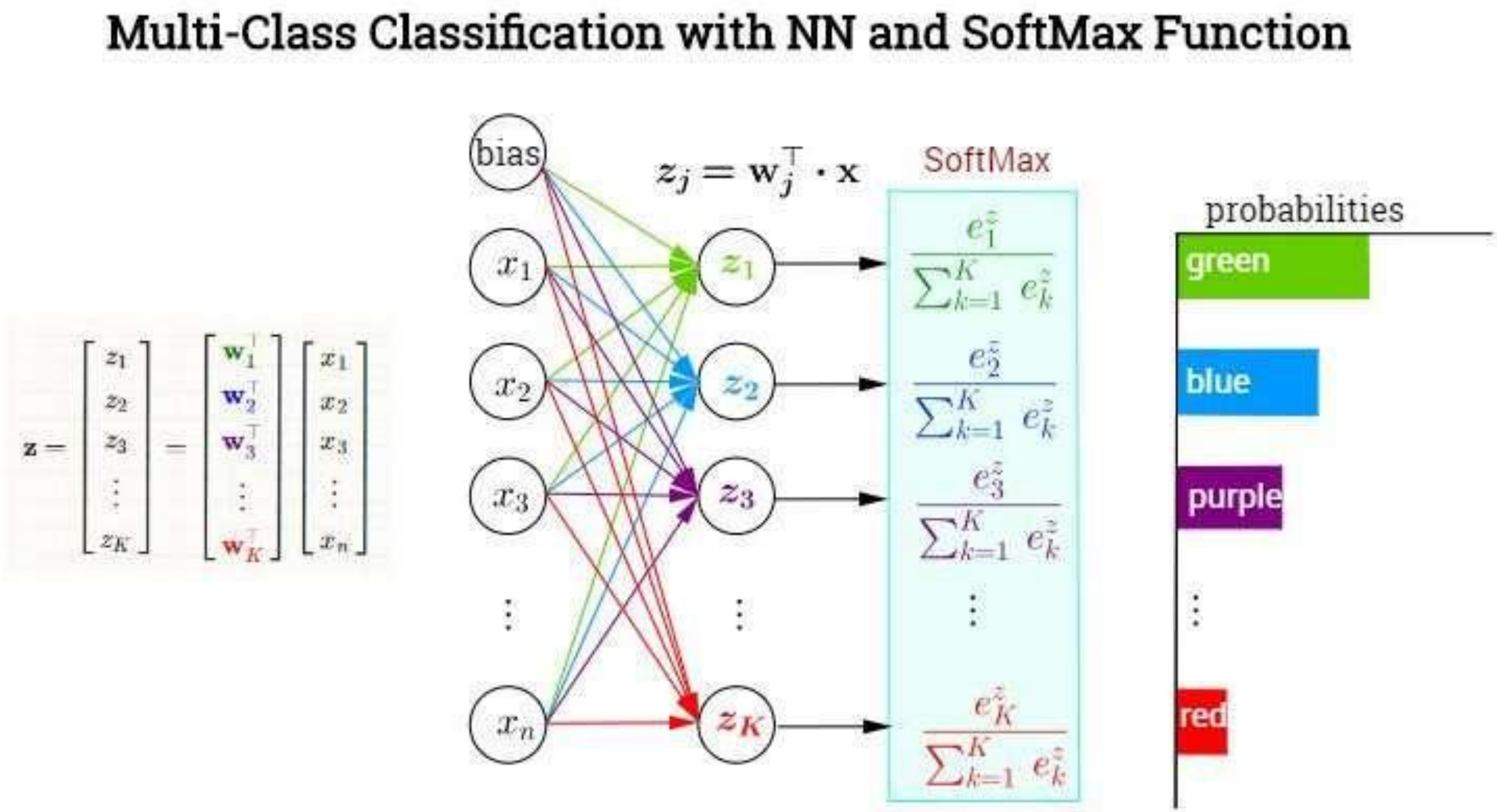
基于Softmax得每个类别概率

$$\boldsymbol{P} = [p(y=1|\boldsymbol{x}), p(y=2|\boldsymbol{x}), \cdots, p(y=C|\boldsymbol{x})]^T$$

$$= \left[\frac{e^{z_1}}{\sum_{c\prime=1}^{C} e^{z_{c'}}}, \frac{e^{z_2}}{\sum_{c\prime=1}^{C} e^{z_{c'}}}, \cdots, \frac{e^{z_C}}{\sum_{c\prime=1}^{C} e^{z_{c'}}}\right]^T$$

多分类逻辑回归

- 模型描述



多分类逻辑回归

模型描述

- 求解变量：　$\boldsymbol{w} = [\boldsymbol{w}_1, \boldsymbol{w}_2, \boldsymbol{w}_3, \cdots, \boldsymbol{w}_C]^T \in \mathbb{R}^{C\times d}$

- 最大似然求解：假设数据集中样本$\{(\boldsymbol{x}_i, y_i) \mid i \in \{1,2,\cdots N\}\}$是独立同分布的**(i.i.d.)** →最大化**所有样本上真实类别的概率**

$$L(\boldsymbol{w}) = \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i)$$

$$1\{y^{(i)} = j\} = \begin{cases} 1, & \text{如果样本 } i \text{ 属于类别 } j \\ 0, & \text{否则} \end{cases}$$

$$= \prod_{i=1}^{N}\prod_{j=1}^{C}(p(y_i = j|\boldsymbol{x}_i))^{1\{y_i=j\}}$$

$$= \prod_{i=1}^{N}\prod_{j=1}^{C}\left(\frac{e^{\boldsymbol{w}_j{}^T\boldsymbol{x}}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}}\right)^{1\{y_i=j\}}$$

多分类逻辑回归

- 最大对数似然估计

$$\log L(\boldsymbol{W}) = \log \prod_{i=1}^{N} p(y_i|\boldsymbol{x}_i)$$

$$= \sum_{i=1}^{N} \log \prod_{j=1}^{C} \left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)^{1\{y_i=j\}}$$

$$= \sum_{i=1}^{N} \sum_{j=1}^{C} 1(y_i = j) \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)$$

- 目标 $\max_{\boldsymbol{w}} \log L(\boldsymbol{w})$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 计算梯度

$$\frac{\partial \log L(\boldsymbol{W})}{\partial \boldsymbol{W}} = \frac{\partial}{\partial \boldsymbol{W}} \sum_{i=1}^{N} \sum_{j=1}^{C} 1(y_i = j) \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)$$

- 对每个类别的 $\boldsymbol{w}_k$ 分别计算导数

$$\frac{\partial \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)}{\partial \boldsymbol{w}_k} = \frac{\partial}{\partial \boldsymbol{w}_k} \left(\boldsymbol{w}_j{}^T \boldsymbol{x}_i - \log \sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}\right)$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 对每个类别的 $\boldsymbol{w}_k$ 分别计算导数

$$\frac{\partial \log\left(\frac{e^{\boldsymbol{w}_j{}^T\boldsymbol{x}}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}}\right)}{\partial \boldsymbol{w}_k} = \frac{\partial}{\partial \boldsymbol{w}_k}\left(\boldsymbol{w}_j{}^T\boldsymbol{x}_i - \log\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}\right)$$

➢ **第一项：**

当 $j = k$ 时，$w_j^T x_i = w_k^T x_i$，所以：

$$\frac{\partial}{\partial w_k} w_k^T x_i = x_i$$

当 $j \neq k$ 时，$w_j^T x_i$ 和 w_k **无关**，所以：

$$\frac{\partial}{\partial w_k} w_j^T x_i = 0$$

逻辑斯蒂回归（Logistic Regression）

- 第二项$\frac{\partial}{\partial w_k}\log\sum_{j'=1}^{C} e^{{w_{j'}}^T x_i}$：

利用对数求导公式：

$$\frac{\partial}{\partial w_k}\log Z=\frac{1}{Z}\cdot\frac{\partial Z}{\partial w_k}$$

其中：

$$Z=\sum_{j'=1}^{C} e^{w_{j'}^T x_i}$$

逻辑斯蒂回归（Logistic Regression）

- 第二项$\frac{\partial}{\partial w_k}\log\sum_{j'=1}^{C} e^{{w_{j'}}^T x_i}$：

对 Z 求导：

$$\frac{\partial}{\partial w_k} Z = \sum_{j'=1}^{C} \frac{\partial}{\partial w_k} e^{w_{j'}^T x_i}$$

对于 $j' = k$：

$$\frac{\partial}{\partial w_k} e^{w_k^T x_i} = x_i e^{w_k^T x_i}$$

对于 $j' \neq k$：

$$\frac{\partial}{\partial w_k} e^{w_{j'}^T x_i} = 0$$

逻辑斯蒂回归（Logistic Regression）

- 第二项$\frac{\partial}{\partial w_k}\log\sum_{j'=1}^{C} e^{w_{j'}{}^{T}x_i}$：

所以：

$$\frac{\partial}{\partial w_k}Z = x_i e^{w_k^T x_i}$$

带入对数求导：

$$\frac{\partial}{\partial w_k}\log Z = \frac{x_i e^{w_k^T x_i}}{\sum_{j'=1}^{C} e^{w_{j'}^T x_i}}$$

逻辑斯蒂回归（Logistic Regression）

➢ 合并：

$$\frac{\partial \log\left(\frac{e^{\boldsymbol{w}_j{}^T\boldsymbol{x}}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}}\right)}{\partial \boldsymbol{w}_k} = \frac{\partial}{\partial \boldsymbol{w}_k}\left(\boldsymbol{w}_j{}^T\boldsymbol{x}_i - \log\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}\right)$$

$$= \begin{cases} \boldsymbol{x}_i - \dfrac{\boldsymbol{x}_i e^{\boldsymbol{w}_k{}^T\boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}}, & j = k \\ -\dfrac{\boldsymbol{x}_i e^{\boldsymbol{w}_k{}^T\boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T\boldsymbol{x}_i}}, & j \neq k \end{cases}$$

$$= \begin{cases} (1 - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i, & j = k \\ -p(y_i = k|\boldsymbol{x}_i)\boldsymbol{x}_i, & j \neq k \end{cases}$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 计算梯度

$$\frac{\partial \log L(\boldsymbol{W})}{\partial \boldsymbol{w}_k} = \boxed{\frac{\partial}{\partial \boldsymbol{w}_k}} \sum_{i=1}^{N} \boxed{\sum_{j=1}^{C} 1(y_i = j)} \boxed{\log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)}$$

- 对每个类别的 $\boldsymbol{w}_k$ 分别计算导数

$$\frac{\partial \sum_{j=1}^{C} 1(y_i = j) \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)}{\partial \boldsymbol{w}_k} = \begin{cases} (1 - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i, & y_i = j = k \\ -p(y_i = k|\boldsymbol{x}_i)\boldsymbol{x}_i, & y_i = j \neq k \end{cases}$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 计算梯度

$$\frac{\partial \sum_{j=1}^{C} 1(y_i = j) \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right)}{\partial \boldsymbol{w}_k}$$

$$= \begin{cases} (1 - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i, & y_i = j = k \\ -p(y_i = k|\boldsymbol{x}_i)\boldsymbol{x}_i, & y_i = j \neq k \end{cases}$$

$$= \boxed{(1(y_i = k) - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i}$$ 误差×输入

$$\frac{\partial \log L(\boldsymbol{W})}{\partial \boldsymbol{w}_k} = \frac{\partial}{\partial \boldsymbol{w}_k} \sum_{i=1}^{N} \sum_{j=1}^{C} 1(y_i = j) \log\left(\frac{e^{\boldsymbol{w}_j{}^T \boldsymbol{x}_i}}{\sum_{j'=1}^{C} e^{\boldsymbol{w}_{j'}{}^T \boldsymbol{x}_i}}\right) = \sum_{i=1}^{N} (1(y_i = k) - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i$$

逻辑斯蒂回归（Logistic Regression）

■ 最大似然估计：梯度上升

- 梯度上升，直至收敛

$$\boldsymbol{w}_k := \boldsymbol{w}_k + \alpha \sum_{i=1}^{N} (1(y_i = k) - \boxed{p(y_i = k|\boldsymbol{x}_i)})\boldsymbol{x}_i$$

基于softmax函数计算

- 随机梯度上升，直至收敛

$$\boldsymbol{w}_k := \boldsymbol{w}_k + \alpha \sum_{i=1}^{N'} (1(y_i = k) - p(y_i = k|\boldsymbol{x}_i))\boldsymbol{x}_i$$

逻辑斯蒂回归（Logistic Regression）

其他优化方法求解最大似然估计

- 牛顿法
- 拟牛顿法(BFGS)
- 有限记忆拟牛顿法(L-BFGS)
- 共轭梯度法
- …

逻辑斯蒂回归（Logistic Regression）

■ 交叉熵损失函数→等价于负对数似然函数

给定**训练集** $D = \{(x_i, y_i)\}_{i=1}^{N}$，其中 y_i 是真实类别标签（one-hot 形式），交叉熵损失函数定义为：

$$L(W) = -\sum_{i=1}^{N}\sum_{j=1}^{C} 1(y_i = j)\log p(y = j|x_i; W)$$

其中：

- $1(y_i = j)$ 是指示函数，若样本 x_i 属于类别 j，则为 1，否则为 0。
- 该式表示真实类别的负对数概率的总和，即对于每个样本，我们只计算其真实类别对应的损失。

如果模型对正确类别的预测概率 **很高**（接近 1），则 $\log p(y = j)$ **接近 0**，损失很小。

如果模型对正确类别的预测概率 **很低**（接近 0），则 $\log p(y = j)$ **趋向负无穷**，损失很大，促使模型学习更好的参数。

逻辑斯蒂回归（Logistic Regression）

- 交叉熵损失函数→等价于负对数似然函数

给定**训练集** $D=\{(x_i,y_i)\}_{i=1}^{N}$，其中 y_i 是真实类别标签（one-hot 形式），交叉熵损失函数定义为：

$$L(W)=-\sum_{i=1}^{N}\sum_{j=1}^{C}1(y_i=j)\log p(y=j|x_i;W)$$

可用梯度下降法、牛顿法等优化方法求解！


Thank You!