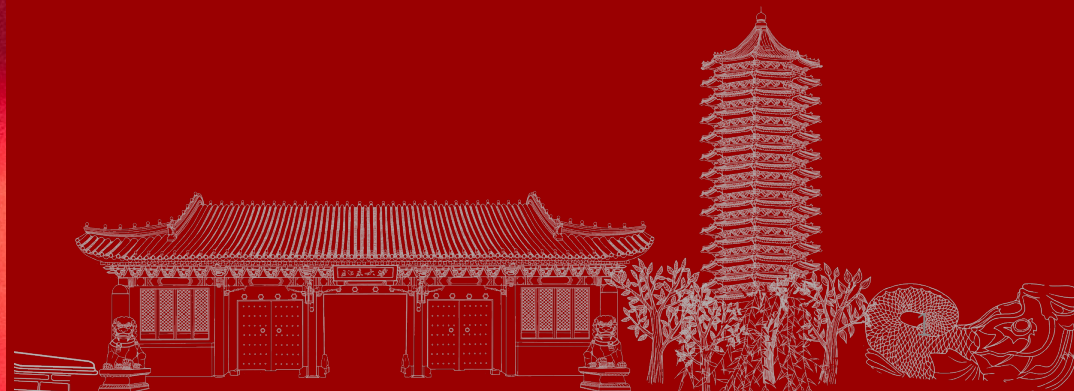


人工智能引论

10. 逻辑回归、多分类 与正则化

授课教师 周嘉欢

2024年03月25日



- **二分类问题**

- 逻辑回归 --- 对条件概率 $p(y|x)$ 建模并用最大似然估计参数

- **多分类问题**

- 扩展到多分类, 可用Softmax回归

- **正则化**

- L2正则化降低过拟合风险
 - 调整超参数, cross-validation

线性回归回顾

- 模型

$$f(x) = w^T x + b$$

- 输入 $x \in \mathbb{R}^d$
- 参数 $w \in \mathbb{R}^d, b \in \mathbb{R}$, 权重 (weight) 和 偏置 (bias)
- 输出 $f(x) \in \mathbb{R}$

- 损失函数

- 平方损失函数 (squared loss, L2 loss)
- $J(w, b) = \frac{1}{n} \sum_{i \in [n]} L(f(x_i), y_i) = \frac{1}{n} \sum_{i \in [n]} (w^T x_i + b - y_i)^2$
- 梯度下降训练

经验风险最小化框架

- Empirical Risk Minimization (ERM)

- 首先确定采用的模型 $f(x)$

- 比如, 线性模型 $f(x) = w^T x + b$

- 其次, 确定损失函数 (loss function) $L(f(x), y)$

- 比如, 平方损失函数 $L(f(x), y) = (f(x) - y)^2$

- 在训练集上最小化损失函数的平均值

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} L(f(x_i), y_i)$$

- 一般都可以采用梯度下降优化参数

- 大部分监督学习算法都遵循以上经验风险最小化框架 (ERM), 区别仅在于具体选择的 $f(x)$ 和 $L(f(x), y)$

- 线性回归 (Linear Regression)
 - 处理回归问题
 - 线性模型 $f(x) = w^T x + b$
 - 平方损失函数
- 逻辑回归 (Logistic Regression)
 - 处理二分类问题 (虽然名字叫回归)
 - 线性模型 $f(x) = w^T x + b$
 - 交叉熵损失函数 (cross entropy loss)

二分类 (Binary Classification)

- 标签只有两种
 - $y \in \{-1, 1\}$, -1代表负类 (negative class), 1代表正类 (positive class)
 - 如垃圾邮件识别, 1代表垃圾邮件, -1代表正常邮件

- 一般不直接让 $f(x) \in \mathbb{R}$ 拟合 $y \in \{-1, 1\}$

- 为了将实数输出转换为类别 $\{-1, 1\}$, 采用 $\text{sign}()$ 函数

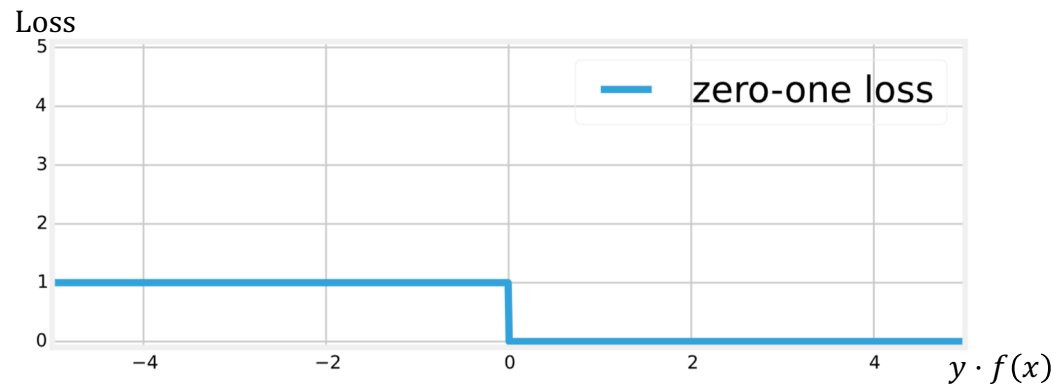
- $\text{sign}(f(x)) = \begin{cases} 1, & \text{if } f(x) > 0 \\ -1, & \text{if } f(x) < 0 \end{cases}$

- 使用什么损失函数?

- 最直接的目标, 最小化分类错误数, 使用零一损失函数 (zero-one loss)

- $L(f(x), y) = \begin{cases} 0, & \text{if } \text{sign}(f(x)) = y \\ 1, & \text{if } \text{sign}(f(x)) = -y \end{cases} \Leftrightarrow L(f(x), y) = \begin{cases} 0, & \text{if } y \cdot f(x) \geq 0 \\ 1, & \text{if } y \cdot f(x) < 0 \end{cases} \quad (f(x) = 0 \text{ 时默认0损失})$

- 但是, 零一损失函数是阶跃函数, 不可微 (non-differentiable) 且不连续 (non-continuous), 无法用梯度下降优化 (在0处不可微, 其余处梯度都为0, 无法提供下降方向)



最大似然框架 (Maximum Likelihood)

- 让我们使用最大似然估计 (Maximum Likelihood Estimation) 来推导适合二分类问题的损失函数
- 最大似然估计 (MLE) 的原则：
 - 对观测数据进行 (条件) 概率建模
 - 对机器学习, 每个观测数据即一个训练样本
 - 对判别式模型, 我们只建模 $p(y|x; \theta)$, θ 为模型参数
 - 通过最大化观测数据在给定概率模型下的似然 (例如: 把训练样本预测为正确标签的概率) 来估计模型参数
 - 如果训练样本互相独立 (独立同分布假设), 则最大似然估计可写为

$$\max_{\theta} \prod_{i \in [n]} p(y = y_i | x = x_i; \theta), \text{ 或简写为 } \max_{\theta} \prod_{i \in [n]} p(y_i | x_i; \theta)$$

- 但是, 大量概率连乘容易造成数值超出计算精度, 例如 $0.5^{1000} \approx 9 \times 10^{-302}$
- 解决方法为, 最大化对数似然 (log-likelihood)

$$\max_{\theta} \log\left(\prod_{i \in [n]} p(y_i | x_i; \theta)\right) \Leftrightarrow \max_{\theta} \sum_{i \in [n]} \log(p(y_i | x_i; \theta))$$

最大似然估计例子

- 给定一枚正反不均匀的硬币
- 已知抛了 n 次硬币，其中正面朝上的次数为 m 次
- 用最大似然估计(MLE)估算硬币正面概率
- 解法：
 - 首先对抛硬币事件进行概率建模，假设每次抛硬币正面朝上的概率为 p ，则反面为 $1 - p$
 - 在这个概率模型中， p 是我们唯一要估算的参数
 - 将观测数据在给定概率模型下的似然写出（假想观测数据为“正反反正正正反正反正正反正反正...”），其中正面 m 次，反面 $n-m$ 次）：

$$\text{Likelihood} = p^m (1 - p)^{n-m}$$

- 最大化对数似然来估计 p

$$\max_p m \log p + (n - m) \log(1 - p)$$

- 让梯度为0，得到 $\frac{m}{p} - \frac{n-m}{1-p} = 0$ ，求得 $p = m/n$



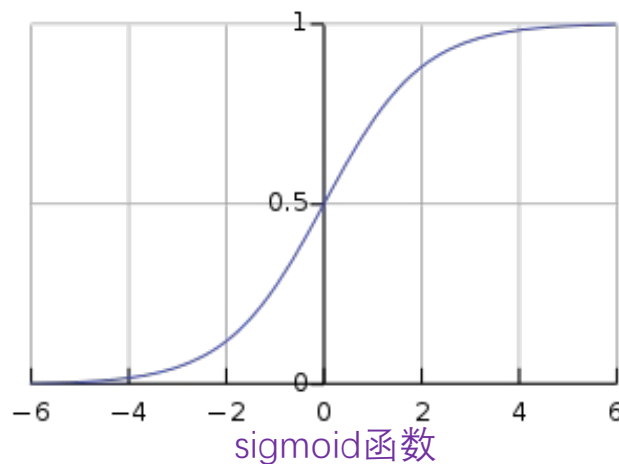
逻辑回归的最大似然估计

- 首先对 $p(y|x; \theta)$ 建模

- 已有线性模型 $f(x) = w^T x + b$, 只需要把它转化成正类的概率
- 采用sigmoid函数 $\sigma: (-\infty, +\infty) \rightarrow [0, 1]$

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

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



- 则 $p(y = 1|x; \theta) = p(y = 1|x; w, b) = \sigma(f(x)) = \sigma(y \cdot f(x))$
- 自然的, $p(y = -1|x; w, b) = 1 - \sigma(f(x)) = \sigma(-f(x)) = \sigma(y \cdot f(x))$
- 说明, 不论 y 取1/-1, 都有 $p(y|x; w, b) = \sigma(y \cdot f(x))!$

逻辑回归的最大似然估计

- 在训练集上最大化对数似然

- $\max_{w,b} \sum_{i \in [n]} \log[p(y_i|x_i; w, b)] = \sum_{i \in [n]} \log[\sigma(y_i \cdot f(x_i))]$

- 代入 $f(x_i) = w^T x_i + b$, 得到最优化问题

$$\begin{aligned} \max_{w,b} \sum_{i \in [n]} \log[\sigma(y_i(w^T x_i + b))] &= \sum_{i \in [n]} \log\left[\frac{1}{1+e^{-y_i(w^T x_i + b)}}\right] \\ &= -\sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}] \end{aligned}$$

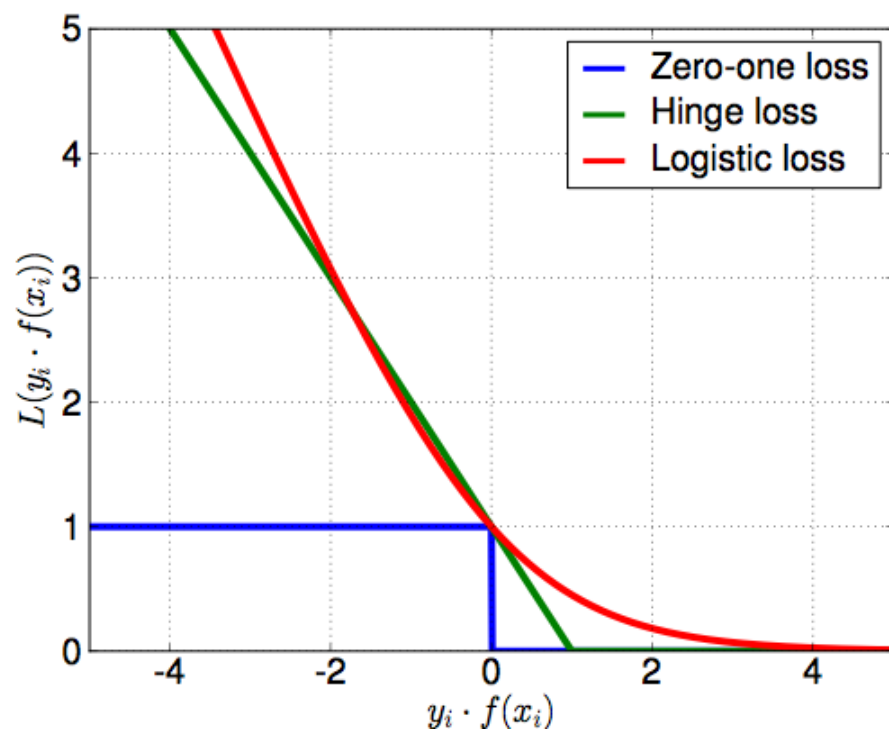
- 写成最小化平均损失函数的形式（逻辑回归的ERM形式）：

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}]$$

- 由最大对数似然推出的损失函数 $L(f(x_i), y_i) = \log[1 + e^{-y_i(w^T x_i + b)}]$ 称为交叉熵损失 (Cross Entropy Loss), 有时也直接称为 Logistic/Log Loss

替代损失函数视角

- 除了从最大似然估计视角推导逻辑回归，我们也可以从替代损失函数(substitute loss) 的视角看待逻辑回归



Logistic loss: $L(f(x_i), y_i) = \log[1 + e^{-y_i f(x_i)}]$

- 逻辑回归的交叉熵损失函数（红线）是零一损失的上界 (upper bound)
- 交叉熵损失函数可微、连续、且是凸函数 (convex)，因此容易优化
- 最小化上界同样可以最小化零一损失函数，即分类错误数
- 因此称为替代损失函数
- 绿线为合页损失函数 (Hinge Loss), $L(f(x_i), y_i) = \max(0, 1 - y_i f(x_i))$, 另一种常见的替代损失函数



- 使用梯度下降求解如下最优化问题

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}]$$

- 对目标 $J(w, b) = \frac{1}{n} \sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}]$ 求梯度

- $\frac{\partial J(w,b)}{\partial w} = -\frac{1}{n} \sum_{i \in [n]} \frac{e^{-y_i(w^T x_i + b)}}{1 + e^{-y_i(w^T x_i + b)}} y_i x_i = -\frac{1}{n} \sum_{i \in [n]} [1 - p(y_i | x_i; w, b)] y_i x_i$

- $\frac{\partial J(w,b)}{\partial b} = -\frac{1}{n} \sum_{i \in [n]} \frac{e^{-y_i(w^T x_i + b)}}{1 + e^{-y_i(w^T x_i + b)}} y_i = -\frac{1}{n} \sum_{i \in [n]} [1 - p(y_i | x_i; w, b)] y_i$

- 把样本 i 预测为其真实标签的概率越接近1（说明已经充分拟合该样本），它对梯度的贡献越小

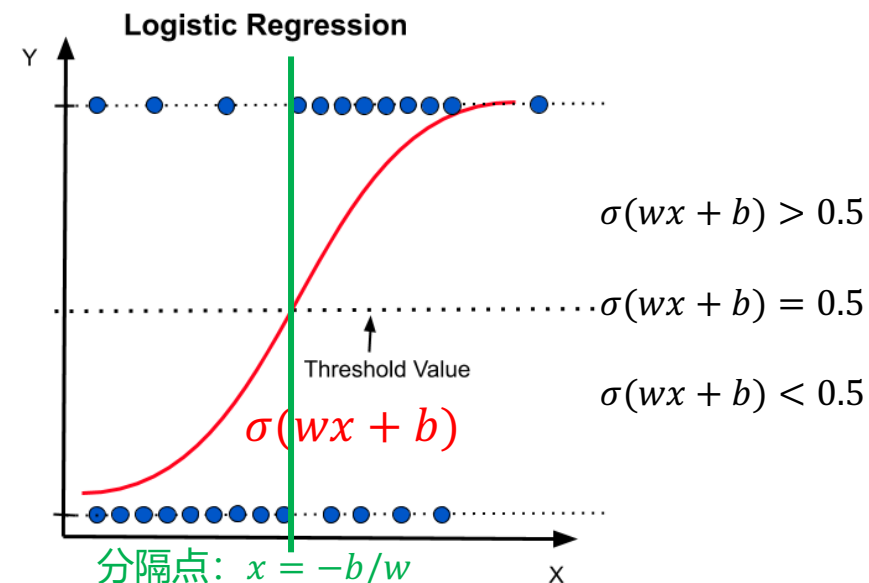
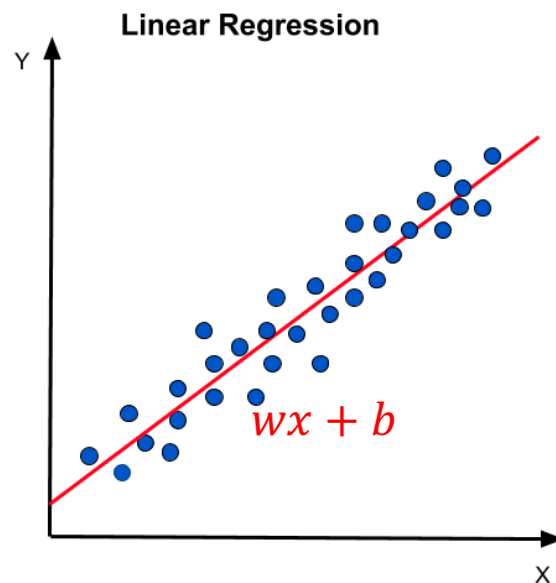
- 迭代更新 w, b 直到 $J(w, b)$ 无法再下降或达到预设的最大次数

$$w \leftarrow w - \alpha \cdot \frac{\partial J(w,b)}{\partial w}, \quad b \leftarrow b - \alpha \cdot \frac{\partial J(w,b)}{\partial b}$$

线性回归 vs 逻辑回归

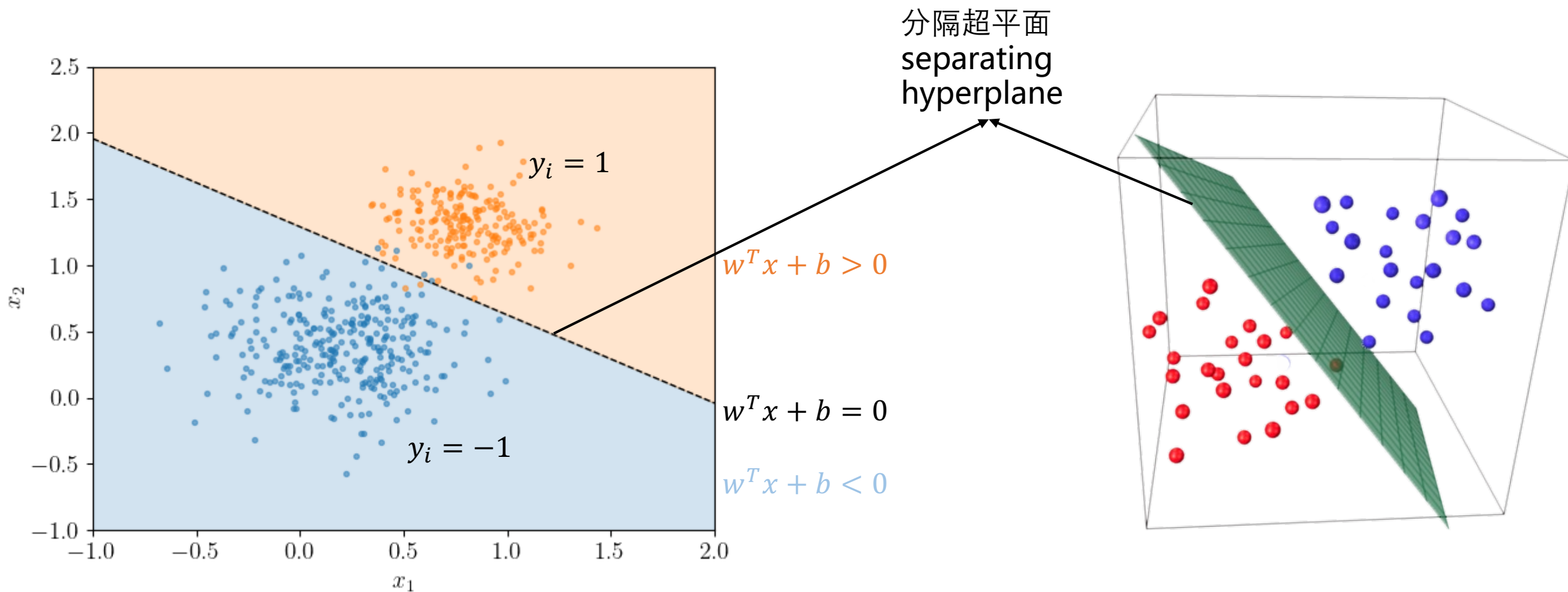
	线性回归	逻辑回归
任务	回归	二分类
模型	线性模型 $f(x) = w^T x + b$	线性模型 $f(x) = w^T x + b$
输出	$f(x)$	$\text{sign}(f(x))$ 直接输出 $\{-1, 1\}$; 或 $\sigma(f(x))$ 输出取1的概率
损失函数	平方损失 $(f(x_i) - y_i)^2$	交叉熵损失 $\log[1 + e^{-y_i(w^T x_i + b)}]$

一维可视化



线性回归 vs 逻辑回归

逻辑回归的高维可视化



逻辑回归总结

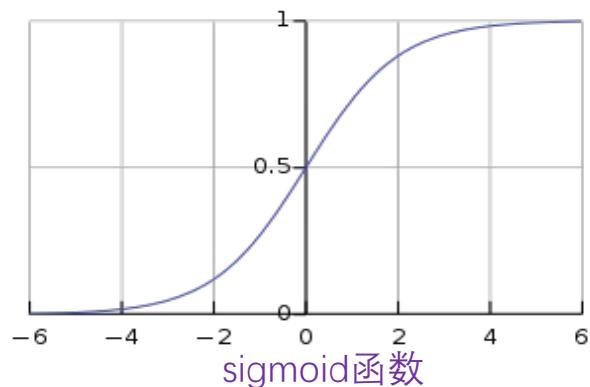
- 模型

$$f(x) = w^T x + b$$

- 输入 $x \in \mathbb{R}^d$, 参数 $w \in \mathbb{R}^d, b \in \mathbb{R}$

- $\sigma(f(x)) = \frac{1}{1+e^{-f(x)}}$ 将实值 $f(x)$ 转换为取正类 ($y = 1$) 的概率,

- $1 - \sigma(f(x))$ 转换为取负类 ($y = -1$) 的概率



- 最大似然估计

- $\max_{w,b} \sum_{i \in [n]} \log[p(y_i | x_i; w, b)] = \sum_{i \in [n]} \log[\sigma(y_i \cdot f(x_i))]$

- 等价于最小化交叉熵损失

- $\min_{w,b} \frac{1}{n} \sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}]$

多分类问题 (Multiclass Classification)

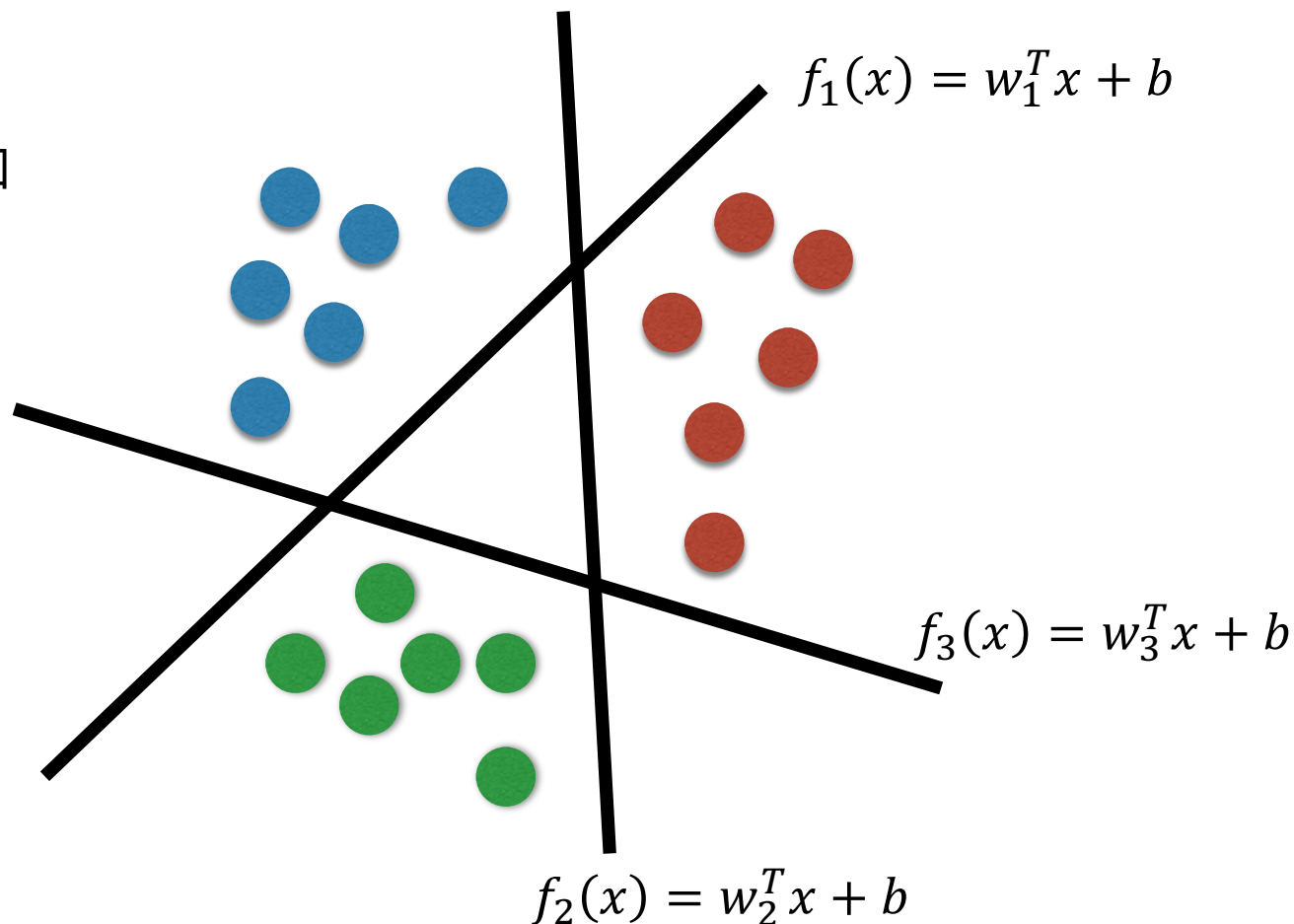
- 如何处理类别数 $K > 2$ 的情况?

- one vs rest

- 对K分类问题训练K个二分类器 (如逻辑回归)
 - 第k个二分类器将第k类当成正类, 其余所有类别都当成负类
 - $y = \operatorname{argmax}_{k \in [K]} f_k(x)$

- 问题:

- 当类别数 K 非常大时, 分别训练K个二分类器代价太高
 - K个二分类器互相独立, 无法统一成一个模型



Softmax 回归

- 专门解决多分类问题（虽然仍然叫回归）
- K 分类问题： $y \in \{1, 2, \dots, K\} = [K], x \in \mathbb{R}^d$
- 共同训练 K 个模型 $f_k(x) \in \mathbb{R}, k \in [K]$
- 将模型输出 $f_k(x)$ 转化为取第 k 类的概率
 - 不能使用 sigmoid 函数，因为需满足 $\sum_{k \in [K]} p(y = k|x) = 1$
 - 解决方法：使用 softmax 函数，使概率归一化

$$p(y = k|x) = \frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}}$$

- e 指数的放大效应会使得如果 $f_k(x) \gg f_j(x), \forall j \neq k$, 则 $p(y = k|x) \approx 1$

Softmax 回归

k	1	2	3	4
$f_k(x)$	2.0	1.0	1.0	0.5
$\frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}}$	0.5105	0.1878	0.1878	0.1139

k	1	2	3	4
$f_k(x)$	3.0	1.0	0.5	0.5
$\frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}}$	0.7695	0.1041	0.0632	0.0632

k	1	2	3	4
$f_k(x)$	10.0	1.0	1.0	1.0
$\frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}}$	0.9996	0.0001	0.0001	0.0001

Softmax 回归

- 预测 y 取第 k 类概率: $p(y = k|x) = \frac{e^{f_k(x)}}{\sum_{j \in [K]} e^{f_j(x)}}$
- 满足归一化条件: $\sum_{k \in [K]} p(y = k|x) = 1$
- 使用最大对数似然估计参数
 - 假设 $f_k(x)$ 的参数为 θ_k , 最大化训练集 $\{(x_i, y_i) | i \in [n]\}$ 的对数似然:

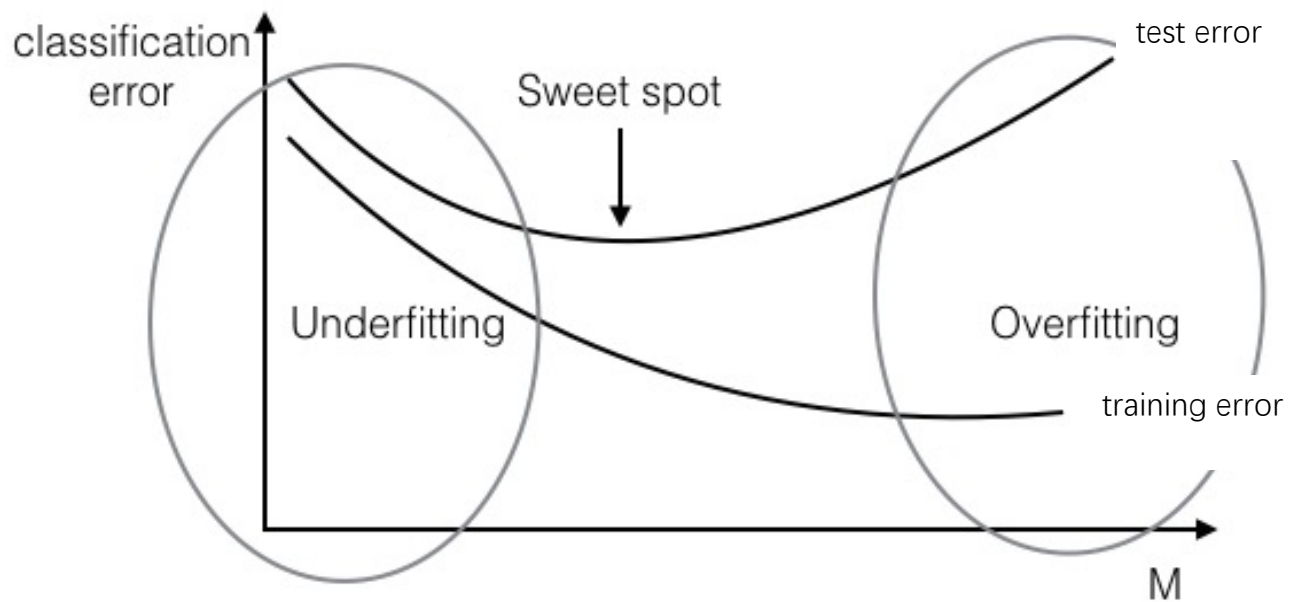
$$\max_{\{\theta_k\}} \sum_{i \in [n]} \log[p(\mathbf{y}_i | x_i)] = \sum_{i \in [n]} \log \left[\frac{e^{f_{\mathbf{y}_i}(x_i)}}{\sum_{j \in [K]} e^{f_j(x_i)}} \right]$$

- 等价于最小化交叉熵损失:

$$\min_{\{\theta_k\}} -\frac{1}{n} \sum_{i \in [n]} \log \left[\frac{e^{f_{y_i}(x_i)}}{\sum_{j \in [K]} e^{f_j(x_i)}} \right] = \frac{1}{n} \sum_{i \in [n]} (\log[\sum_{j \in [K]} e^{f_j(x_i)}] - f_{y_i}(x_i))$$

过拟合现象回顾

- 过拟合 (overfitting): 测试误差远远大于训练误差
 - 错把训练样本中找到的特殊规律当做了普遍规律——应避免这种现象



正则化 (Regularization)

- 实际中我们一般在损失函数后加一项一起优化:

$$\min_f \frac{1}{n} \sum_{i \in [n]} L(f(x_i), y_i) + \lambda \cdot R(f)$$

- $\lambda \cdot R(f)$ 称为正则化项 (regularization term), 用于惩罚过于复杂的模型
- $\lambda > 0$ 是个超参数 (hyperparameter), 预先指定, 不随参数优化
- 为什么需要正则化?
 - 防止过拟合 (overfitting), 即测试误差远远高于训练误差
 - 常见过拟合原因:
 - 某几个特征维度 j 支配 (dominate) 了预测, 即这些维度的权重 w_j 过大
 - 输入数据中存在大量没用的特征维度, 但仍然赋予了它们非零的权重

正则化 (Regularization)

- 常见过拟合原因：
 - 某几个特征维度 j 支配 (dominate) 了预测，即这些维度的权重 w_j 过大
 - 解决方法：L2 正则化 $R(f) = \|w\|_2^2 = w^T w = \sum_{j \in [d]} w_j^2$ ，简写为 $\|w\|^2$
 - 作用： w_j^2 会放大较大的权重，用于惩罚少数过大的权重维度，使权重分配更平均
 - 输入数据中存在大量没用的特征维度，但仍然赋予了它们非零的权重
 - 解决方法：L1 正则化 $R(f) = \|w\|_1 = \sum_{j \in [d]} |w_j|$
 - 作用：鼓励稀疏的 w ，即 w 中大部分维度为零，仅有少数维度非零（具体原因超出本课程范围）

带正则化的回归

- 岭回归 (Ridge Regression)

- 线性回归 + L2 正则化

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} (w^T x_i + b - y_i)^2 + \lambda \|w\|^2$$

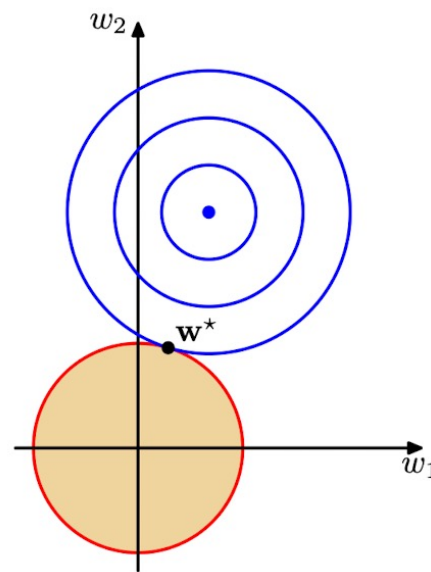
- 梯度

$$\frac{\partial J(w,b)}{\partial w} = \frac{2}{n} \sum_{i \in [n]} (w^T x_i + b - y_i) x_i + 2\lambda w \in \mathbb{R}^d$$

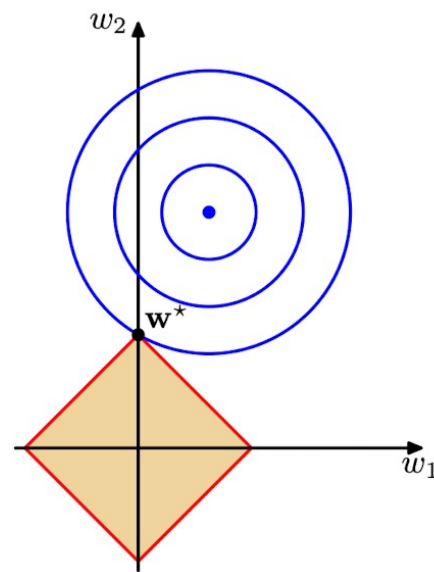
- Lasso 回归

- 线性回归 + L1 正则化

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} (w^T x_i + b - y_i)^2 + \lambda \|w\|_1$$



Ridge regression



Lasso regression

带正则化的分类

- 逻辑回归 (完整形式)

- 交叉熵损失 + L2 正则化

$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} \log[1 + e^{-y_i(w^T x_i + b)}] + \lambda \|w\|^2$$

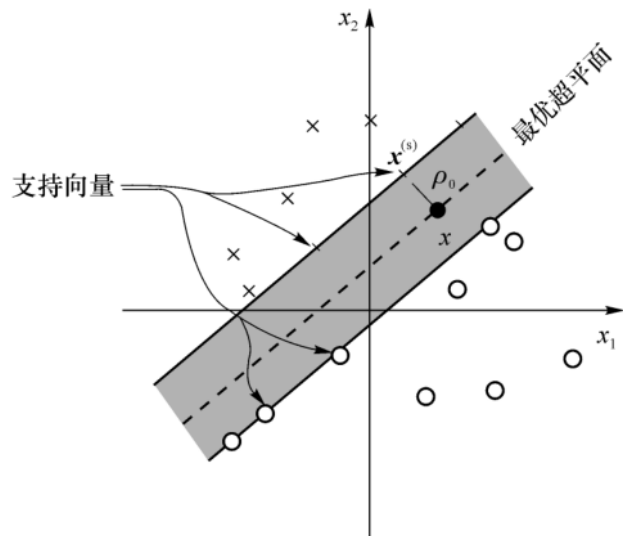
- 梯度

$$\frac{\partial J(w,b)}{\partial w} = -\frac{1}{n} \sum_{i \in [n]} [1 - p(y_i | x_i; w, b)] y_i x_i + 2\lambda w \in \mathbb{R}^d$$

- 支持向量机 (Support Vector Machine, SVM) (最大间隔准则)

- 合页损失 (Hinge Loss) + L2 正则化

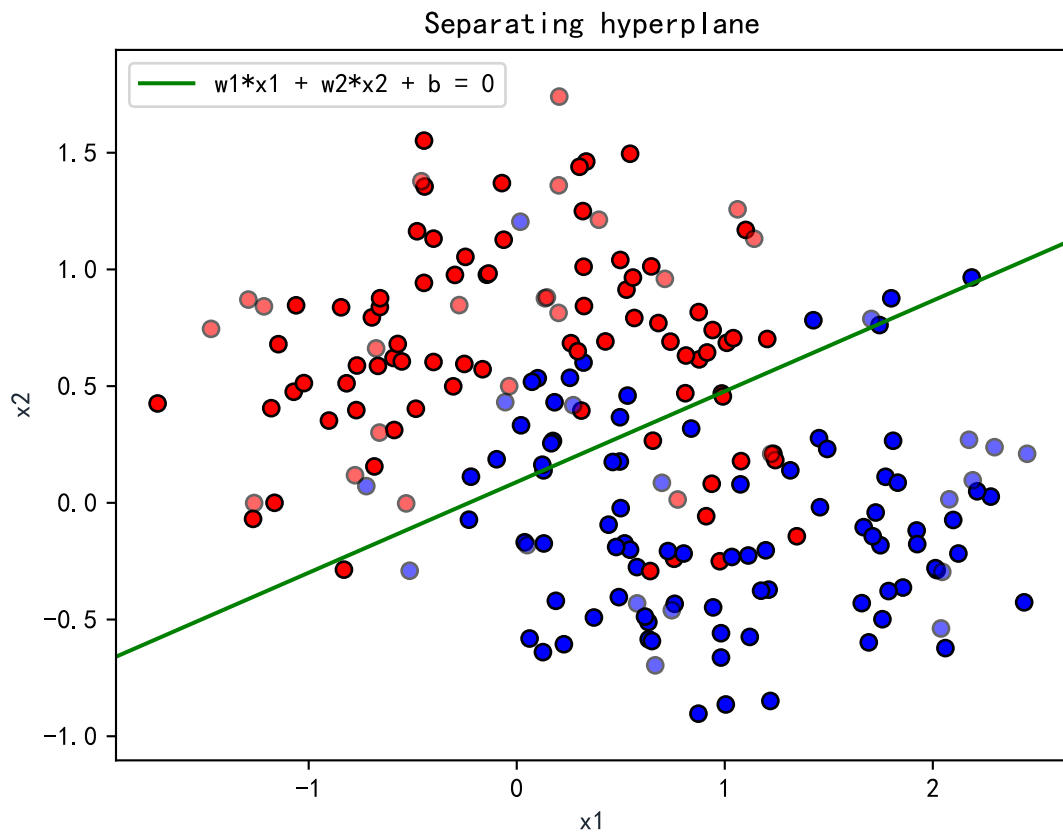
$$\min_{w,b} \frac{1}{n} \sum_{i \in [n]} \max(0, 1 - y_i(w^T x_i + b)) + \lambda \|w\|^2$$



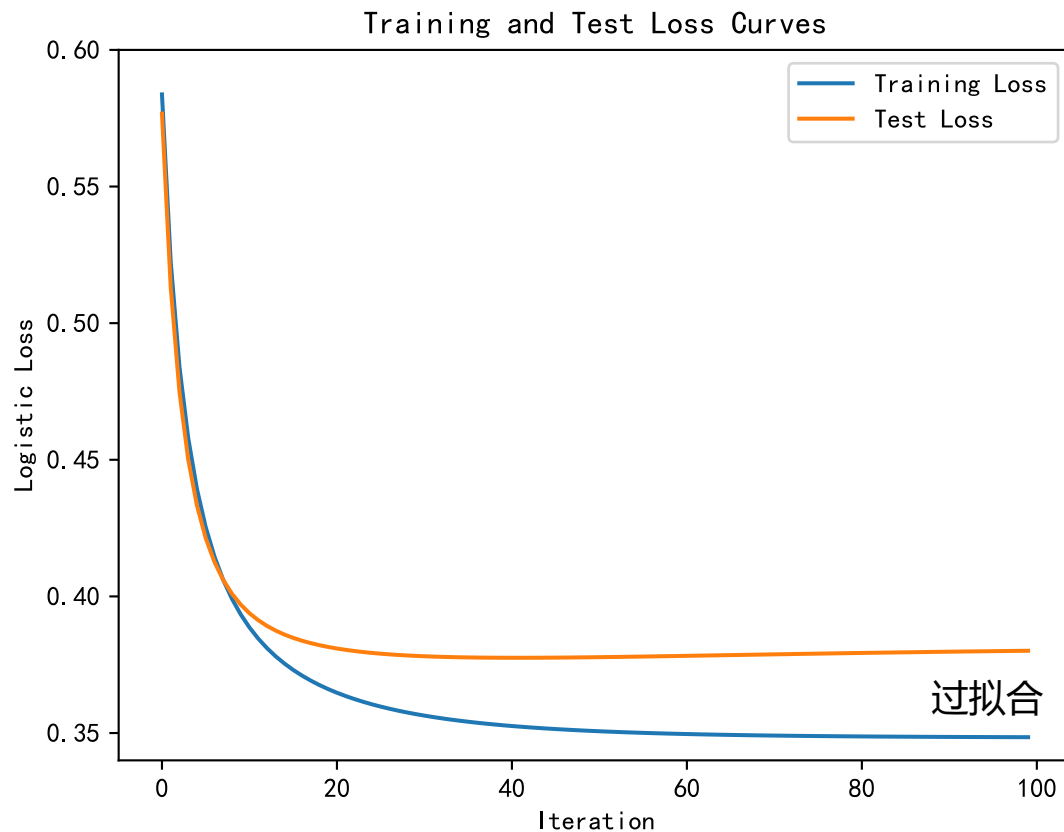
逻辑回归例子

- 不使用L2正则化

分隔超平面



训练/测试损失曲线

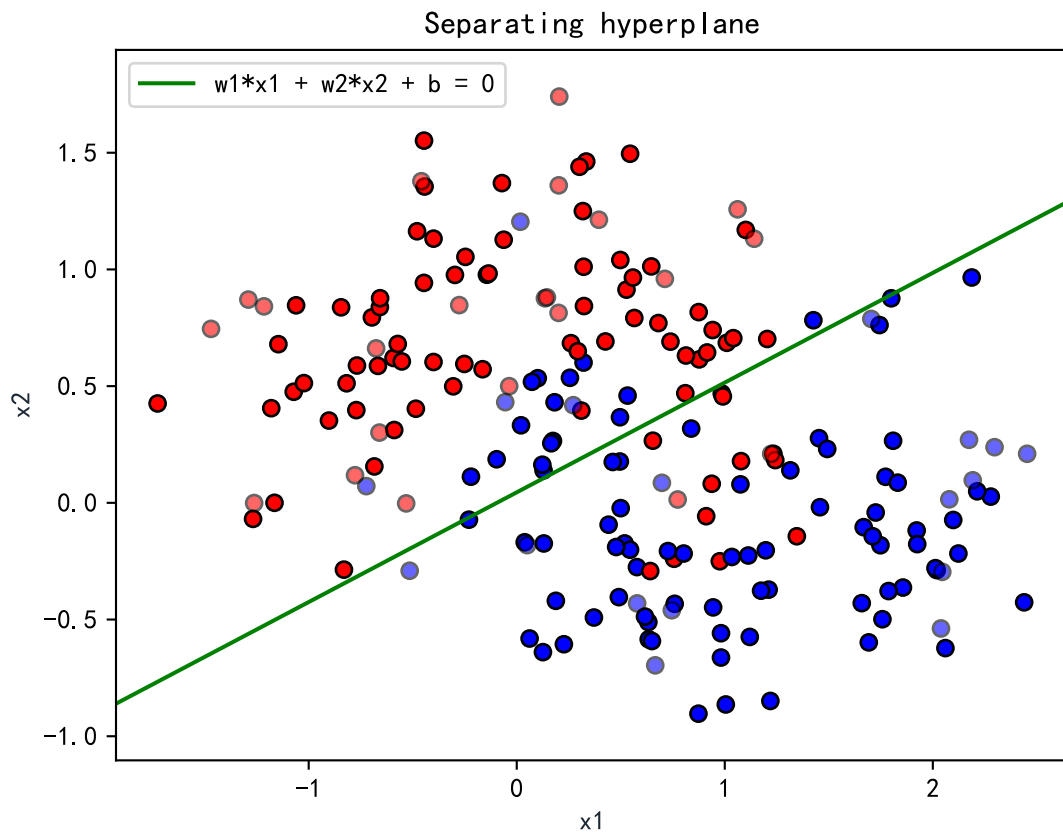


Training Loss = 0.348
Test Loss = 0.380

逻辑回归例子

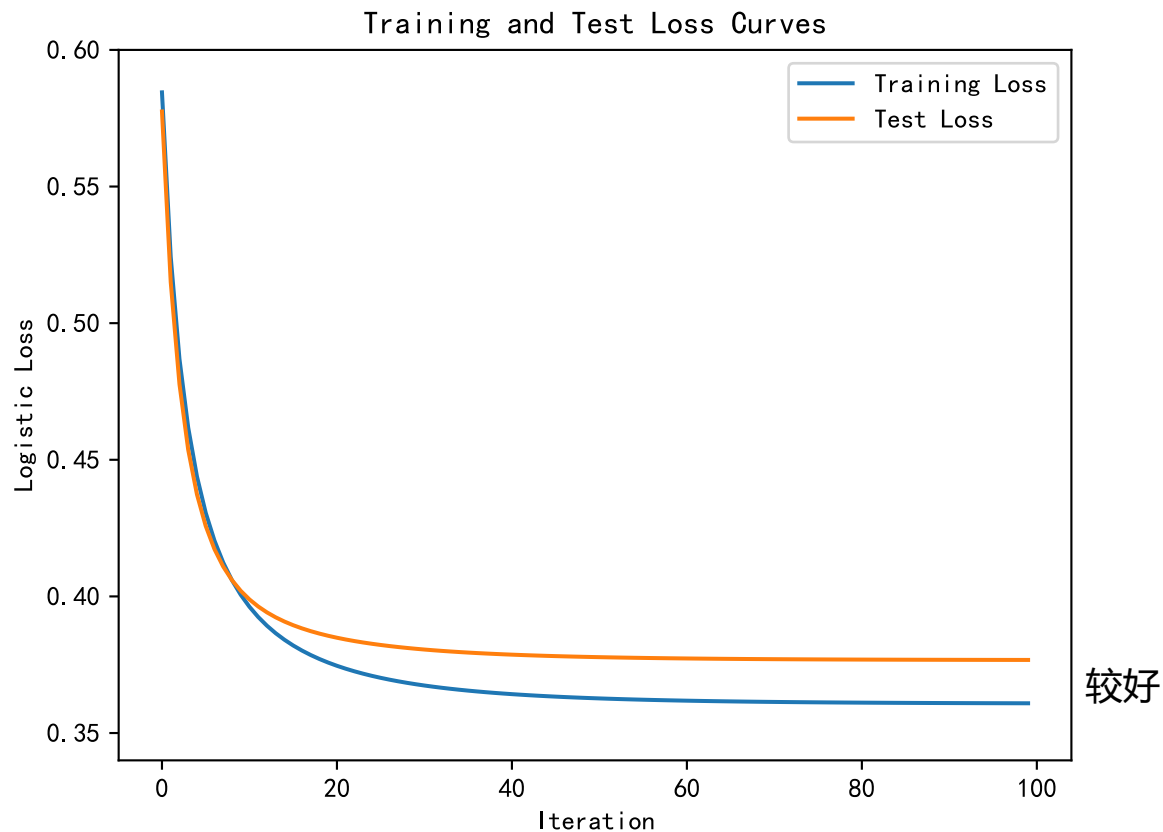
- L2正则化 $\lambda = 0.01$

分隔超平面



$$w_1 = 1.132, w_2 = -2.407$$

训练/测试损失曲线



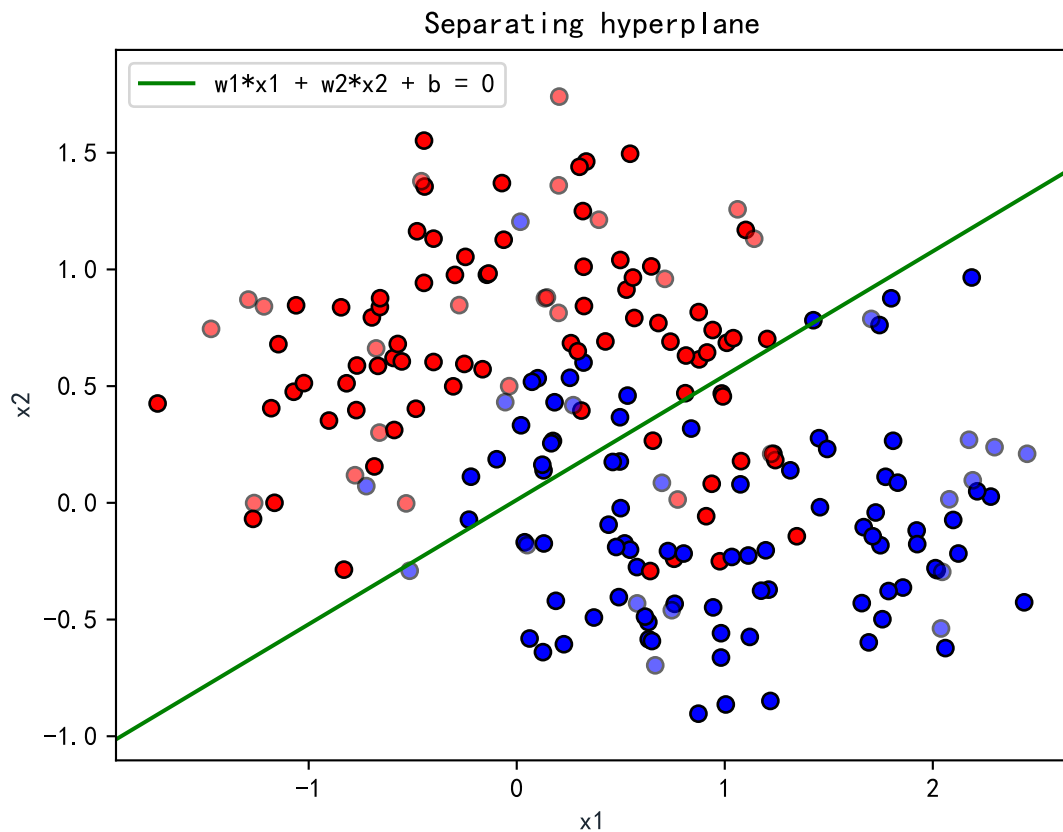
$$\begin{aligned} \text{Training Loss} &= 0.361 \\ \text{Test Loss} &= 0.377 \end{aligned}$$

较好

逻辑回归例子

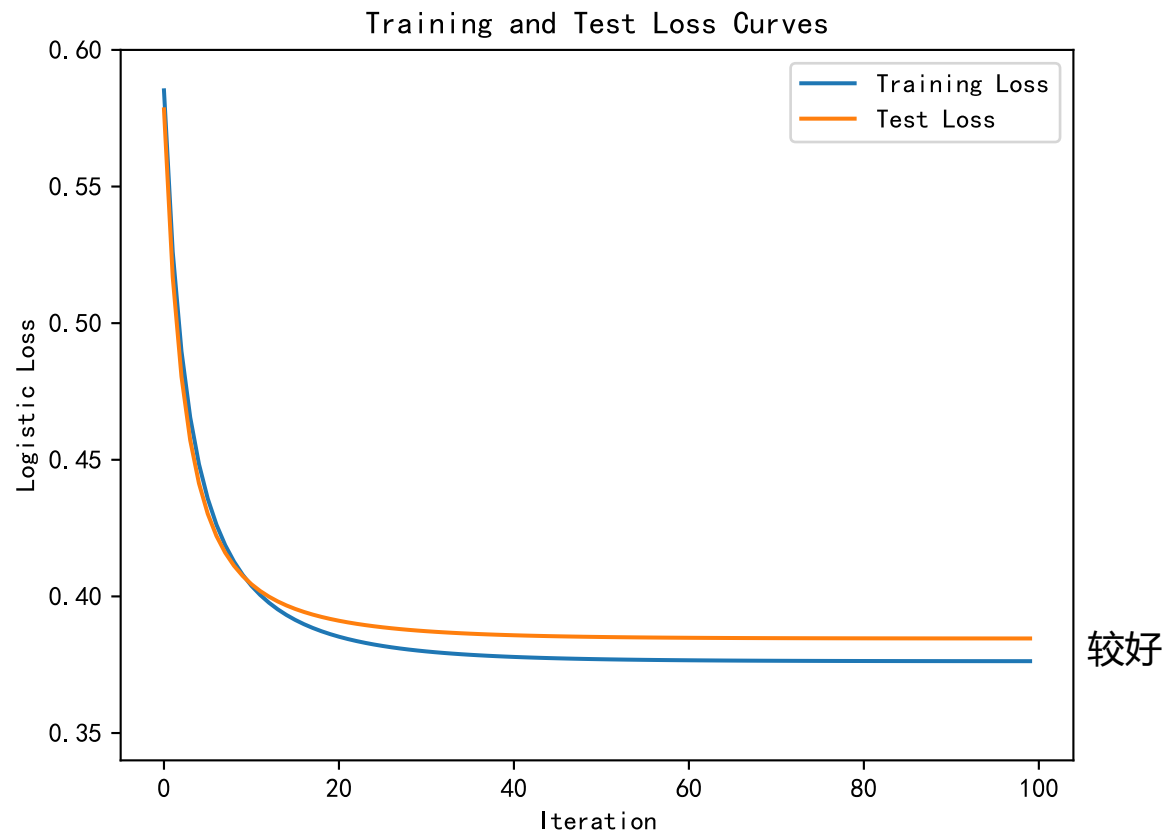
- L2正则化 $\lambda = 0.02$

分隔超平面



$$w_1 = 1.044, w_2 = -1.960$$

训练/测试损失曲线

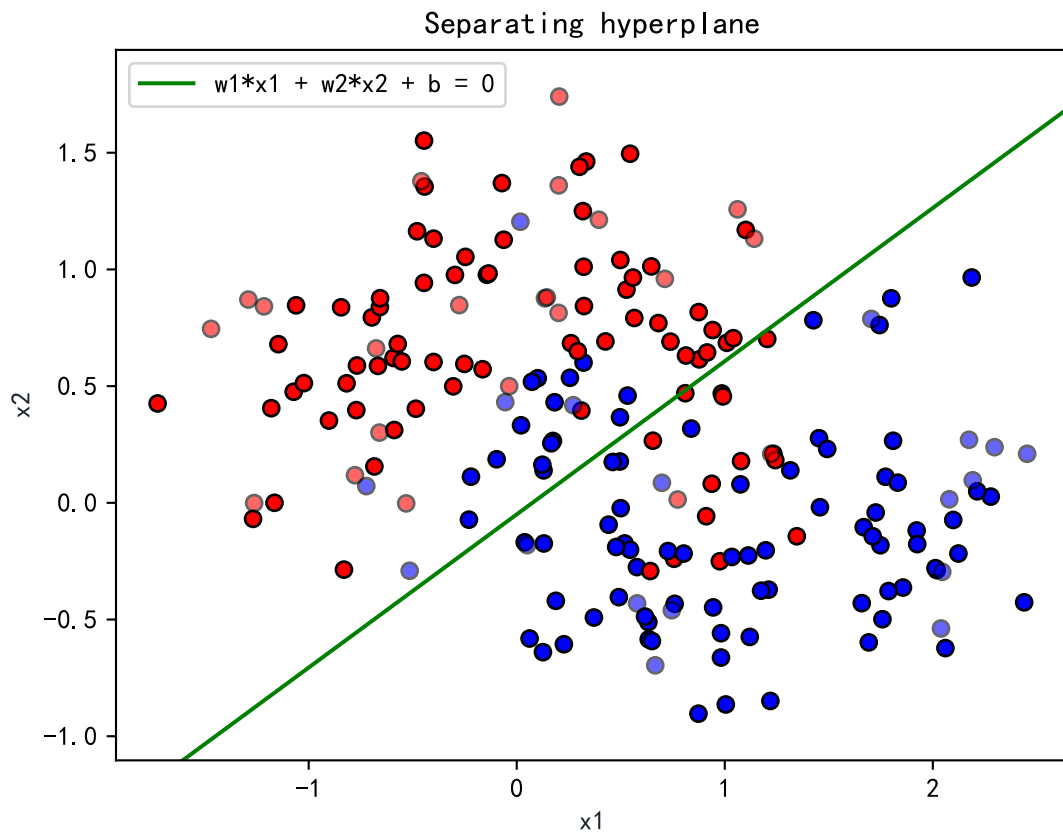


$$\begin{aligned} \text{Training Loss} &= 0.376 \\ \text{Test Loss} &= 0.385 \end{aligned}$$

逻辑回归例子

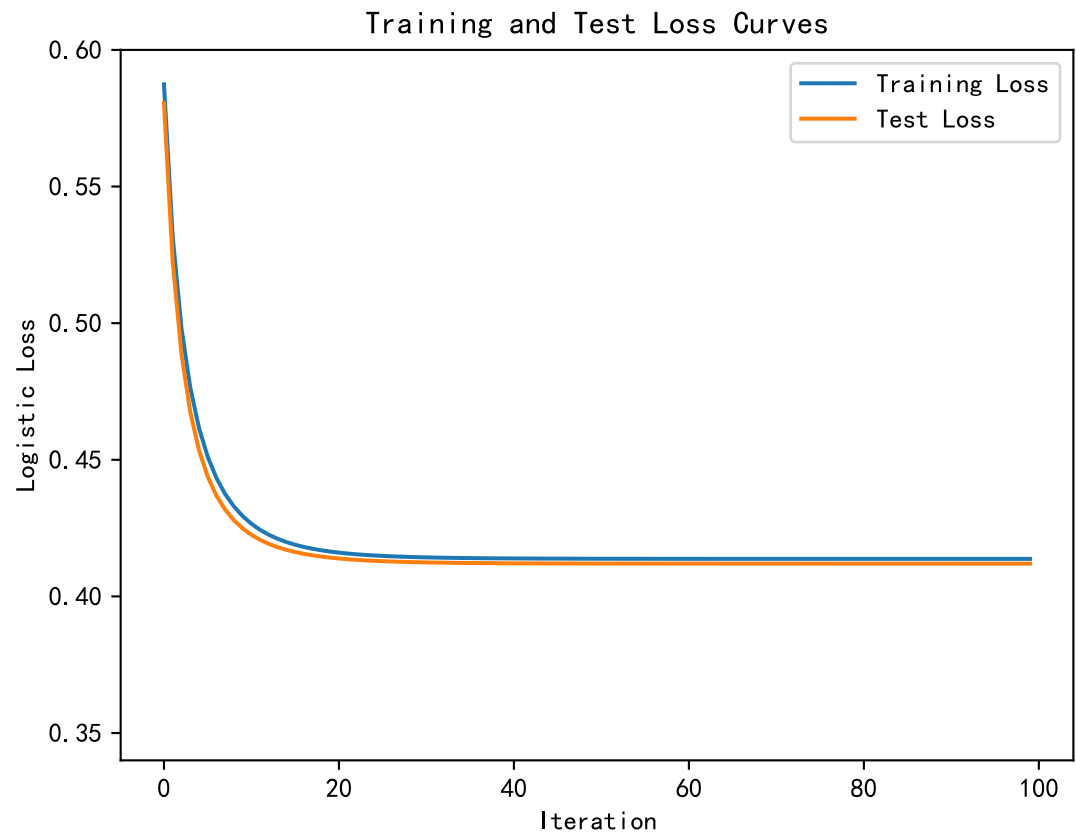
- L2正则化 $\lambda = 0.05$

分隔超平面



$$w_1 = 0.882, w_2 = -1.344$$

训练/测试损失曲线



$$\begin{aligned} \text{Training Loss} &= 0.414 \\ \text{Test Loss} &= 0.412 \end{aligned}$$

欠拟合

超参数与模型选择

- 训练过程中一般涉及一些超参数 (hyperparameters)
 - 梯度下降的学习率 (learning rate) α , 正则化项的系数 λ 等
 - 此外, 使用哪种模型、哪种损失函数、哪种正则化等也可看作超参数
 - 找到所有可能的超参数组合, 例如 $\alpha \in \{0.1, 0.01, 0.001\}$, $\lambda \in \{10, 1, 0.1\}$, $\text{Loss} \in \{\text{Hinge}, \text{Cross Entropy}\}$, 则其中一组超参数组合为 $\{\alpha: 0.01, \lambda: 0.1, \text{Loss: Hinge}\}$
- 使用验证集 (validation set) 来选择最优超参数
 - 在训练集、测试集之外引入验证集, 通常按照训练集/验证集/测试集分别 80%/10%/10% 的比例划分
 - 对每一组超参数的组合, 在训练集上训练参数, 在验证集上验证模型误差
 - 遍历所有可能的超参数组合, 选择在验证集上误差最小的模型和超参数
 - 使用选定的模型和超参数在测试集上最终测试模型误差, 估计模型泛化能力

K-折交叉验证 (K-fold Cross-Validation)



- 有时总数据量比较少，10%的验证集不足以准确地验证模型性能
- K-折交叉验证：
 - 将测试集外的所有数据随机分为 K 份
 - 对一组给定的超参数组合：
 - 每次使用 $K-1$ 份数据训练模型，用 1 份验证模型误差
 - 将 K 次验证的模型误差取平均来衡量该组超参数的好坏
 - 遍历所有可能的超参数组合
 - 返回平均误差最低的超参数组合，在测试集上最终测试模型性能

总结

- **二分类问题**

- 一种新的参数估计框架：最大似然估计 (MLE)
- 逻辑回归 --- 用 $\sigma(f(x))$ 对条件概率 $p(y = 1|x)$ 建模
- 用MLE推导交叉熵损失函数

- **多分类问题**

- 扩展到多分类，使用Softmax建模 $p(y = k|x)$
- 使用MLE推导交叉熵损失函数

- **正则化**

- 降低过拟合风险
- L2正则化, L1正则化
- 使用验证集调整超参数, cross-validation

- **下周：决策树**

期中考试安排

4月22日，周一

地点：待定

谢谢



北京大学
PEKING UNIVERSITY

