第3章：线性学习

蒋良孝 |  中国地质大学（武汉）

机器学习与数据挖掘团队

ljiang@cug.edu.cn

http://www.escience.cn/people/jlx/

本章内容

一、线性回归

二、广义线性回归

三、逻辑斯蒂回归

四、多分类学习

一、线性回归

- **提及线性学习，我们首先会想到线性回归。回归跟分类的区别在于要预测的目标函数是连续值。**
- 给定由m个属性描述的样本$\boldsymbol{x}=(x_1;x_2;\ldots;x_m)$，其中$x_i$是$\boldsymbol{x}$在第$i$个属性上的取值，线性回归（linear regression）试图学得一个通过属性值的线性组合来进行预测的函数：

$$f(\boldsymbol{x})=w_1x_1+w_2x_2+\ldots+w_mx_m+b$$

- 一般用向量的形式写成：

$$f(\boldsymbol{x})=\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b$$

其中$\boldsymbol{w}=(w_1;w_2;\ldots;w_m)$。

一、线性回归

- 给定训练数$D = \{(\boldsymbol{x}_1, y_1), (\boldsymbol{x}_2, y_2), \ldots, (\boldsymbol{x}_i, y_i), \ldots, (\boldsymbol{x}_n, y_n)\}$
  其中$\boldsymbol{x}_i = (x_{i1}; x_{i2}; \ldots; x_{im})$, $y_i \in \mathbb{R}$

- 可用最小二乘法 (least square method) 对$\mathbf{w}$ 和b 进行估计。

一、线性回归

- 下面以一元线性回归为例，来详细讲解 w 和 b 的最小二乘法估计

$$f(x_i) = wx_i + b, \text{ 使得 } f(x_i) \simeq y_i$$

- 最小二乘法就是基于预测值和真实值的均方差最小化的方法来估计参数 w 和 b：

$$\begin{aligned}(w^*, b^*) &= \arg\min_{(w,b)} \sum_{i=1}^{n} \left(f\left(x_i\right) - y_i\right)^2 \\ &= \arg\min_{(w,b)} \sum_{i=1}^{n} \left(y_i - wx_i - b\right)^2\end{aligned}$$

一、线性回归

- 最小化均方误差

$$E_{(w,b)} = \sum_{i=1}^{n} (y_i - wx_i - b)^2$$

- 分别对 w 和 b 求偏导，可得

$$\frac{\partial E_{(w,b)}}{\partial w} = 2\left(w\sum_{i=1}^{n} x_i^2 - \sum_{i=1}^{n} (y_i - b)\, x_i \right)$$

$$\frac{\partial E_{(w,b)}}{\partial b} = 2\left(nb - \sum_{i=1}^{n} (y_i - wx_i) \right)$$

一、线性回归

- 令上两式为零可得到 w 和 b 最优解的闭式(closed-form）解：

$$w = \frac{\sum_{i=1}^{n} y_i \left(x_i - \bar{x}\right)}{\sum_{i=1}^{n} x_i^2 - \frac{1}{n}\left(\sum_{i=1}^{n} x_i\right)^2}$$

$$b = \frac{1}{n}\sum_{i=1}^{n} \left(y_i - w x_i\right)$$

其中，$\bar{x} = \frac{1}{n}\sum_{i=1}^{n} x_i$

二、广义线性回归

- 只要学到w和b，模型就可以确定；对于任意的测试样例$\boldsymbol{x}$，只要输入它的属性值，就可以输出它的预测值。
- 线性回归假定输入空间到输出空间的函数映射成线性关系，但现实应用中，很多问题都是非线性的。为拓展其应用场景，我们可以将线性回归的预测值做一个非线性的函数变化去逼近真实值，这样得到的模型统称为广义线性回归 (generalized linear regression)：

$$y = g\left(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b\right)$$

其中$g(\cdot)$称为联系函数（link function）。

二、广义线性回归

- 理论上，联系函数 $g(\cdot)$ 可以是任意函数，比如当 $g(\cdot)$ 被指定为指数函数时，得到的回归模型称为对数线性回归：

$$y = e^{w^{\mathrm{T}}x+b}$$

- 之所以叫对数线性回归，是因为它将真实值的对数作为线性回归逼近的目标，即：

$$\ln y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$$

三、逻辑斯蒂回归

- 前面的内容都是在讲解如何利用线性模型进行回归学习，完成回归任务。但如果我们要做的是分类任务该怎么办？
- 为了简化，我们先考虑二分类任务，其输出标记 $y \in \{0,1\}$ ，但线性回归模型产生的预测值 $z = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$ 是实值，因此，我们需将实值z转换为0/1值。最容易想到的联系函数 $g(\cdot)$ 当然是单位阶跃函数：

$$y = \begin{cases} 0, & z < 0; \\ 0.5, & z = 0; \\ 1, & z > 0, \end{cases}$$

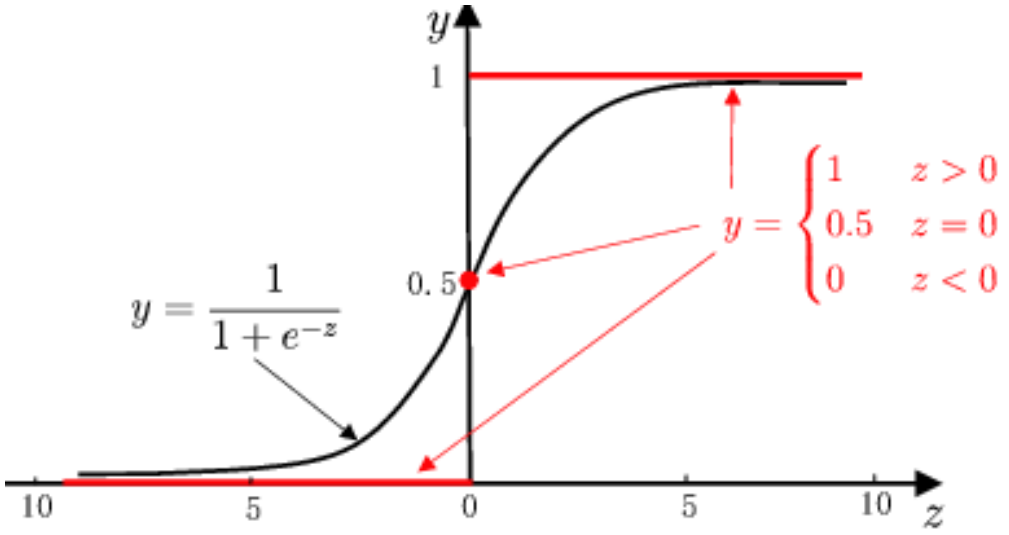
如预测值大于零就判为正例，
小于零就判为反例，
预测值为临界值零则可任意判别

三、逻辑斯蒂回归

- 但单位阶跃函数不连续，因此不能直接用作联系函数 $g(\cdot)$。于是我们希望找到能在一定程度上近似单位阶跃函数的替代函数，并希望它在临界点连续且单调可微。逻辑斯蒂函数(logistic function) 正是这样一个常用的替代函数：

$$y = \frac{1}{1+e^{-z}}$$

$$y = \frac{1}{1+e^{-(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)}}$$



单位阶跃函数与对数几率函数的比较

三、逻辑斯蒂回归

- 逻辑斯蒂(logistic function) 函数形似s，是Sigmoid函数的典型代表，它将z值转化为一个接近0或1的y值，并且其输出值在z=0附近变化很陡。
- 其对应的模型称为逻辑斯蒂回归(logistic regression)。需要特别说明的是，虽然它的名字是“回归”，但实际上却是一种分类学习方法。
- 逻辑斯蒂回归有很多优点：1）可以直接对分类可能性进行预测，将y视为样本x作为正例的概率；2）无需事先假设数据分布，这样就避免了假设分布不准确所带来的问题；3）是任意阶可导的凸函数，可直接应用现有数值优化算法求取最优解。

三、逻辑斯蒂回归

- 将y视为样本x属于正例的概率 $p(y=1 \mid \boldsymbol{x})$，根据逻辑斯蒂函数很容易得到：

$$y=\frac{1}{1+e^{-(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b)}}$$

$$p\left(y=1 \mid \boldsymbol{x}\right)=\frac{e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}{1+e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}$$

$$p\left(y=0 \mid \boldsymbol{x}\right)=\frac{1}{1+e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}$$

- 给定训练数据集 $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{n}$，可通过“极大似然法”(maximum likelihood method)来估计ω和b，即最大化样本属于其真实标记的概率（对数似然）：

$$\ell\left(\boldsymbol{w}, b\right)=\sum_{i=1}^{n} \ln p\left(y_i \mid \boldsymbol{x}_i ; \boldsymbol{w}, b\right)$$

三、逻辑斯蒂回归

- 逻辑斯蒂回归只能求解连续属性值问题，不能求解离散属性值问题，对于离散属性值的处理：
  - ✓ 若属性值之间存在“序”关系：通过连续化将其转化为连续值
  - ✓ 若属性值之间不存在“序”关系：通常可将k个属性值转换为k维向量

四、多分类学习

- 前面讲到的都是二分类学习任务，现实应用中常常会遇到多分类学习任务。
- 多分类学习方法
  - ✓ 二分类学习方法推广到多类
  - ✓ 利用二分类学习器解决多分类问题（常用）
    - ◆对问题进行拆分，为拆出的每个二分类任务训练一个分类器
    - ◆对每个分类器的预测结果进行集成以获得最终的多分类结果
- 拆分策略
  - ✓ 一对一（One vs. One, OvO）
  - ✓ 一对其余（One vs. Rest, OvR）
  - ✓ 多对多（Many vs. Many, MvM）

四、多分类学习

- 一对一拆分：

  拆分阶段

  - ✓ N个类别两两配对
    - ◆N(N-1)/2 个二类任务
  - ✓ 各个二类任务学习分类器
    - ◆N(N-1)/2 个二类分类器

  测试阶段

  - ✓ 新样本提交给所有分类器预测
    - ◆N(N-1)/2 个分类结果
  - ✓ 投票产生最终分类结果
    - ◆被预测最多的类别为最终类别

四、多分类学习

- 一对其余拆分：

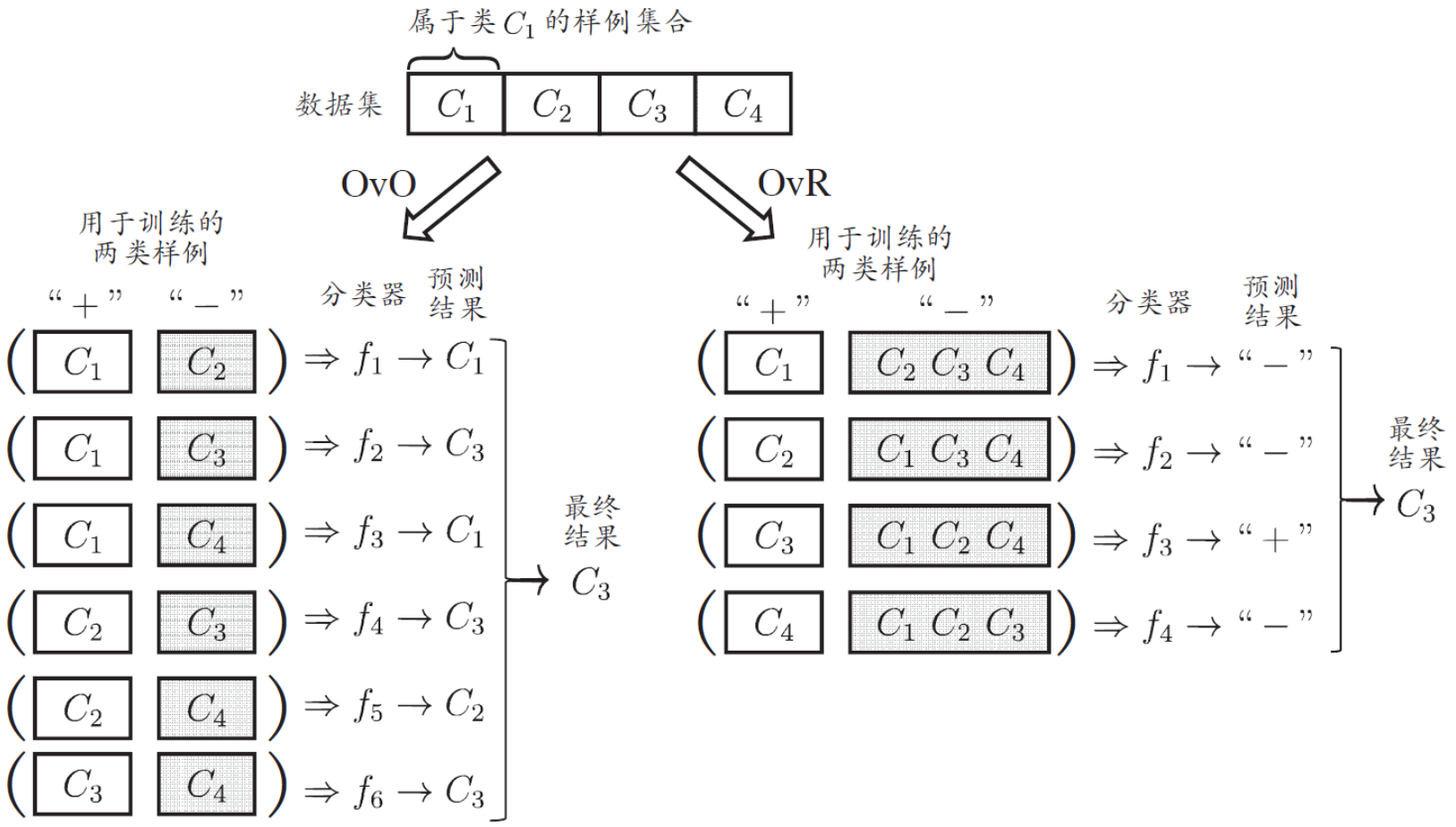
  拆分阶段

  - ✓ 某一类作为正例，其余类作为反例
    - ◆N 个二类任务
  - ✓ 各个二类任务学习分类器
    - ◆N 个二类分类器

  测试阶段

  - ✓ 新样本提交给所有分类器预测
    - ◆N 个分类结果
  - ✓ 比较各分类器的预测置信度
    - ◆仅有一个分类器预测为正类，则对应的类别标记作为最终分类结果；若有多个分类器预测为正类，选择置信度最大类别作为最终类别

四、多分类学习


属于类C_1的样例集合
数据集
C_1 C_2 C_3 C_4
OvO
OvR
用于训练的两类样例
“+” “−”
分类器
预测结果
$(C_1\ C_2) \Rightarrow f_1 \rightarrow C_1$
$(C_1\ C_3) \Rightarrow f_2 \rightarrow C_3$
$(C_1\ C_4) \Rightarrow f_3 \rightarrow C_1$
$(C_2\ C_3) \Rightarrow f_4 \rightarrow C_3$
$(C_2\ C_4) \Rightarrow f_5 \rightarrow C_2$
$(C_3\ C_4) \Rightarrow f_6 \rightarrow C_3$
最终结果
C_3
用于训练的两类样例
“+” “−”
分类器
预测结果
$(C_1\ C_2\ C_3\ C_4) \Rightarrow f_1 \rightarrow$ “−”
$(C_2\ C_1\ C_3\ C_4) \Rightarrow f_2 \rightarrow$ “−”
$(C_3\ C_1\ C_2\ C_4) \Rightarrow f_3 \rightarrow$ “+”
$(C_4\ C_1\ C_2\ C_3) \Rightarrow f_4 \rightarrow$ “−”
最终结果
C_3

四、多分类学习

一对一

- 训练N(N-1)/2个分类器，存储开销和测试时间大
- 训练只用两个类的样例，训练时间短

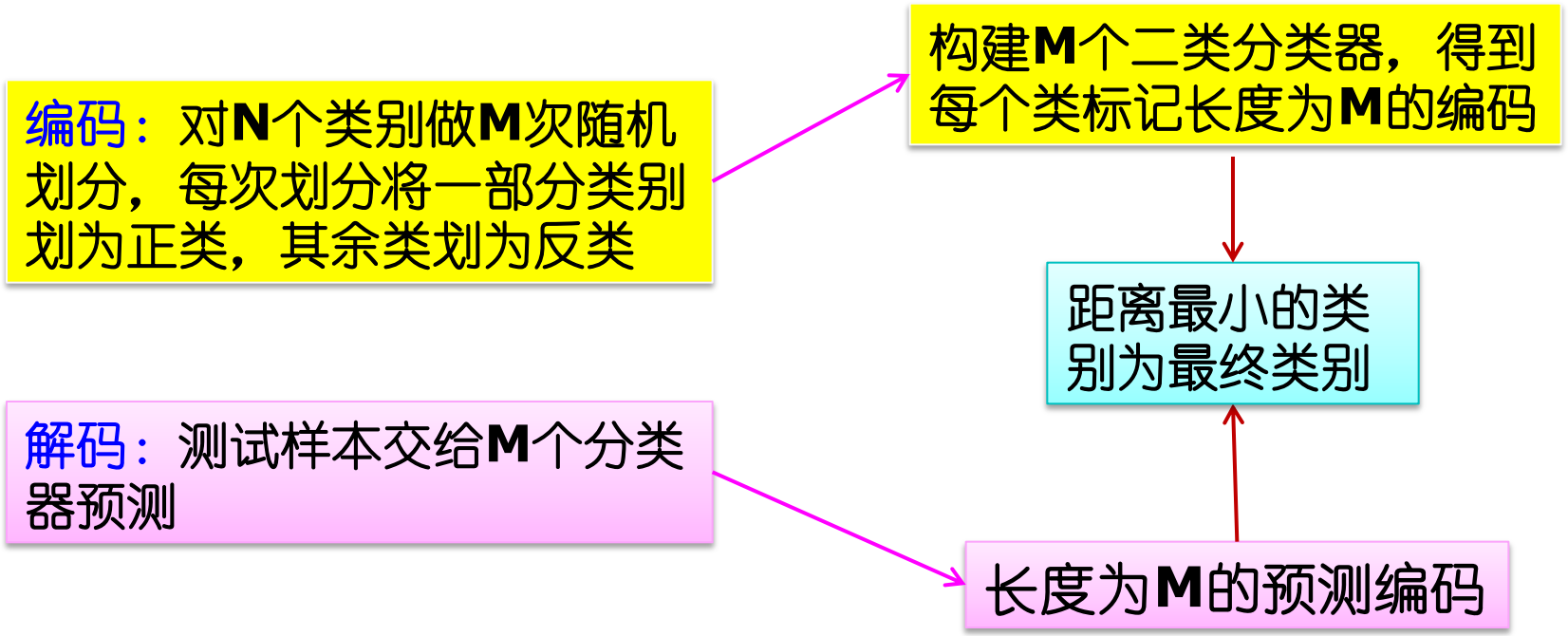
一对其余

- 训练N个分类器，存储开销和测试时间小
- 训练用到全部训练样例，训练时间长

预测性能取决于具体数据分布，多数情况下两者差不多

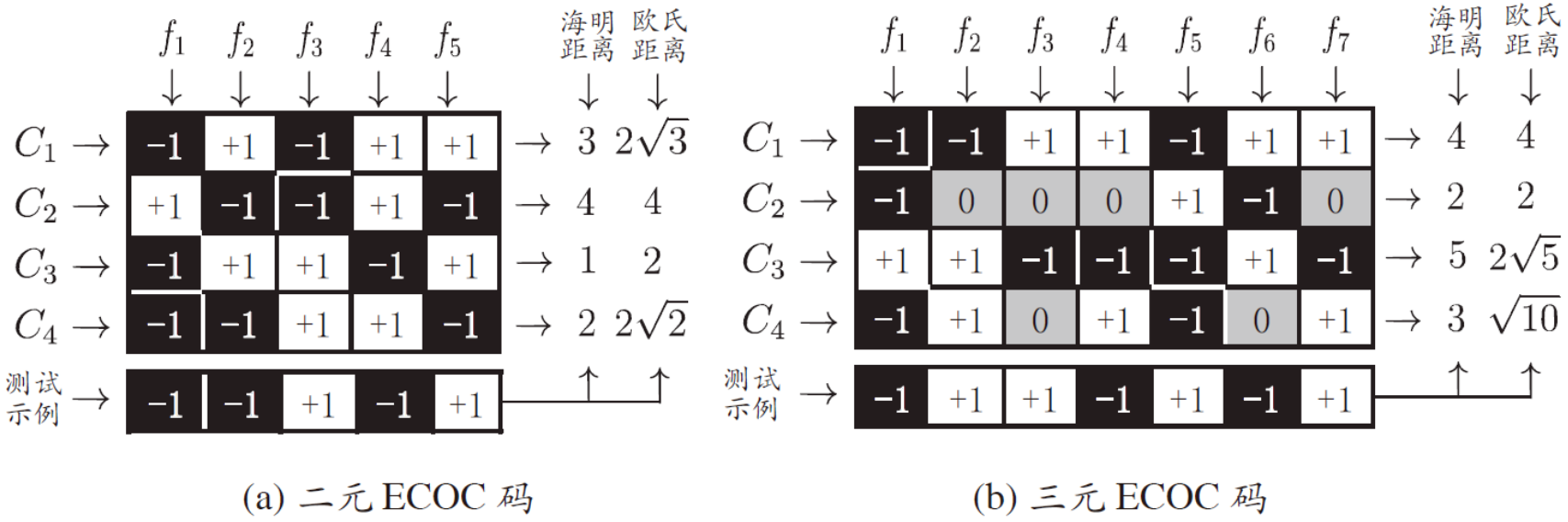
四、多分类学习

- 多对多（Many vs Many, MvM）
  - ✓ 若干类作为正类，若干个其他类作为反类
  - ✓ 纠错输出码（**Error Correcting Output Code, ECOC**）



四、多分类学习

- 纠错输出码：二元码和三元码



- 对分类器错误有一定容忍和修正能力，编码越长、纠错能力越强
- 对同等长度的编码，理论上来说，任意两个类别之间的编码距离越远，则纠错能力越强