线性模型

------BY 2020.2 YANGZHONGXIU

内容

- 基本形式
- 线性回归
- 对数几率回归
- 线性判别分析
- 多分类学习
- 类别不平衡问题

基本形式

给定由d 个属性描述的示例x = (x1; x2;...; Xd) ， 其中均是xi在第i个属性上的取值，线性模型(linear model)试图学得一个通过属性的线性组合来进行预测的函数，即

$$f(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d + b \ ,$$

一般用向量形式写成:

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

其中ω=(ω1;ω2;...;ωd)， ω 和b 学得之后， 模型就得以确定。

基本形式

$$f(\boldsymbol{x}) = w_1 x_1 + w_2 x_2 + \ldots + w_d x_d + b \,,$$

系数代表什么意思？

ω 直观表达了各属性在预测中的重要性，因此线性模型有很好的可解释性(comprehensibility)。

例如若在西瓜问题中学得 $f_{好瓜}(x)=0.2\cdot x_{色泽}+0.5\cdot x_{根蒂}+0.3\cdot x_{敲声}+1$ ，

则意味着可通过综合考虑色泽、根蒂和敲声来判断瓜好不好，其中根蒂最要紧，而敲声比色泽更重要.

线性回归

给定数据集$D = \{(x_1, y_1), (x_2, y_2), \ldots, (x_m, y_m)\}$， 其中

$x_i = (x_{i1}; x_{i2}; \ldots; x_{id})$, **$y_i \in R$**。

"**线性回归**" (linear regression)试图学得一个线性模型以尽可能准确地预测实值输出标记。

一个问题： 离散值如何连续化？

线性回归

离散值的连续化

对离散属性，**若属性值间存在“序” (order)关系**，可通过连续化将其转化为连续值，例如二值属性“身高”的取值“高” “矮”可转化为{1，0.5}，三值属性“高度”的取值“高” “中” “低”可转化为{1，0.5，0.0}。

若属性值间不存在序关系，假定有k 个属性值，则通常转化为k 维向量，例如属性“瓜类”的取值“西瓜” “南瓜” “黄瓜”可转化为(0，0，1)，(0, 1，0)，(1 ,0, 0)。

线性回归

线性回归试图学得：

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

如何确定ω 和b 呢?显然?关键在于如何衡量f(x) 与y 之间的差别。2.3 节介绍过，**均方误差是回归任务中最常用的性能度量**，因此我们可试图让均方误差最小化。

最小二乘法

$$(w^*, b^*) = \arg\min_{(w,b)} \sum_{i=1}^{m} (f(x_i) - y_i)^2$$

$$= \arg\min_{(w,b)} \sum_{i=1}^{m} (y_i - wx_i - b)^2 \ .$$

ω *，　b* 表示ω和b 的解.

均方误差的几何意义：对应常用的欧几里得距离或简称"欧氏距离" (Euclidean distance)。

基于均方误差最小化来进行模型求解的方法称为“最小二乘法” (least square method)。

在线性回归中，**最小二乘法就是试图找到一条直线，使所有样本到直线上的欧氏距离之和最小。**

线性回归

求解ω 和b 使

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

最小化的过程，称为线性回归模型的最小二乘"参数估计"(parameter estimation).

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

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

线性回归

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

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

为零可得到ω和b 最优解的闭式(closed-form) 解

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

$$b = \frac{1}{m}\sum_{i=1}^{m}(y_i - wx_i),$$

其中 $\bar{x} = \frac{1}{m}\sum_{i=1}^{m} x_i$ 为 x 的均值.

线性回归—多元线性回归

更一般的情形是如本节开头的数据集D， 样本由d 个属性描述。此时我们试图学得：

$$f(\boldsymbol{x}_i) = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}_i + b, \text{ 使得 } f(\boldsymbol{x}_i) \simeq y_i \ ,$$

这称为“多元线性回归” (multivariate linear regression)

线性回归—多元线性回归

同样可利用最小二乘法来对ω 和b 进行估计.为便于讨论，我们把ω和b 吸收入向量形式 $\hat{\boldsymbol{w}} = (\omega ; b)$ ， 相应的，把数据集D 表示为一个m x (d +1)大小的矩阵X ，其中每行对应于一个示例，该行前d 个元素对应于示例的d 个属性值，最后一个元素恒置为1 ，即

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1^{\mathrm{T}} & 1 \\ \boldsymbol{x}_2^{\mathrm{T}} & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_m^{\mathrm{T}} & 1 \end{pmatrix}$$

线性回归—多元线性回归

再把标记也写成向量形式y=(y1; y2;... ;ym)，则有：

$$\hat{\boldsymbol{w}}^* = \arg\min_{\hat{\boldsymbol{w}}} (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})^{\mathrm{T}} (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})$$

令 $E_{\hat{\boldsymbol{w}}} = (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})^{\mathrm{T}} (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})$, 对 $\hat{\boldsymbol{w}}$ 求导得到

$$\frac{\partial E_{\hat{\boldsymbol{w}}}}{\partial \hat{\boldsymbol{w}}} = 2\,\mathbf{X}^{\mathrm{T}} (\mathbf{X}\hat{\boldsymbol{w}} - \boldsymbol{y}) \ .$$

线性回归—多元线性回归

令 $E_{\hat{\boldsymbol{w}}} = (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})^{\mathrm{T}} (\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}})$, 对 $\hat{\boldsymbol{w}}$ 求导得到

$$\frac{\partial E_{\hat{\boldsymbol{w}}}}{\partial \hat{\boldsymbol{w}}} = 2\,\mathbf{X}^{\mathrm{T}} (\mathbf{X}\hat{\boldsymbol{w}} - \boldsymbol{y}) \ .$$

令上式为零可得最优解的闭式解，但由于涉及矩阵逆的计算，比单变量情形要复杂一些.下面我们做一个简单的讨论.

当X^TX 为满秩矩阵(full-rank matrix)或正定矩阵(positive definite matrix)时，令式上式为零可得

$$\hat{\boldsymbol{w}}^* = (\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y} \ ,$$

线性回归模型为

$$f(\hat{\boldsymbol{x}}_i) = \hat{\boldsymbol{x}}_i^{\mathrm{T}}(\mathbf{X}^{\mathrm{T}}\mathbf{X})^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y} \ .$$

然而，现实任务中x^Tx 往往不是满秩矩阵.例如在许多任务中会遇到大量的变量，其数目甚至超过样例数，导致X 的列数多于行数，x^Tx 显然不满秩.此时可解出多个解，它们都能使均方误差最小化.选择哪一个解作为输出，将由学习算法的归纳偏好决定，常见的做法是引入正则化(regularization) 项.

线性模型虽简单，却有丰富的变化. 例如对于样例（x, y）， $y \in \mathbb{R}$ 当我们希望线性模型的预测值逼近真实标记y时， 就得到了线性回归模型。为便于观察，我们把线性回归模型简写为

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

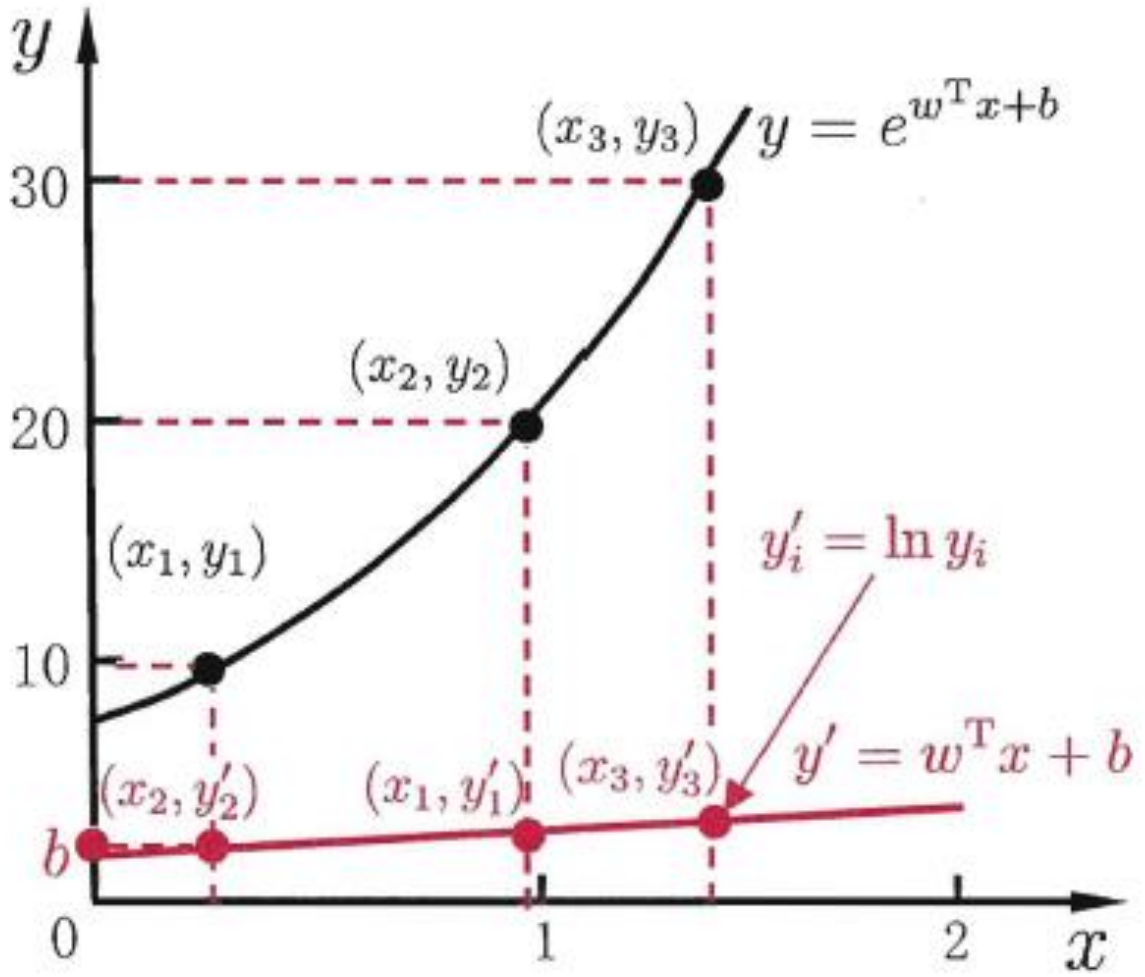
可否令模型预测值逼近y的衍生物呢? 譬如说，假设我们认为示例所对应的输出标记是在指数尺度上变化，那就可**将输出标记的对数作为线性模型逼近的目标**， 即

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

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

这就是“对数线性回归” (log-linear regression)，它实际上是在试图让 $e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}$ 逼近y .

该式在形式上仍是线性回归，但实质上已是在求取输入空间到输出空间的非线性函数映射，如图所示. 这里的对数函数起到了将线性回归模型的预测值与真实标记联系起来的作用



对数线性回归示意图

更一般地，考虑单调可微函数g(.)， 令

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

这样得到的模型称为**“广义线性模型” (generalized linear model)**，其中函数g(·) 称为“联系函数” (link function). 显然， 对数线性回归是广义线性模型在*g(·)* = ln(·) 时的特例。

对数几率回归

对数几率回归是对于分类作预测。

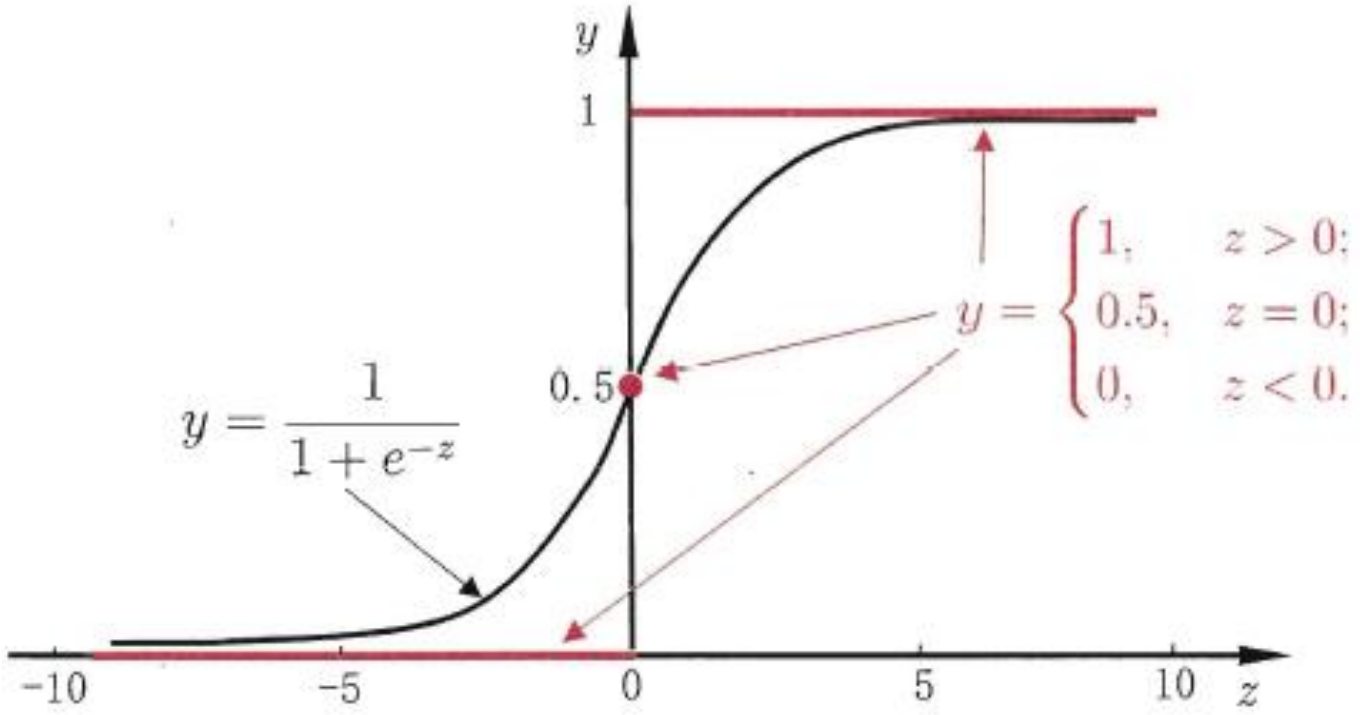
$$y = g^{-1}(\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b)$$

对于该公式，只需要找一个单微调可微函数将分类任务的真实标记y与线性回归模型的预测值联系起来。

考虑二分类任务， 其输出标记 y ∈ {0， 1} ，而线性回归模型产生的预测值z = ωTx +b 是实值，需将实值z 转换为0/ 1 值. 最理想的是"单位阶跃函数" (unit-step function)

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

即若预测值z ：大于零就判为正例；小于零则判为反例；预测值为临界值零则可任意判别



单位阶跃函数与对数几率函数

单位阶跃函数不连续，因此不能直接用作式

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

的g-(•)．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”(surrogate function)，并希望它单调可微.对数几率函数(logistic function)正是这样一个常用的替代函数:

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

对数几率函数是一种"Sigmoid 函数"，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.将对数几率函数作为g一(·)代入式，

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

得到

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

可变化为

$$\ln \frac{y}{1-y} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b .$$

$$\ln \frac{y}{1-y} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \ .$$

若将y视为样本x作为正例的可能性，则1-y 是其反例可能性，两者的比值y/(1-y) 则称为“几率” (odds)，反映了x 作为正例的相对可能性。对几率取对数则得到"对数几率" (log odds，亦称logit)

$$\ln \frac{y}{1-y}$$

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

由此可看出，上式实际上是在**用线性回归模型的预测结果去逼近真实标记的对数几率**，因此，其对应的模型称为“对数几率回归”(logistic regression，亦称logit regression)。**特别需注意到，虽然它的名字是"回归"，实际却是一种分类学习方法.**

该方法优点：（1）它是直接对分类可能性进行建模，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；
（2）它不是仅预测出"类别"，而是**可得到近似概率预测**，这对许多需利用概率辅助决策的任务很有用；
（3）对**率函数是任意阶可导的凸函数**，有很好的数学性质，现有的许多数值优化算法都**可直接用于求取最优解**。

如何确定ω和b。

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

b $$\ln \frac{y}{1-y} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b .$$

我们来看看如何确定式a式中的ω 和b. 若将式a式中的y视为类后验概率估计p(y = 1 | x)， 则式b式可重写为

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

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

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

可通过“极大似然法”（maximum likelihood method）来估计ω 和b 给定数据集$\{(\boldsymbol{x}_i, y_i)\}_{i=1}^m$，对率回归模型最大化“对数似然”(loglikelihood)

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

即令每个样本属于其真实标记的概率越大越好. 为便于讨论, 令 $\boldsymbol{\beta} = (\boldsymbol{w}; b)$, $\hat{\boldsymbol{x}} = (\boldsymbol{x}; 1)$, 则 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$ 可简写为 $\boldsymbol{\beta}^{\mathrm{T}}\hat{\boldsymbol{x}}$. 再令 $p_1(\hat{\boldsymbol{x}}; \boldsymbol{\beta}) = p(y = 1 \mid \hat{\boldsymbol{x}}; \boldsymbol{\beta})$, $p_0(\hat{\boldsymbol{x}}; \boldsymbol{\beta}) = p(y = 0 \mid \hat{\boldsymbol{x}}; \boldsymbol{\beta}) = 1 - p_1(\hat{\boldsymbol{x}}; \boldsymbol{\beta})$, 则 上式 中的似然项可重写为

$$p(y_i \mid \boldsymbol{x}_i; \boldsymbol{w}, b) = y_i p_1(\hat{\boldsymbol{x}}_i; \boldsymbol{\beta}) + (1 - y_i) p_0(\hat{\boldsymbol{x}}_i; \boldsymbol{\beta}) \ .$$

$$\ell(\boldsymbol{\beta}) = \sum_{i=1}^{m} \left(-y_i \boldsymbol{\beta}^{\mathrm{T}} \hat{\boldsymbol{x}}_i + \ln \left(1 + e^{\boldsymbol{\beta}^{\mathrm{T}} \hat{\boldsymbol{x}}_i} \right) \right) .$$

是关于β 的高阶可导连续凸函数，根据凸优化理论[Boyd and Vandenberghe, 2004]，经典的数值优化算法如梯度下降法(gradient descent method) 、牛顿法(Newton method)等都可求得其最优解，于是就得到

$$\boldsymbol{\beta}^* = \arg\min_{\boldsymbol{\beta}} \ell(\boldsymbol{\beta}) .$$

以牛顿法为例, 其第 $t+1$ 轮迭代解的更新公式为

$$\boldsymbol{\beta}^{t+1} = \boldsymbol{\beta}^t - \left(\frac{\partial^2 \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \, \partial \boldsymbol{\beta}^{\mathrm{T}}} \right)^{-1} \frac{\partial \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}}$$

其中关于 $\boldsymbol{\beta}$ 的一阶、二阶导数分别为

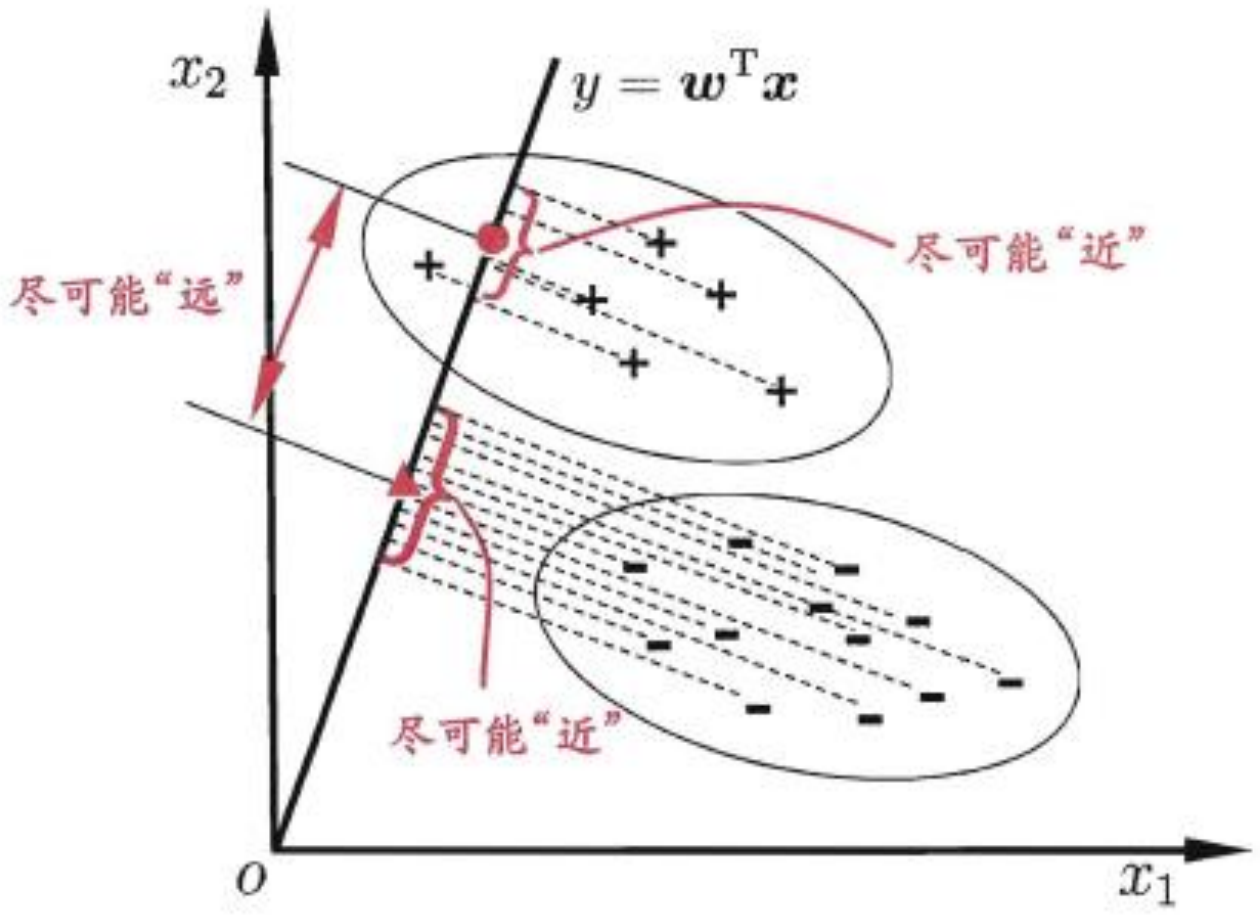
$$\frac{\partial \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta}} = -\sum_{i=1}^{m} \hat{\boldsymbol{x}}_i (y_i - p_1(\hat{\boldsymbol{x}}_i; \boldsymbol{\beta})) \ ,$$

$$\frac{\partial^2 \ell(\boldsymbol{\beta})}{\partial \boldsymbol{\beta} \partial \boldsymbol{\beta}^{\mathrm{T}}} = \sum_{i=1}^{m} \hat{\boldsymbol{x}}_i \hat{\boldsymbol{x}}_i^{\mathrm{T}} p_1(\hat{\boldsymbol{x}}_i; \boldsymbol{\beta})(1 - p_1(\hat{\boldsymbol{x}}_i; \boldsymbol{\beta})) \ .$$

线性判别分析(LDA)

线性判别分析(Linear Discriminant Analysis，简称LDA) 是一种经典的线性学习方法，在二分类问题上因为最早由[Fisher，1936] 提出，亦称“Fisher 判别分析“。

LDA 的思想: 给定训练样例集，设法将样例投影到一条直线上，使得同类样例的投影点尽可能接近、异类样例的投影点尽可能远离；在对新样本进行分类时，将其投影到同样的这条直线上，再根据投影点的位置来确定新样本的类别。 下图给出了一个二维示意图.



LDA 的二维示意图

″+″、" -″分别代表正例和反例，椭圆表示数据簇的外轮廓，虚线表示投影，红色实心园和实心三角形分别表示两类样本投影后的中心点.

多分类学习

现实中常遇到多分类学习任务.有些二分类学习方法可直接推广到多分类，但在更多情形下，我们是基于一些基本策略，利用二分类学习器来解决多分类问题。

不失一般性，考虑N 个类别C1， C2， …， CN， 多分类学习的基本思路是“拆解法”，**即将多分类任务拆为若干个二分类任务求解**。具体来说，先对问题进行拆分，然后为拆出的每个二分类任务训练一个分类器;在测试时，对这些分类器的预测结果进行集成以获得最终的多分类结果.这里的关键是如何对多分类任务进行拆分，以及如何对多个分类器进行集成。

本节主要介绍拆分策略。

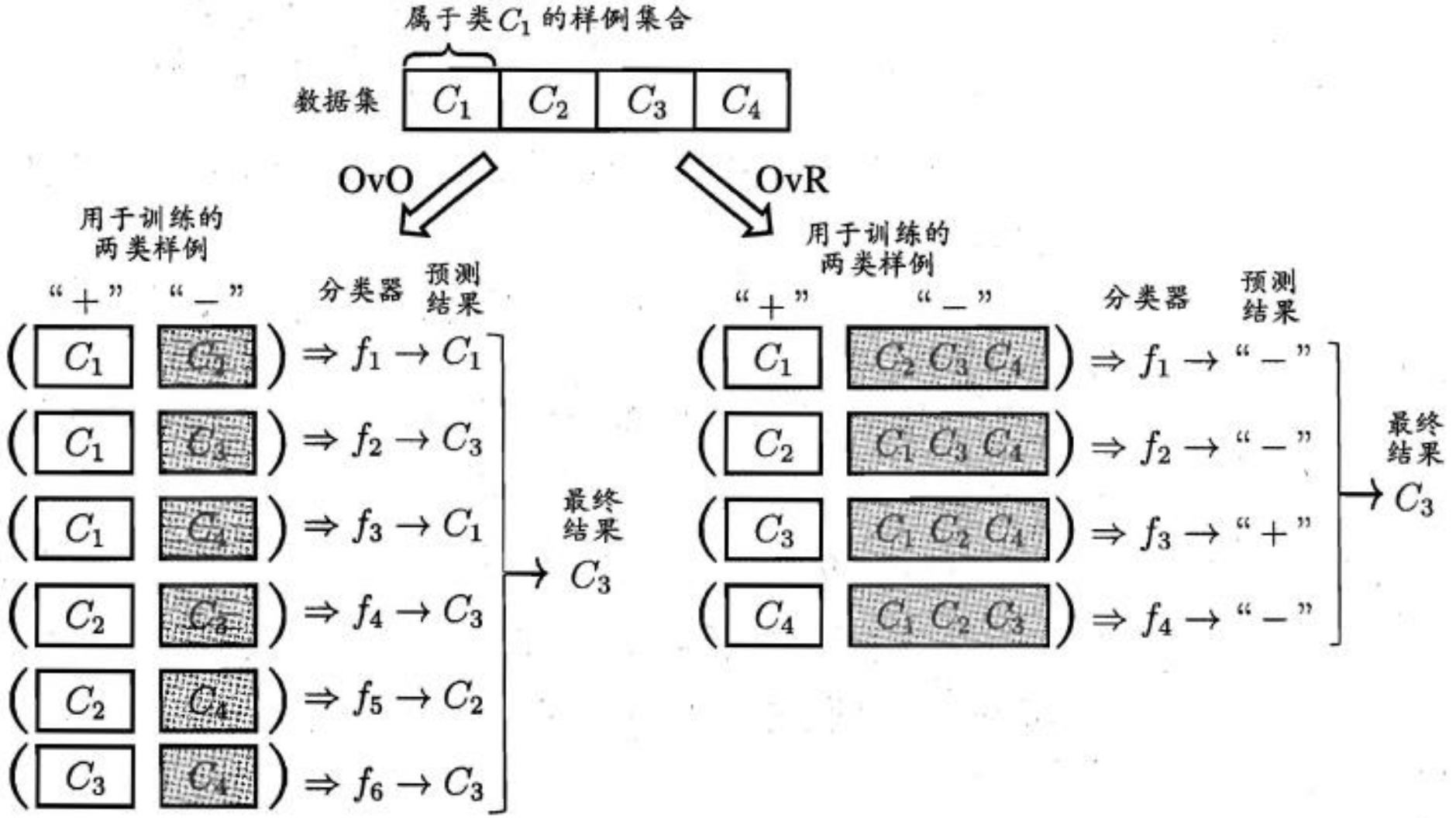
多分类学习

最经典的拆分策略有三种：“一对一” (One vs. One， 简称OvO) 、“一对其余” (One vs. Rest ， 简称OvR)和“多对多” (Many vs. Ma町，简称MvM)。

给定数据集D = *{(xl ,y1) ,* (x2 ,y2)， . . . ， (xm,ym)} ， yi ϵ{C1 ，C2 ， ...， *CN}*。OvO 将这N 个类别两两配对，从而产生N(N -1)/2 个三分类任务，例如OvO将为区分类别Ci 和Cj 训练个分类器，该分类器把D 中的Ci类样例作为正例， Cj 类样例作为反例。在测试阶段，新样本将同时提交给所有分类器，于是将得到N(N -1)/2 个分类结果，最终结果可通过投票产生：即把被预测得最多的类别作为最终分类结果。

多分类学习

OvR 则是每次将一个类的样例作为正例、所有其他类的样例作为反例来训练N 个分类器。在测试时**若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**，如图所示。**若有多个分类器预测为正类，则通常考虑各分类器的预测置信度，选择置信度最大的类别标记作为分类结果。**


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

多分类学习

容易看出， OvR 只需训练N 个分类器， 而OvO 需训练N(N -1)/2 个分类器， 因此， OvO的存储开销和测试时间开销通常比OvR 更大。但在训练时，OvR 的每个分类器均使用全部训练样例，而OvO 的每个分类器仅用到两个类的样例。因此，在类别很多时， OvO 的训练时间开销通常比OvR 更小。 至于预测性能， 则取决于具体的数据分布， 在多数情形下两者差不多。

多分类学习

MvM 是每次将若干个类作为正类，若干个其他类作为反类。显然， OvO 和OvR 是MvM 的特例。MvM 的正、反类构造必须有特殊的设计，不能随意选取。这里我们介绍一种最常用的MvM 技术“**纠错输出码**” (Error Correcting Output Codes，简称ECOC)。

ECOC [Dietterich and Bakiri, 1995] 是将编码的思想引入类别拆分，并尽可能在解码过程中具有容错性。

多分类学习

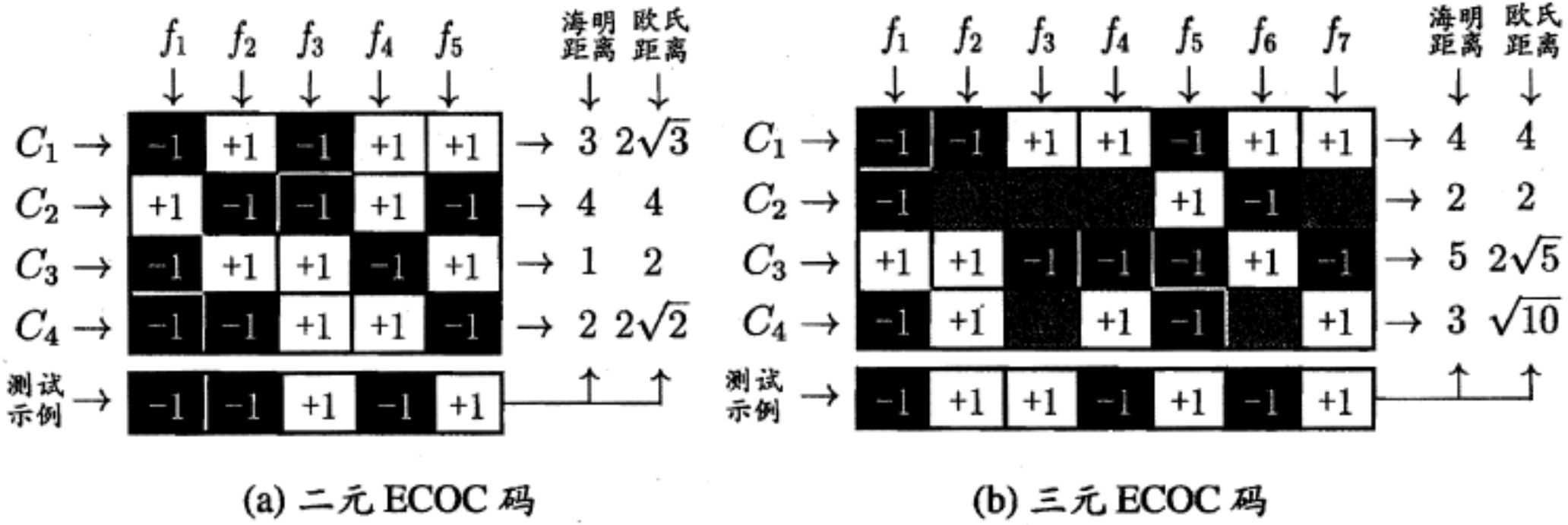
ECOC 工作过程主要分为两步:

- **编码**:对N个类别做M 次划分， 每次划分将一部分类别划为正类，一部分划为反类，从而形成一个二分类训练集;这样一共产生M个训练集，可训练出M 个分类器。

- **解码**:M个分类器分别对测试样本进行预测，这些预测标记组成一个编码.将这个预测编码与每个类别各自的编码进行比较，返回其中距离最小的类别作为最终预测结果。

多分类学习

类别划分通过“**编码矩阵**” (coding matrix)指定。编码矩阵有多种形式，常见的主要有二元码[Dietterich and Barkiri， 1995] 和三元码[Allwein et al.,2000]。前者将每个类别分别指定为正类和反类，后者在正、反类之外，还可指定“停用类”。给出了一个示意图，



在图 (a) 中，分类器f2 将C1类和C3 类的样例作为正例， C2 类和C4 类的样例作为反例；在图 (b) 中，分类器f4 将C1 类和C4 类的样例作为正例， C3 类的样例作为反例。在解码阶段，各分类器的预测结果联合起来形成了测试示例的编码，该编码与各类所对应的编码进行比较，将距离最小的编码所对应的类别作为预测结果。例如在图 (a) 中，若基于欧氏距离，预测结果将是C3。

判别不平衡问题

问题引入：

前面介绍的分类学习方法都有一个共同的基本假设，即不同类别的训练样例数目相当。如果**不同类别的训练样例数目稍有差别，通常影响不大，但若差别很大，则会对学习过程造成困扰**。例如有998 个反例，但正例只有2 个，那么学习方法只需返回一个永远将新样本预测为反例的学习器，就能达到99.8% 的精度；然而这样的学习器往往没有价值，因为它不能预测出任何正例。

判别不平衡问题

类别不平衡(class imbalance)就是指分类任务中不同类别的训练样例数目差别很大的情况。不失一般性，本节假定正类样例较少，反类样例较多.在现实的分类学马任务中，我们经常会遇到类别不平衡。例如在通过拆分法解决多分类问题时，即使原始问题中不同类别的训练样例数目相当，在使用OvR 、MvM策略后产生的二分类任务仍可能出现类别不平衡现象，因此有必要了解类别不平衡性处理的基本方法。

判别不平衡问题

从线性分类器的角度讨论容易理解，在我们用$y=w^Tx+b$ 对新样本m进行分类时，事实上是在用预测出的y 值与一个阈值进行比较，例如通常在$y > 0.5$ 时判别为正例，否则为反例x 实际上表达了正例的国能性，几率y/(1-y)则反映了正例可能性与反例可能性之比值，阈值设置为0.5 恰表明分类器认为真实正、反例可能性相同，即分类器决策规则为

若 $\frac{y}{1-y} > 1$ 则 预测为正例.

式1

判别不平衡问题

然而，当训练集中正、反例的数目不同时，令m^+表示正倒数目，m^-表示反倒数目，则观测几率是m^+ / m^-，由于我们通常假设训练集是真实样本总体的无偏采样，因此观测几率就代表了真实几率。于是，只要分类器的预测几率高于观测几率就应判定为正例，即：

$$\text{若 } \frac{y}{1-y} > \frac{m^+}{m^-} \text{ 则 预测为正例.}$$

分类器是基于式1进行决策，因此，需对其预测值进行调整，使其在基于式1决策时，实际是在执行上式，要做到这点很容易，只需令

$$\frac{y'}{1-y'} = \frac{y}{1-y} \times \frac{m^-}{m^+} .$$

这就是类别不平衡学习的一个基本策略“**再缩放**” (rescaling)。再缩放的思想虽简单，但实际操作却并不平凡，主要因为“**训练集是真实样本总体的无偏采样**”**这个假设往往并不成立**，也就是说， 我们未必能有效基于训练集观测几率来推断出真实几率。

现有技术大体上有三类做法:

- 第一类是**直接对训练集里的反类样例进行“欠采样”** (undersampling)，即去除一些反例使得正、反例数接近， 然后再进行学习;
- 第二类是**对训练集里的正类样例进行“过采样” (oversampling)**，即增加一些正例使得正、反例数目接近，然后再进行学习;
- 第三类则是直接基于原始训练集进行学习，但在用训练好的分类器进行预测时，将式 $\frac{y'}{1-y'} = \frac{y}{1-y} \times \frac{m^-}{m^+}$ 嵌入到其决策过程中，称为**“阈值移动" (threshold-moving)**.

欠采样法的时间开销通常远小于过采样法，因为前者丢弃了很多反例，使得分类器训练集远小子初始训练集，而过采样法增加了很多正例，其训练、集大于初始训练集。**需注意的是，过采样法不能简单地对初始正例样本进行重复采样，否则会招致严重的过拟合**；过采样法的代表性算法SMOTE［Chawlaet al., 2002］是通过对训练集里的正例进行插值来产生额外的正例。另一方面，**欠采样法若随机丢弃反例，可能丢失一些重要信息；**欠采样法的代表性算法EasyEnsemble [Liu et al，2009］则是利用集成学习机制，将反倒划分为若干个集合供不同学习器使用，这样对每个学习器来看都进行了欠采样，但在全局来看却不会丢失重要信息。

值得一提的是，“再缩放”也是“代价敏感学习” (cost-sensitive learning)的基础。在代价敏感学习中将式中的m-/m+ 用cost+ /cost- 代替即可，其中cost+ 是将正例误分为反例的代价， cost- 是将反例误分为正例的代价.

本章总结

- 线性模型的基本形式
- 线性回归的数学模型推导
- 对数几率回归方法思想
- 线性判别分析方法思想
- 多分类学习拆分策略
- 类别不平衡的处理方式