第2章
线性模型
Linear Models

向 世 明
smxiang@nlpr.ia.ac.cn
http://www.escience.cn/people/smxiang/index.html

时空数据分析与学习课题组(STDAL)
中科院自动化研究所 模式识别国家重点实验室

助教： 方深 (shen.fang@nlpr.ia.ac.cn)

内容提要

- 基本形式
- 线性回归
- 广义线性回归
- 对数几率回归
- Softmax回归
- 线性判别分析
- 局部线性判别分析
- 特征选择
- 多分类问题

2.1 基本形式

- **线性模型**
  - 给定由 d 个属性(特征)描述的示例(样本) $\mathbf{x} = [x_1, x_2, \ldots, x_d]^T \in R^d$，线性模型学习如下关于特征的组合函数：

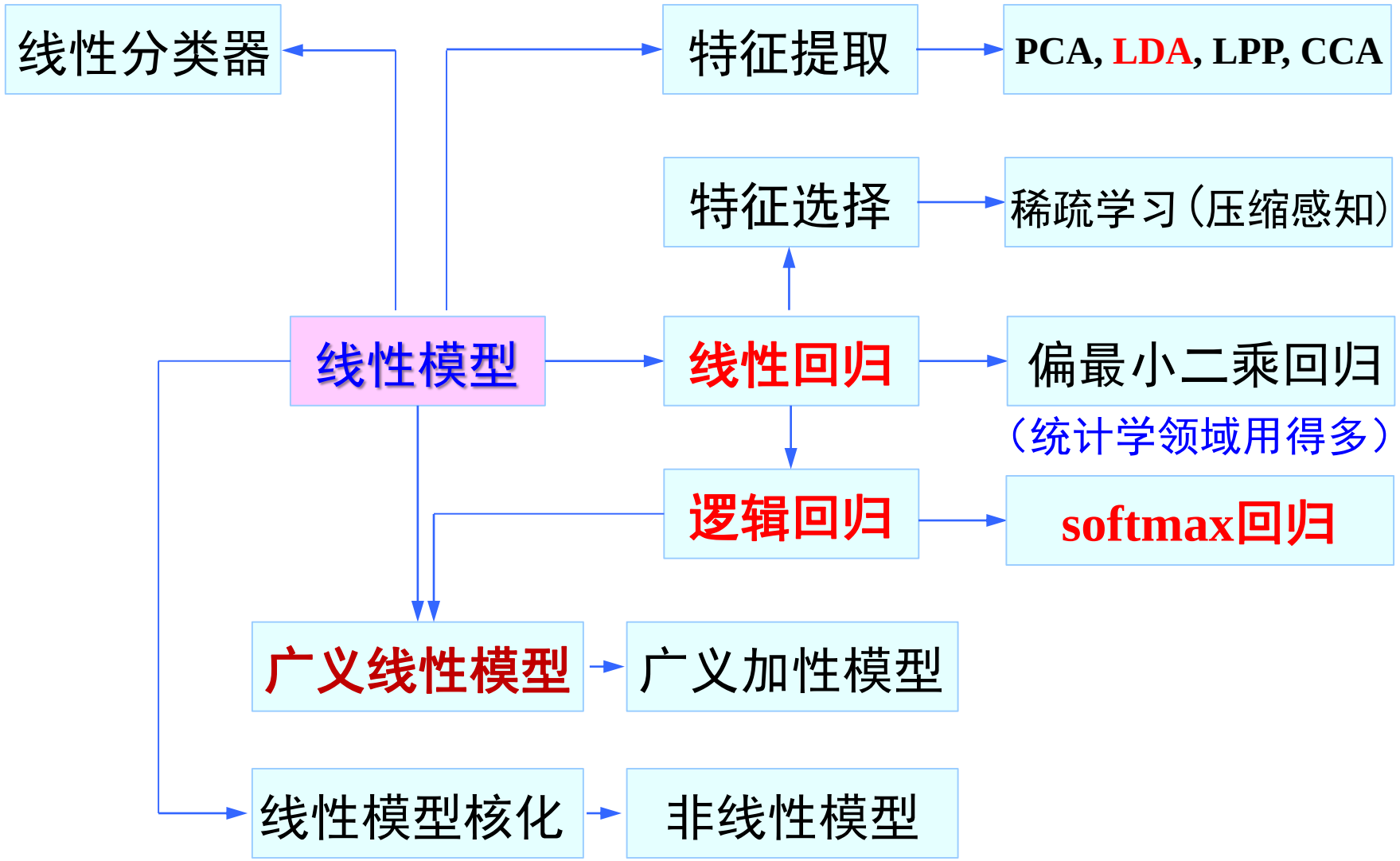
$$f(\mathbf{x}) = w_1 x_1 + w_2 x_2 + \cdots + w_d x_d + b = \mathbf{w}^T \mathbf{x} + b$$

  - 在线性模型中，$\mathbf{w}$直观地表达了各属性在预测中的重要性，因此线性模型具有很好的可解释性。

$$f_{\text{好瓜}}(\mathbf{x}) = 0.2x_{\text{色泽}} + 0.5x_{\text{根蒂}} + 0.3x_{\text{敲声}} + 1$$

  - 线性模型简单，易于建模，但却蕴含着机器学习中的一些重要思想。许多非线性模型可在线性模型的基础上通过引入高维映射或者层级结构来得到。

- **线性模型扩展**


线性分类器
特征提取
PCA, LDA, LPP, CCA
特征选择
稀疏学习（压缩感知）
线性模型
线性回归
偏最小二乘回归
（统计学领域用得多）
逻辑回归
softmax回归
广义线性模型
广义加性模型
线性模型核化
非线性模型

2.2 线性回归

- **线性回归**
  - 样本集$D=\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2), \ldots, (\mathbf{x}_n, y_n)\}$，线性回归试图学习一个线性模型以尽可能地预测实值输出标记。
  - **采用均方误差最小化**，学习问题：

$$(\mathbf{w}^*, b^*) = \arg\min_{\mathbf{w}\in R^d, b\in R} \sum_{i=1}^{n} \left(\mathbf{w}^T\mathbf{x}_i + b - y_i\right)^2$$

均方误差有很好的几何意义，它对应着欧氏距离。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。

线性回归的几何意义

- **求解目标函数：**

$$E(\mathbf{w},b)=\sum_{i=1}^{n}\left(\mathbf{w}^T\mathbf{x}_i+b-y_i\right)^2=\sum_{i=1}^{n}\left|\mathbf{w}^T\mathbf{x}_i+b-y_i\right|^2$$

$$\mathbf{X}=\begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{n1} & x_{n2} & \cdots & x_{nd} & 1 \end{pmatrix}\in R^{n\times(d+1)},\quad \hat{\mathbf{w}}=\begin{pmatrix} w_1 \\ \vdots \\ w_d \\ b \end{pmatrix}\in R^{d+1},\quad \mathbf{y}=\begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}\in R^{n}$$

（齐次坐标）　　（参数）　　（回归目标）

$$E(\mathbf{w},b)=E(\hat{\mathbf{w}})=\left(\mathbf{X}\hat{\mathbf{w}}-\mathbf{y}\right)^T\left(\mathbf{X}\hat{\mathbf{w}}-\mathbf{y}\right)=\hat{\mathbf{w}}^T\mathbf{X}^T\mathbf{X}\hat{\mathbf{w}}-2\hat{\mathbf{w}}^T\mathbf{X}^T\mathbf{y}+\mathbf{y}^T\mathbf{y}$$

（代价函数）

$$\hat{\mathbf{w}}^*=\arg\min_{\hat{\mathbf{w}}\in R^{d+1}}E(\hat{\mathbf{w}})$$

2.2 线性回归

- **求解**
  - 求偏导数：

$$\frac{\partial E(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}} = \frac{\partial\left(\left(\mathbf{X}\hat{\mathbf{w}}-\mathbf{y}\right)^T\left(\mathbf{X}\hat{\mathbf{w}}-\mathbf{y}\right)\right)}{\partial \hat{\mathbf{w}}}$$

$$= \frac{\partial\left(\hat{\mathbf{w}}^T\mathbf{X}^T\mathbf{X}\hat{\mathbf{w}}-2\hat{\mathbf{w}}^T\mathbf{X}^T\mathbf{y}+\mathbf{y}^T\mathbf{y}\right)}{\partial \hat{\mathbf{w}}}$$

$$= 2\mathbf{X}^T\mathbf{X}\hat{\mathbf{w}}-2\mathbf{X}^T\mathbf{y}$$

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

$$f(\hat{\mathbf{x}}) = \hat{\mathbf{x}}^T(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}, \quad \hat{\mathbf{x}} = [x_1, x_2, \cdots, x_d, 1]^T \in R^{d+1}$$

2.2 线性回归

- **多回归任务：将向量映射为向量**
  - 样本集$D=\{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2),\ldots,(\mathbf{x}_n,\mathbf{y}_n)\}$，它试图学习一组线性变换：以尽可能地预测实值输出标记向量。
  - 设目标值$\mathbf{y}_i$，$i=1, 2,\ldots, n$, 为 m 维空间向量，则需要学习 m 个线性变换：

$$f_i(\mathbf{x}) = w_{1i}x_1 + w_{2i}x_2 + \cdots + w_{di}x_d + b_i = \mathbf{w}_i^T\mathbf{x} + b_i,$$
$$i = 1, 2, \cdots, m$$

其中，$f_i(\mathbf{x})$ 负责将$\mathbf{x}$映射到其目标向量$\mathbf{y}$的第 i 个分量。

多回归任务：考虑如何回归到输出空间的第 i 个变量

$i=1,2,\cdots,m$:

引入两个变量：

$$\hat{\mathbf{w}}_i=\begin{pmatrix} w_{1i} \\ \vdots \\ w_{di} \\ b_i \end{pmatrix}\in R^{d+1},\quad \mathbf{y}^i=\begin{pmatrix} y_{1i} \\ y_{2i} \\ \vdots \\ y_{ni} \end{pmatrix}\in R^n$$

目标函数：

$$E(\hat{\mathbf{w}}_i)=\left(\mathbf{X}\hat{\mathbf{w}}_i-\mathbf{y}^i\right)^T\left(\mathbf{X}\hat{\mathbf{w}}_i-\mathbf{y}^i\right)$$

模型：

$$\hat{\mathbf{w}}_i^*=\arg\min_{\hat{\mathbf{w}}_i\in R^{d+1}} E(\hat{\mathbf{w}}_i)$$

解：

$$\hat{\mathbf{w}}_i^*=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{y}^i$$

2.2 线性回归

- **多回归任务：**综合为矩阵形式，令

$$\mathbf{Y}=[\mathbf{y}^1,\mathbf{y}^2,\cdots,\mathbf{y}^m]=\begin{pmatrix} y_{11} & y_{12} & \cdots & y_{1m} \\ y_{21} & y_{22} & \cdots & y_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ y_{n1} & y_{n2} & \cdots & y_{nm} \end{pmatrix}\in R^{n\times m},\qquad \hat{\mathbf{W}}=\begin{pmatrix} w_{11} & w_{12} & \cdots & w_{1m} \\ w_{21} & w_{22} & \cdots & w_{2m} \\ \vdots & \vdots & \ddots & \vdots \\ w_{d1} & w_{d2} & \cdots & w_{dm} \\ b_1 & b_2 & \cdots & b_m \end{pmatrix}\in R^{(d+1)\times m}$$

$$\hat{\mathbf{W}}^*=[\hat{\mathbf{w}}_1^*,\hat{\mathbf{w}}_2^*,\cdots,\hat{\mathbf{w}}_1^*]=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T[\mathbf{y}^1,\mathbf{y}^2,\cdots,\mathbf{y}^m]=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

2.2 线性回归

- **多回归任务：**
  - 采用紧凑的向量回归形式，即学习如下变换：

$$\mathbf{y}=[f_1(\mathbf{x}),f_2(\mathbf{x}),\cdots,f_m(\mathbf{x})]^T=\mathbf{W}^T\mathbf{x}+\mathbf{b}=:\hat{\mathbf{W}}^T\hat{\mathbf{x}}$$

$$\hat{\mathbf{x}}=[\mathbf{x}^T,1]^T\in R^{(d+1)},\quad \mathbf{x}\in R^d,\quad \mathbf{y}\in R^m,$$

$$\mathbf{W}=[\mathbf{w}_1,\mathbf{w}_2,\cdots,\mathbf{w}_m]\in R^{d\times m},\quad \mathbf{b}=[b_1,b_2,\cdots,b_m]^T\in R^m,$$

$$\hat{\mathbf{W}}=[\mathbf{W}^T,\mathbf{b}]^T=[\hat{\mathbf{w}}_1,\hat{\mathbf{w}}_2,\cdots,\hat{\mathbf{w}}_m]\in R^{(d+1)\times m}.$$

  - 目标函数：

$$E(\hat{\mathbf{W}})=\sum_{i=1}^{m}E(\hat{\mathbf{w}}_i)=\sum_{i=1}^{m}\left(\mathbf{X}\hat{\mathbf{w}}_i-\mathbf{y}^i\right)^T\left(\mathbf{X}\hat{\mathbf{w}}_i-\mathbf{y}^i\right)=\left\|\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y}\right\|_F^2\quad\left(\because\ \|\mathbf{A}\|_F^2=\sum_{i,j}a_{ij}^2\right)$$

- **多回归任务**

$$\left\|\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y}\right\|_F^2=\mathrm{trace}((\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y})^T(\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y}))$$

$$=\mathrm{trace}(\hat{\mathbf{W}}^T\mathbf{X}^T\mathbf{X}\hat{\mathbf{W}}-2\hat{\mathbf{W}}^T\mathbf{X}^T\mathbf{Y}+\mathbf{Y}^T\mathbf{Y})$$

$$\frac{\partial E(\hat{\mathbf{W}})}{\partial\hat{\mathbf{W}}}=2\mathbf{X}^T\mathbf{X}\hat{\mathbf{W}}-2\mathbf{X}^T\mathbf{Y}=\mathbf{0}$$

$$\hat{\mathbf{W}}=(\mathbf{X}^T\mathbf{X})^{-1}\mathbf{X}^T\mathbf{Y}$$

- **计算问题**
  - $\mathbf{X}^T\mathbf{X}$可能是不可逆的，此时，可以在$\mathbf{X}^T\mathbf{X}$的主对角线元素上加上一个很小的正数λ，此时有：

$$\hat{\mathbf{W}}=(\mathbf{X}^T\mathbf{X}+\lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$$

2.2 线性回归

- 计算问题

$$\hat{\mathbf{W}} = (\mathbf{X}^T\mathbf{X} + \lambda\mathbf{I})^{-1}\mathbf{X}^T\mathbf{Y}$$

  - 可以证明，“上式”即为如下结构化风险最小化模型（正则化模型）的最优解：

$$\min_{\hat{\mathbf{W}}} \left\|\mathbf{X}\hat{\mathbf{W}} - \mathbf{Y}\right\|_F^2 + \lambda\left\|\hat{\mathbf{W}}\right\|_F^2$$

  - 从贝叶斯决策的角度进行解释：

$$\max \quad e^{-\left(\left\|\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y}\right\|_F^2+\lambda\left\|\hat{\mathbf{W}}\right\|_F^2\right)} = e^{-\left\|\mathbf{X}\hat{\mathbf{W}}-\mathbf{Y}\right\|_F^2} \times e^{-\lambda\left\|\hat{\mathbf{W}}\right\|_F^2}$$

似然概率(误差分布)　　先验概率（参数分布）

2.3 广义线性回归

- **对数线性回归**
  - 线性回归虽然简单，却有丰富的变化。
  - 能否令模型预测y的衍生物，**即y的函数**？比如，输出标记是在对数尺度上的变化：

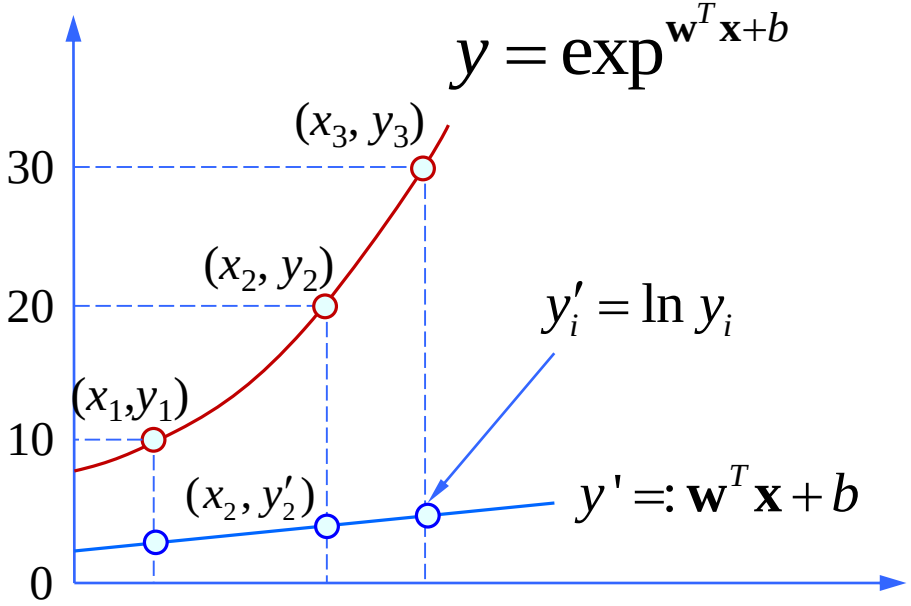
$$\ln y = \mathbf{w}^T \mathbf{x} + b$$

  - 这就是对数线性回归。
  - 其形式上仍然是线性回归，但其实质它是采用如下变换逼近y，因此是非线性的：

$$y = \exp^{\mathbf{w}^T \mathbf{x} + b}$$

2.3 广义线性回归

- 对数线性回归



- 更一般地（广义线性回归），考虑单调可微函数g，则

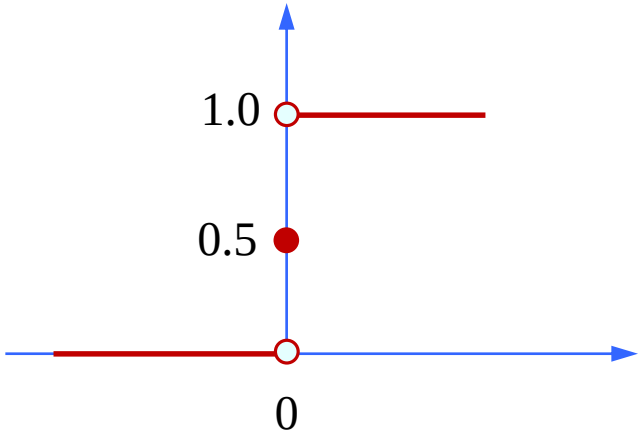
$$y = g^{-1}(\mathbf{w}^T\mathbf{x}+b)$$

2.4 对数几率回归

- **对数几率回归**
  - **如何用线性回归做分类任务，应该怎么做呢？**
  - 一种常用的做法是采用类别标签向量：
    - 对于两类分类问题，我们期望正例样本被回归到“+1”,负例样本被回归到“-1”。
    - 对于多类问题，通常将样本回归到**one-zero hot**向量，该向量只有一个元素为1，其余元素全为零。
  - 对数几率回归则利用广义线性回归的思想，期望找到一个单调可微函数**将分类任务的真实标记**与**线性回归模型的预测值**联系起来。

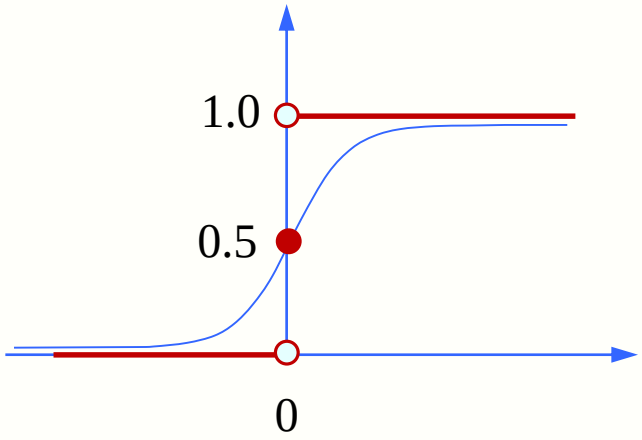
对数几率回归

- 考虑两类分类问题，其输出标记为{0,1}，而线性回归模型产生的预测值$z=\mathbf{w}^T\mathbf{x}+b$是实值。**于是我们将实值z转换为0/1值**。为此可采用单位阶跃函数：



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

- 阶跃函数不连续，且反函数不存在。需要找到可近似单位阶跃函数的替换函数（surrogate function）。
- 对数几率函数（logistic function）正是这样的函数（一种sigmoid函数）：



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

2.4 对数几率回归

- 真实的变换函数

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

- 本质上是线性的：

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

若将 y 视为正例的可能性，$1-y$ 则为负例的可能性，因此 $y/(1-y)$ 称为几率(odds)，度量样本 $\mathbf{x}$ 作为正例的相对可能性。对几率取对数则得到对数几率。

2.4 对数几率回归

- **分析**
  - 对数几率回归是一种两类分类方法。
  - 对数几率回归（logistic regression, 亦称logit regression）也称为逻辑斯特回归方法。
  - 输出样本分类结果的可能性（软标签）。

2.4 对数几率回归

- **软标签**
  - 如果将 y 视为类后验概率估计 $p(y=1\,|\,\mathbf{x})$，有：

$$\ln\frac{p(y=1\,|\,\mathbf{x})}{p(y=0\,|\,\mathbf{x})}=\ln\frac{y}{1-y}=\mathbf{w}^T\mathbf{x}+b$$

  - 进一步，由变换可获得预测概率：

$$y=\frac{1}{1+\exp^{-(\mathbf{w}^T\mathbf{x}+b)}}\quad\Rightarrow\quad p(y=1\,|\,\mathbf{x})=\frac{\exp^{\mathbf{w}^T\mathbf{x}+b}}{1+\exp^{\mathbf{w}^T\mathbf{x}+b}}$$

$$p(y=0\,|\,\mathbf{x})=\frac{1}{1+\exp^{\mathbf{w}^T\mathbf{x}+b}}$$

- 学习w和*b*

$$p(y=1|\mathbf{x})=\frac{\exp^{\mathbf{w}^T\mathbf{x}+b}}{1+\exp^{\mathbf{w}^T\mathbf{x}+b}} \qquad p(y=0|\mathbf{x})=\frac{1}{1+\exp^{\mathbf{w}^T\mathbf{x}+b}}$$

$$p(y_i|\mathbf{x}_i;\mathbf{w},b)=y_i\,p(y_i=1|\mathbf{x}_i;\mathbf{w},b)+(1-y_i)\,p(y_i=0|\mathbf{x}_i;\mathbf{w},b)$$

对数似然函数：

$$l(\mathbf{w},b)=\sum_{i=1}^{n}\ln\,p(y_i|\mathbf{x}_i;\mathbf{w},b)$$

$$l(\hat{\mathbf{w}})=\sum_{i=1}^{n}\ln\left(y_i\,p(y_i=1|\hat{\mathbf{x}}_i;\hat{\mathbf{w}})+(1-y_i)\,p(y_i=0|\hat{\mathbf{x}}_i;\hat{\mathbf{w}})\right)$$

$$=\sum_{i=1}^{n}\ln\left(\frac{\left(y_i\exp^{\mathbf{w}^T\mathbf{x}+b}\right)+(1-y_i)}{1+\exp^{\mathbf{w}^T\mathbf{x}+b}}\right)$$

$$=\sum_{i=1}^{n}\left(y_i(\mathbf{w}^T\mathbf{x}+b)-\ln\left(1+\exp^{\mathbf{w}^T\mathbf{x}+b}\right)\right)$$

$$\ln\left(y_i\exp^{\mathbf{w}^T\mathbf{x}+b}+1-y_i\right)=y_i(\mathbf{w}^T\mathbf{x}+b)$$

在最大对数似然的框架下学习w和b：

$$\max l(\hat{\mathbf{w}}) \quad \Leftrightarrow \quad \min g(\hat{\mathbf{w}}) \quad \Leftrightarrow \quad \min \sum_{i=1}^{n} \left(\ln\left(1+\exp^{\mathbf{w}^T\mathbf{x}+b}\right) - y_i(\mathbf{w}^T\mathbf{x}+b) \right)$$

可采用梯度法或牛顿法进行求解。比如采用牛顿法：

$$\hat{\mathbf{w}}^{t+1} = \hat{\mathbf{w}}^{t} - \left(\frac{\partial^2 g(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}} \partial \hat{\mathbf{w}}^T} \right)^{-1} \frac{\partial g(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}}$$

（牛顿法的一般表示）

$$\frac{\partial g(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}} = -\sum_{i=1}^{m} \hat{\mathbf{x}}_i (y_i - p(y_i = 1 | \hat{\mathbf{x}}_i; \hat{\mathbf{w}}))$$

$$\frac{\partial^2 g(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}} \partial \hat{\mathbf{w}}^T} = \frac{\partial}{\partial \hat{\mathbf{w}}^T} \left(\frac{\partial g(\hat{\mathbf{w}})}{\partial \hat{\mathbf{w}}^T} \right)^T = \sum_{i=1}^{m} \hat{\mathbf{x}}_i \hat{\mathbf{x}}_i^T p(y_i = 1 | \hat{\mathbf{x}}_i; \hat{\mathbf{w}})(1 - p(y_i = 1 | \hat{\mathbf{x}}_i; \hat{\mathbf{w}}))$$

(Hessian 矩阵)

- **小结**
  - 给定n个训练样本$(\mathbf{x}_1, y_1), \ldots, (\mathbf{x}_n, y_n)$，其中$\mathbf{x}_i \in R^d$，$i=1:n$为 d 维样本特征，$y_i \in \{0,1\}$为其对应的类别标签。
  - Logistic 回归采用的假设函数：

$$h(\mathbf{x} \mid \mathbf{w}, b) = \frac{1}{1+\exp^{-(\mathbf{w}^T \mathbf{x}+b)}}$$

  - 目标：训练模型参数$(\mathbf{w}, b)$，最小化代价函数：
    - 采用最大似然准则：

$$l(\mathbf{w}, b) = -\sum_{i=1}^{n} \log\left(y_i h(\mathbf{x}_i \mid \mathbf{w}, b) + (1-y_i)(1-h(\mathbf{x}_i \mid \mathbf{w}, b))\right)$$

    - 也可以采用交叉熵损失：

$$l(\mathbf{w}, b) = -\sum_{i=1}^{n} \left(y_i \log\left(h(\mathbf{x}_i \mid \mathbf{w}, b)\right) + (1-y_i)\log\left(1-h(\mathbf{x}_i \mid \mathbf{w}, b)\right)\right)$$

$$\text{loss of cross entropy} = -\left(p_{true} \log p_{predicted} + (1-p_{true}) \log(1-p_{predicted})\right)$$

2.5 Softmax回归

- **问题的背景**
  - Logistic回归是一种两类分类算法，具有很多应用，如广告计算等。
  - Logistic回归利用后验概率最大化去计算权重**w**。
  - 但它不能处理多类分类问题。
  - 可以采用一对多或一对一的策略来构造多个分类器，然后采用最大输出决策或投票决策来实现多类分类。
  - **Softmax Regression是Logistic回归的推广**，能以更加紧凑的方式来处理 Logistic 回归中所面临的多类分类问题。

2.5 Softmax回归

- **Softmax函数**
  - 设$\mathbf{x}=[x_1, x_2,\ldots, x_c]^T$为一个$c$维空间中的一个向量，softmax函数$\sigma$是一个$R^c$ 到 $[0,1]^c$上的一个c维映射函数：

$$[\sigma(\mathbf{x})]_j = \frac{e^{x_j}}{\sum_{i=1}^{c} e^{x_i}}, \quad j = 1, 2, \cdots, c$$

  - Softmax函数的输出可以用来描述关于类别的分布：

$$P(y = j \mid \mathbf{x}) = \frac{e^{\mathbf{w}_j^T \mathbf{x} + b_j}}{\sum_{i=1}^{c} e^{\mathbf{w}_i^T \mathbf{x} + b_i}}, \quad j = 1, 2, \cdots, c$$

2.5 Softmax回归

- **The task of softmax regression**
  - 给定n个训练样本$(\mathbf{x}_1, y_1)$, $(\mathbf{x}_2, y_2)$, …, $(\mathbf{x}_n, y_n)$ ，其中$\mathbf{x}_i \in R^d$， $i = 1,2,\ldots, n$为 d 维空间中的样本特征，$y_i \in \{1,2, \ldots, c\}$为其对应的类别标签。
  - 给定测试样本$\mathbf{x}$，期望基于所学的假设函数能对每个类别 j 估算出概率值 $P(y = j \mid \mathbf{x})$，即**估计 x 的每一种分类结果出现的概率**。
  - 因此，假设函数应该有能力输出一个 c 维的向量（向量元素的和为1）来表示这 c 个估计的概率值。

2.5 Softmax回归

- Hypothesis function
  - 就是softmax函数！所有softmax函数构成假设空间。
  - 假定数据**x**是采用齐坐标表示的，即**x**的维数是d+1维。齐次坐标则对应着平移量b。
  - 假设函数的具体形式（概率形式）：

$$\mathbf{h}(\mathbf{x};\mathbf{W})=\begin{pmatrix} P(y=1\,|\,\mathbf{x};\mathbf{W}) \\ P(y=2\,|\,\mathbf{x};\mathbf{W}) \\ \vdots \\ P(y=c\,|\,\mathbf{x};\mathbf{W}) \end{pmatrix}=\frac{1}{\sum_{i=1}^{c} e^{\mathbf{w}_i^T\mathbf{x}}}\begin{pmatrix} e^{\mathbf{w}_1^T\mathbf{x}} \\ e^{\mathbf{w}_2^T\mathbf{x}} \\ \vdots \\ e^{\mathbf{w}_c^T\mathbf{x}} \end{pmatrix} \in R^c$$

where $\mathbf{W}=[\mathbf{w}_1,\mathbf{w}_2,\cdots,\mathbf{w}_c]\in R^{(d+1)\times c}$

2.5 Softmax回归

- 代价函数：从两类到多类

采用交叉熵损失

$$l(\mathbf{w},b)=-\sum_{i=1}^{n}\left(y_i\log\left(h(\mathbf{x}_i\,|\,\mathbf{w},b)\right)+\left(1-y_i\right)\log\left(1-h(\mathbf{x}_i\,|\,\mathbf{w},b)\right)\right)$$

$$=-\sum_{i=1}^{n}\left[\sum_{j=0}^{1}\delta(y_i=j)\log P(y_i=j\,|\,\mathbf{x}_i\,|\,\mathbf{w},b)\right]$$

$\delta(\cdot)$为指示函数

$$l(\mathbf{W})=-\sum_{i=1}^{n}\left[\sum_{j=1}^{c}\delta(y_i=j)\log\frac{e^{\mathbf{w}_j^T\mathbf{x}_i}}{\sum_{k=1}^{c}e^{\mathbf{w}_k^T\mathbf{x}_i}}\right]$$

$\delta(\cdot)$为指示函数

2.5 Softmax回归

- 求解

$$l(\mathbf{W}) = -\sum_{i=1}^{n}\left[\sum_{j=1}^{c}\delta(y_i = j)\log\frac{e^{\mathbf{w}_j^T\mathbf{x}_i}}{\sum_{k=1}^{c}e^{\mathbf{w}_k^T\mathbf{x}_i}}\right]$$

$$\frac{\partial l(\mathbf{W})}{\partial \mathbf{w}_j} = -\sum_{i=1}^{n}\left(\mathbf{x}_i\delta(y_i = j) - \frac{e^{\mathbf{w}_j^T\mathbf{x}_i}}{\sum_{k=1}^{c}e^{\mathbf{w}_k^T\mathbf{x}_i}}\right) = \sum_{i=1}^{n}\mathbf{x}_i\left(\delta(y_i = j) - P(y_i = j \mid \mathbf{x}; \mathbf{W})\right)$$

$$j = 1, 2, \cdots, c$$

采用梯度下降低法：$\mathbf{w}_j^{t+1} = \mathbf{w}_j^t - \eta\frac{\partial l(\mathbf{W})}{\partial \mathbf{w}_j^t}, \quad j = 1, 2, \cdots, c$

2.5 Softmax回归

- 参数化特点

对任意 $\theta \in R^{d+1}$:

$$P(y=j \mid \mathbf{x};\mathbf{W}) = \frac{e^{(\mathbf{w}_j-\boldsymbol{\theta})^T\mathbf{x}_i}}{\sum_{k=1}^{c} e^{(\mathbf{w}_k-\boldsymbol{\theta})^T\mathbf{x}_i}} = \frac{e^{\mathbf{w}_j^T\mathbf{x}_i}e^{-\boldsymbol{\theta}^T\mathbf{x}_i}}{\sum_{k=1}^{c} e^{\mathbf{w}_k^T\mathbf{x}_i}e^{-\boldsymbol{\theta}^T\mathbf{x}_i}} = \frac{e^{\mathbf{w}_j^T\mathbf{x}_i}}{\sum_{k=1}^{c} e^{\mathbf{w}_k^T\mathbf{x}_i}}$$

这表明：上述 softmax 回归模型中存在冗余的参数。即，Softmax 模型被过度参数化：对于任意一个用于拟合数据的假设函数，可以求出多组参数值，这些参数得到的是完全相同的假设函数 $\mathbf{h}(\mathbf{x};\mathbf{W})$。

- 参数化特点
  - 因此，使 $l(\mathbf{W})$ 最小化的解不是唯一的。
  - 有趣的是，由于$l(\mathbf{W})$ 是一个凸函数，梯度下降时不会遇到局部最优解的问题。
  - 但是，Hessian 矩阵是奇异的/不可逆的，因此采用牛顿法优化会遇到数值计算的问题。
  - 注意，当$\theta=\mathbf{w}_1$ 时，总可以将$\mathbf{w}_1$替换为$\mathbf{w}_1-\theta$ ，并且这种变换不会影响假设函数。
    - 因此，可以去掉参数$\mathbf{w}_1$（或者其它任意一个），但不影响假设函数的表达能力。
    - 可以令$\mathbf{w}_1=\mathbf{0}$，只优化剩余的$c-1$个参数。
    - 但是，实际中我们很少这样做！

2.5 Softmax回归

- 权重衰减：新的学习模型

$$l(\mathbf{W}) = -\sum_{i=1}^{n}\left[\sum_{j=1}^{c}\delta(y_i = j)\log\frac{e^{\mathbf{w}_j^T\mathbf{x}_i}}{\sum_{k=1}^{c}e^{\mathbf{w}_k^T\mathbf{x}_i}}\right] + \lambda\|\mathbf{W}\|_F^2$$

$$\frac{\partial l(\mathbf{W})}{\partial \mathbf{w}_j} = \sum_{i=1}^{n}\mathbf{x}_i\left(\delta(y_i = j) - P(y_i = j \mid \mathbf{x}; \mathbf{W})\right) + 2\lambda\mathbf{w}_j$$

引入权重衰减项（正则项）后，代价函数就变成了严格的凸函数，可保证得到唯一的解。此时的 Hessian矩阵变为可逆矩阵，并且因为是凸函数，梯度下降法和 L-BFGS 等算法可以保证收敛到全局最优解。

2.5 Softmax回归

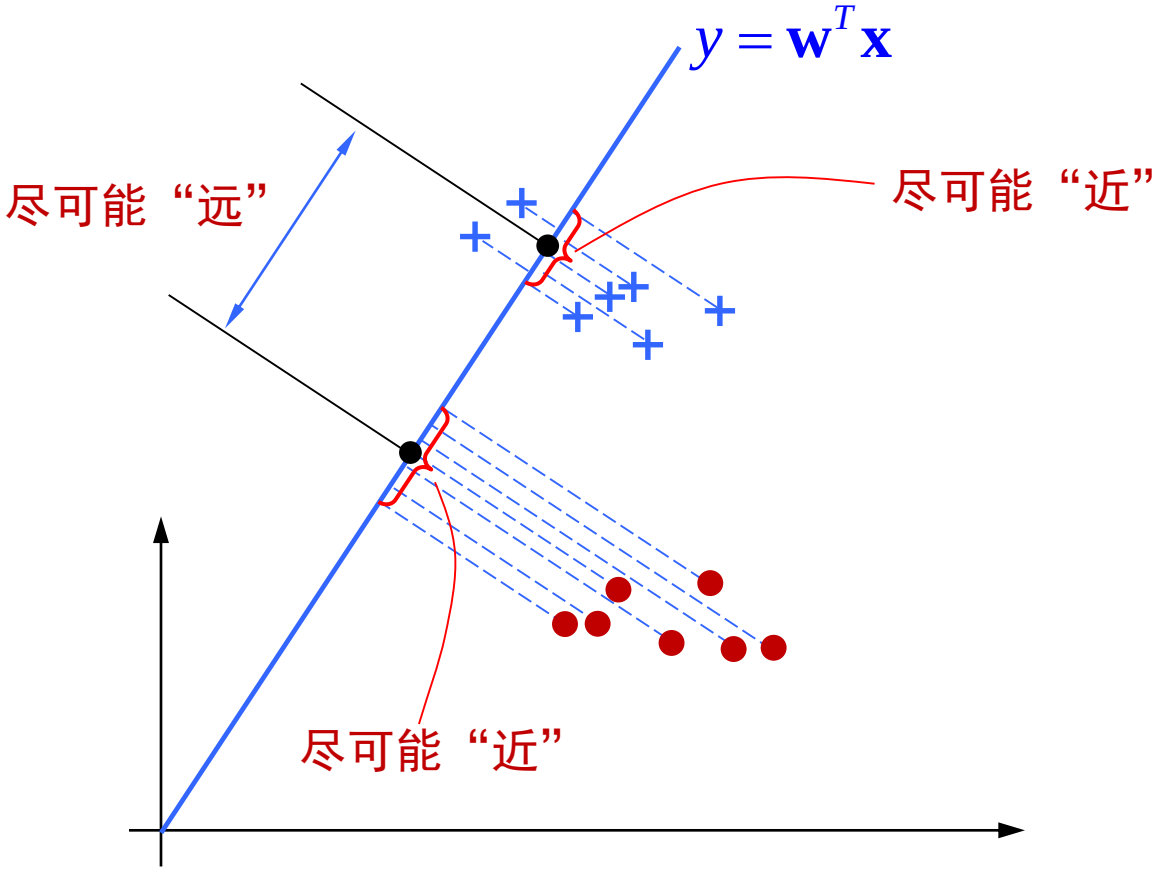
- Sofmax regression VS 多个 Logistic regression
  - 将图像分到三个不同类别中。
    - (i) 假设这三个类别分别是：室内场景、户外城区场景、户外荒野场景。
    - (ii) 现在假设这三个类别分别是室内场景、黑白图像、包含人物的图像。
    - 在第一个例子中，三个类别是互斥的，因此更适于选择softmax回归分类器。
    - 在第二个例子中，建立三个独立的logistic回归分类器可能更加合适。

2.6 线性判别分析

- **算法思想**
  - 线性判别分析（Linear Discriminant Analysis, LDA）是一种经典的线性学习方法。
  - LAD的思想较直观：对于两类分类问题，给定训练集，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，不同类样例的投影点尽可能相互远离。
  - 在对新样本进行分类时，奖其投影到这条直线上，再根据投影点的位置来判断其类别。

2.6 线性判别分析

- 算法思想



2.6 线性判别分析

- **算法思想**
  - 样本集$D=\{(\mathbf{x}_1, y_1), (\mathbf{x}_2, y_2),\ldots,(\mathbf{x}_n, y_n)\}$，$y_i\in\{0,1\}$，令$\mathbf{X}_i$、$\boldsymbol{\mu}_i$、$\boldsymbol{\Sigma}_i$分别表示第$i\in\{0,1\}$类的示例集合、均值向量、协方差矩阵。
  - 若将数据到直线上，则两类样本的中心在直线上的投影分别为$\mathbf{w}^T\boldsymbol{\mu}_0$和$\mathbf{w}^T\boldsymbol{\mu}_1$。
  - 若将两类样本点投影到直线，则两类样本的协方差分别为$\mathbf{w}^T\boldsymbol{\Sigma}_0\mathbf{w}$和$\mathbf{w}^T\boldsymbol{\Sigma}_1\mathbf{w}$。
  - 欲使同类样本的投影点尽可能接近，可以让同类样本投影点的协方差尽可能小，即$\mathbf{w}^T\boldsymbol{\Sigma}_0\mathbf{w}+\mathbf{w}^T\boldsymbol{\Sigma}_1\mathbf{w}$。
  - 欲使异类样本的投影点尽可能远离，则可让类中心点之间的距离尽可能大，即$\|\mathbf{w}^T\boldsymbol{\mu}_0-\mathbf{w}^T\boldsymbol{\mu}_1\|^2$尽可能大。

2.6 线性判别分析

- **算法思想：**最大化如下目标函数

两个类的中心尽可能远

$$J = \frac{\left\| \mathbf{w}^T \boldsymbol{\mu}_0 - \mathbf{w}^T \boldsymbol{\mu}_1 \right\|_2^2}{\mathbf{w}^T \boldsymbol{\Sigma}_0 \mathbf{w} + \mathbf{w}^T \boldsymbol{\Sigma}_1 \mathbf{w}}$$

两类的类内协方差尽可能小

$$= \frac{\mathbf{w}^T (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T \mathbf{w}}{\mathbf{w}^T (\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1) \mathbf{w}}$$

  - 根据上述算法思想，我们可以定义一些量，从而将算法进行推广。

2.6 线性判别分析

- **LAD算法**
  - 类内散度矩阵：
$$\mathbf{S}_w = \boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1 = \sum_{\mathbf{x}\in X_0} (\mathbf{x}-\boldsymbol{\mu}_0)(\mathbf{x}-\boldsymbol{\mu}_0)^T + \sum_{\mathbf{x}\in X_1} (\mathbf{x}-\boldsymbol{\mu}_1)(\mathbf{x}-\boldsymbol{\mu}_1)^T$$
  - 类间散度矩阵：
$$\mathbf{S}_b = (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T$$
  - 目标函数重写为（**广义Rayleigh商**）：
$$J = \frac{\mathbf{w}^T \mathbf{S}_b \mathbf{w}}{\mathbf{w}^T \mathbf{S}_w \mathbf{w}}$$

注意：J的值与向量的长度无关，只与其方向有关，不失一般性可令$\mathbf{w}$为单位长度的向量。

2.6 线性判别分析

- **LAD算法**
  - 学习目标： $\max \dfrac{\mathbf{w}^T\mathbf{S}_b\mathbf{w}}{\mathbf{w}^T\mathbf{S}_w\mathbf{w}}, \quad s.t. \quad \mathbf{w}^T\mathbf{w}=1$
  - 由于目标函数值与长度无关（只与方向有关），因此可采用一种更直观的方法：令 $\mathbf{w}^T\mathbf{S}_w\mathbf{w}=1$：

$$\max \ \mathbf{w}^T\mathbf{S}_b\mathbf{w}, \quad s.t. \quad \mathbf{w}^T\mathbf{S}_w\mathbf{w}=1$$

  - 根据拉格良日乘子法，于是有：

$$\mathbf{S}_b\mathbf{w}=\lambda\mathbf{S}_w\mathbf{w} \quad \Rightarrow \quad \mathbf{S}_w^{-1}\mathbf{S}_b\mathbf{w}=\lambda\mathbf{w}$$

上式表明： w为是矩阵 $\mathbf{S}_w^{-1}\mathbf{S}_b$ 的特征向量。

2.6 线性判别分析

- **LAD算法：构造性求解方法**

$$\mathbf{S}_b = (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T$$

$$\mathbf{S}_b\mathbf{w} = (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T\mathbf{w} = s\cdot(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1), \quad s = (\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^T\mathbf{w} \in R\text{（标量）}$$

上式表明： $\mathbf{S}_b\mathbf{w}$ 方向与$\boldsymbol{\mu}_0$–$\boldsymbol{\mu}_1$的方向相同。 不妨令：

$$\mathbf{S}_b\mathbf{w} = \lambda(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)$$

$$\mathbf{w} = \mathbf{S}_w^{-1}(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)$$

2.6 线性判别分析

- **多类LAD算法** (设类别数为c)
  - 全局散度矩阵： $\mathbf{S}_t = \mathbf{S}_w + \mathbf{S}_b = \sum_{i=1}^{n} (\mathbf{x}_i - \boldsymbol{\mu})(\mathbf{x}_i - \boldsymbol{\mu})^T, \quad \boldsymbol{\mu} = \frac{1}{n}\sum_{i=1}^{n} \mathbf{x}_i$
  - 类内散度矩阵： $\mathbf{S}_w = \sum_{j=1}^{c} \mathbf{S}_{wj}, \quad \mathbf{S}_{wj} = \sum_{\mathbf{x} \in X_j} (\mathbf{x} - \boldsymbol{\mu}_j)(\mathbf{x} - \boldsymbol{\mu}_j)^T, \quad \boldsymbol{\mu}_j = \frac{1}{n_j}\sum_{\mathbf{x} \in X_j} \mathbf{x}$
  - 类间散度矩阵： $\mathbf{S}_b = \mathbf{S}_t - \mathbf{S}_w = \sum_{j=1}^{c} n_j (\boldsymbol{\mu}_j - \boldsymbol{\mu})(\boldsymbol{\mu}_j - \boldsymbol{\mu})^T$

其中，n_j 为属于第 j 类的样本个数。

注：矩阵$\mathbf{S}_b$的秩小于等于$d-1$！

2.6 线性判别分析

- **多类LAD算法**

最大化投影后的距离

Problem 1:

$$\max \frac{\mathrm{tr}\left(\mathbf{W}^T\mathbf{S}_b\mathbf{W}\right)}{\mathrm{tr}\left(\mathbf{W}^T\mathbf{S}_b\mathbf{W}\right)}, \quad s.t. \quad \mathbf{W}^T\mathbf{W}=\mathbf{I}$$

（迹比值最大化）

最大化投影后数据分布的体积

Problem 2:

$$\max \frac{\left|\mathbf{W}^T\mathbf{S}_b\mathbf{W}\right|}{\left|\mathbf{W}^T\mathbf{S}_w\mathbf{W}\right|}, \quad s.t. \quad \mathbf{W}^T\mathbf{W}=\mathbf{I}$$

（行列式比值最大化）

行列式

2.6 线性判别分析

- **多类LAD算法**
  - 需要指出的是：Problem 1与 Problem 2的解是不同的。
    Problem 2的解可以通过如下广义特征值问题求解得到：

$$\mathbf{S}_b\mathbf{w}=\lambda\mathbf{S}_w\mathbf{w}$$

  - **Problem 1的求解较复杂**，可以参考如下文献：

Shiming Xiang, Feiping Nie, Changshui Zhang. Learning a Mahalanobis distance metric for data clustering and classification. Pattern Recognition, 41(12), Pages 3600 - 3612, 2008

$$\max \quad \frac{\mathrm{tr}(\mathbf{W}^T\mathbf{S}_b\mathbf{W})}{\mathrm{tr}(\mathbf{W}^T\mathbf{S}_w\mathbf{W})}, \quad s.t. \quad \mathbf{W}^T\mathbf{W}=\mathbf{I}$$

2.7 局部线性判别分析

- The re-computation of $\mathbf{S}_b$ and $\boldsymbol{S}_w$

$$\mathbf{S}_w = \sum_{i=1}^{c} \sum_{j:y_j=i} (\mathbf{x}_j - \boldsymbol{\mu}_i)(\mathbf{x}_j - \boldsymbol{\mu}_i)^T = \frac{1}{2} \sum_{i,j} A_{ij}^{(w)} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

$$\mathbf{S}_b = \sum_{i=1}^{c} n_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})^T (\boldsymbol{\mu}_i - \boldsymbol{\mu}) = \frac{1}{2} \sum_{i,j=1}^{n} A_{ij}^{(b)} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

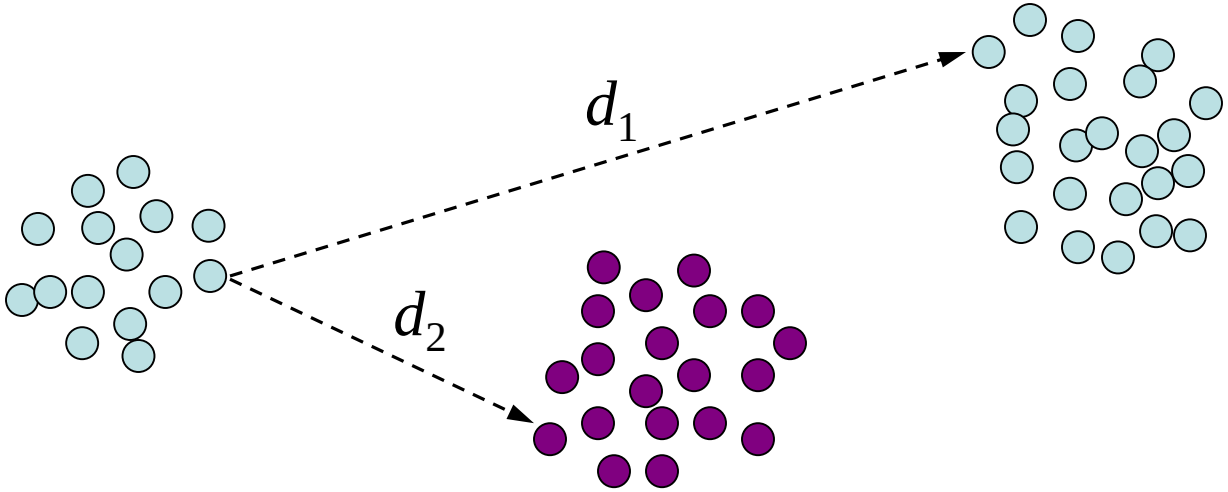
$$\boldsymbol{\mu}_i = \frac{1}{n_i} \sum_{j:y_j=i} \mathbf{x}_j \qquad \boldsymbol{\mu} = \frac{1}{n} \sum_{i=1}^{n} \mathbf{x}_j$$

$$A_{ij}^{(w)} = \begin{cases} 1/n_k, & \text{if } y_i = y_j = k \\ 0, & \text{if } y_i \neq y_j \end{cases} \qquad A_{ij}^{(b)} = \begin{cases} 1/n - 1/n_k, & \text{if } y_i = y_j = k \\ 1/n, & \text{if } y_i \neq y_j \end{cases}$$

这里：下标y_i表示样本$\mathbf{x}_i$的类别标签，即$y_i \in \{1,2,\ldots c\}$。另外，$k \in \{1,2,\ldots c\}$

2.7 局部线性判别分析

- 局限性
  - In each class, the distribution of data is Gaussian

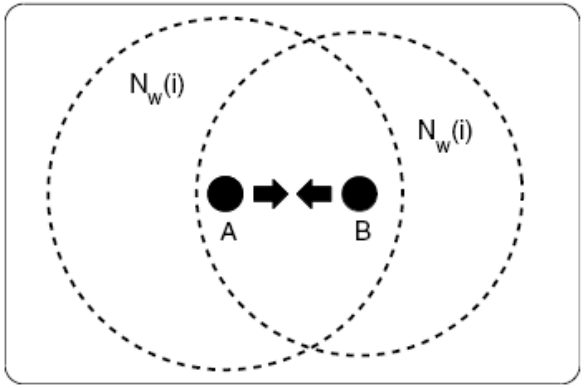


We hope $d_1 < d_2$. **But difficult, or impossible!**

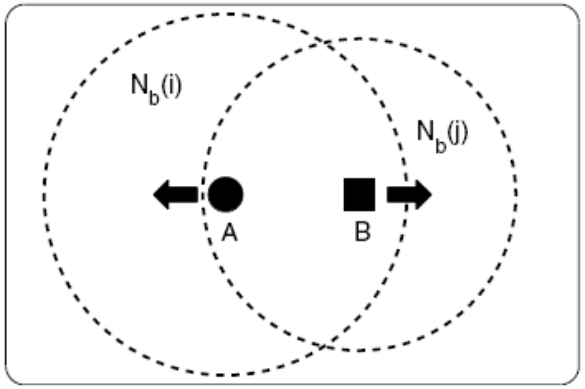
2.7 局部线性判别分析

- Techniques of Local Analysis
  - Neighborhood constraints （方法一）
  - Locally weighting （方法二）
    - Weighting for 1-NN
    - Local Fisher discriminant analysis

Motivation



within class



between classes

2.7 局部线性判别分析

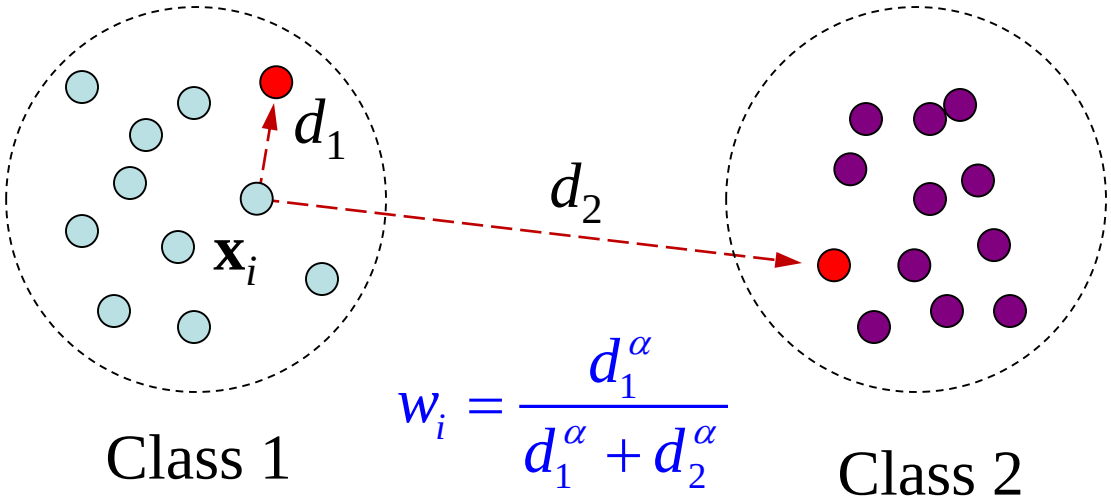
- **Modify $\mathbf{S}_w$ and $\mathbf{S}_b$**
  - Neighborhood Constraints:

$$\mathbf{S}_w = \sum_{\substack{y_i = y_j \\ \mathbf{x}_i \in N(\mathbf{x}_j),\, \mathbf{x}_j \in N(\mathbf{x}_i)}} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

$$\mathbf{S}_b = \sum_{\substack{y_i \neq y_j \\ \mathbf{x}_i \in N(\mathbf{x}_j),\, \mathbf{x}_j \in N(\mathbf{x}_i)}} (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

2.7 局部线性判别分析

- **Nearest Neighbor Discriminant Analysis, NNDA**
  - **近邻加权**
    - A problem in neighborhood constraints is the selection of the number of nearest neighbors (k)

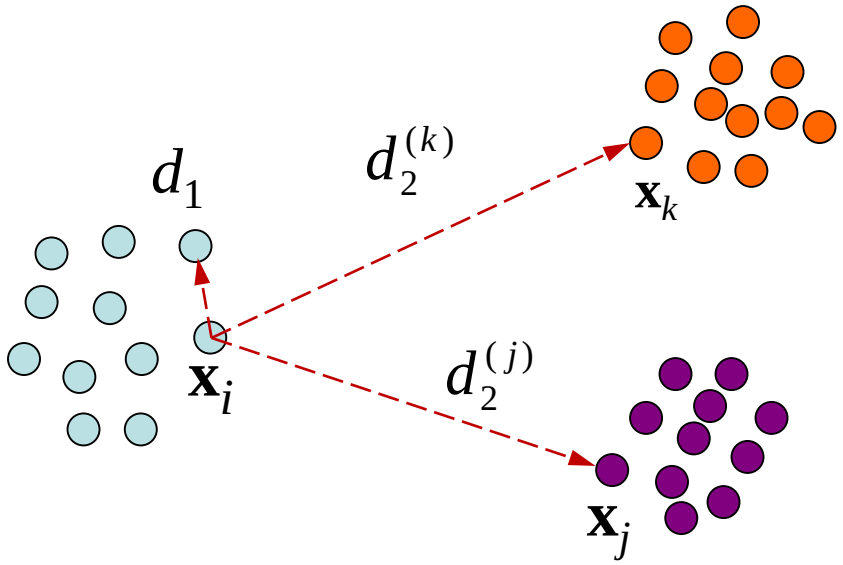


最近邻加权

2.7 局部线性判别分析

- **Nearest Neighbor Discriminant Analysis, NNDA**

  多类最近邻加权



$$w_i = \frac{d_1^{\alpha}}{d_1^{\alpha} + (\min\{d_2^{(j)}\})^{\alpha}}, \quad (0<\alpha<1)$$

2.7 局部线性判别分析

- **Nearest Neighbor Discriminant Analysis, NNDA**

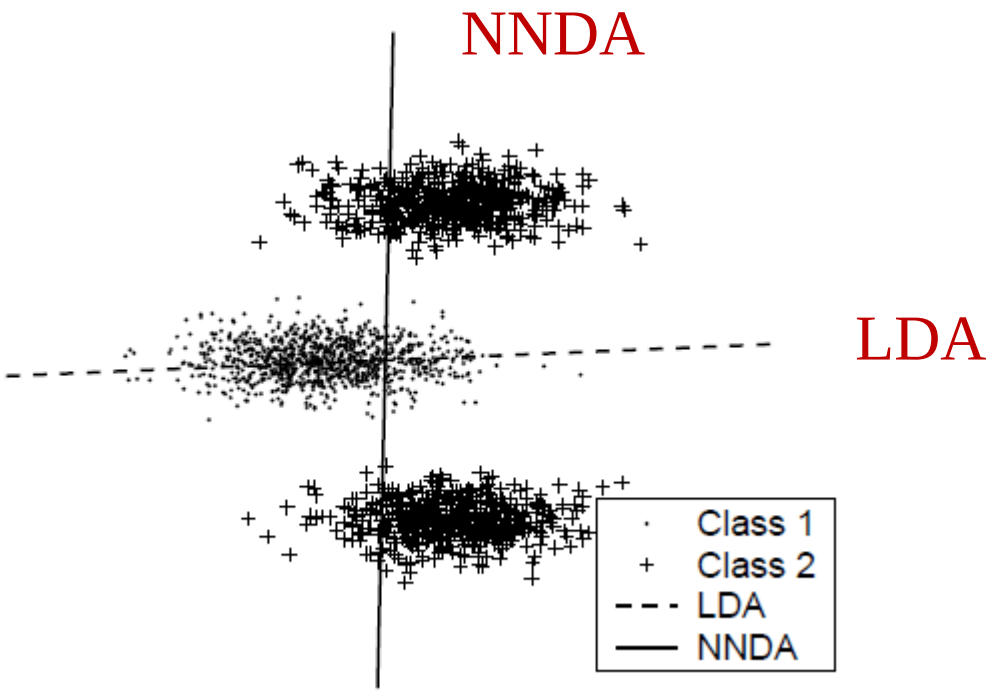
$$\mathbf{S}_w = \sum_{\substack{y_i = y_j \\ \mathbf{x}_j \in N_{1-nn}(\mathbf{x}_i),\, i=1,2,\cdots,n}} w_i (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

$$\mathbf{S}_b = \sum_{\substack{y_i \neq y_j \\ \mathbf{x}_j \in N_{1-nn}(\mathbf{x}_i),\, i=1,2,\cdots,n}} w_i (\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

Xipeng Qiu, Lide Wu: Stepwise Nearest Neighbor Discriminant Analysis. IJCAI 2005: 829-834

2.7 局部线性判别分析

- **Nearest Neighbor Discriminant Analysis, NNDA**



NNDA finds the correct projection direction, but LDA failed !

2.7 局部线性判别分析

- **Local Fisher Discriminant Analysis, LFDA**
  - Motivation
    - LFDA does not impose far-apart data pairs of the same class to be close, by which local structure of the data tends to be preserved.
    - 邻域加权 (Locally Weighting)

Masashi Sugiyama, Local Fisher Discriminant Analysis for Supervised Dimensionality Reduction, ICML, 2006

2.7 局部线性判别分析

- Step1: Construct an affine matrix for *n* data points:

$$\mathbf{A}=\begin{pmatrix} A_{11} & A_{12} & \cdots & A_{1n} \\ A_{21} & A_{22} & \cdots & A_{2n} \\ \cdots & \cdots & \cdots & \cdots \\ A_{n1} & A_{n2} & \cdots & A_{nn} \end{pmatrix}$$

$$A_{ij}=\begin{cases} 1, & \text{if } \mathbf{x}_j \text{ is a neighbor of } \mathbf{x}_i \\ 0, & \text{otherwise} \end{cases}, \text{ or}$$

$$A_{ij}=\begin{cases} \exp(-\|\mathbf{x}_j-\mathbf{x}_i\|^2/(2\sigma^2)), & \text{if } \mathbf{x}_j \text{ is a neighbor of } \mathbf{x}_i \\ 0, & \text{otherwise} \end{cases}$$

2.7 局部线性判别分析

- Step2: Modify $\mathbf{S}_w$ and $\mathbf{S}_b$:

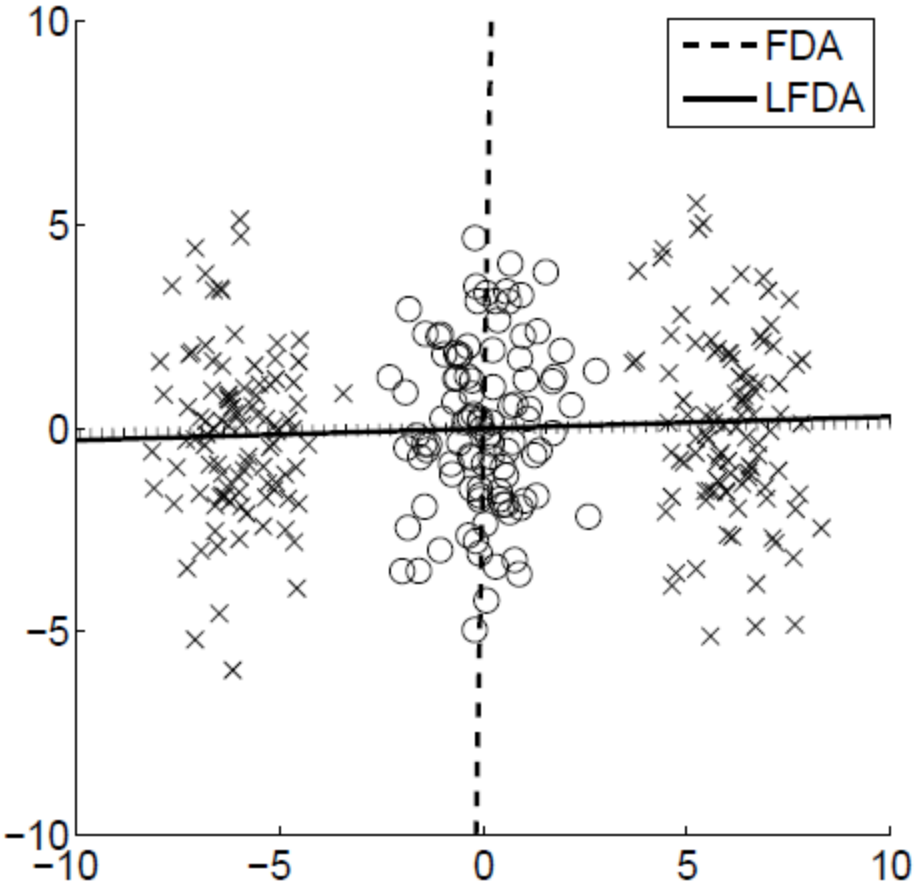
$$\mathbf{S}_w = \frac{1}{2}\sum_{i,j} A_{ij}^{(w)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \qquad \mathbf{S}_b = \frac{1}{2}\sum_{i,j=1}^{n} A_{ij}^{(b)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

$$\mathbf{S}_w = \frac{1}{2}\sum_{i,j} \bar{A}_{ij}^{(w)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T \qquad \mathbf{S}_b = \frac{1}{2}\sum_{i,j=1}^{n} \bar{A}_{ij}^{(b)}(\mathbf{x}_i - \mathbf{x}_j)(\mathbf{x}_i - \mathbf{x}_j)^T$$

$$\bar{A}_{ij}^{(w)} = \begin{cases} A_{ij}/n_c, & \text{if } y_i = y_j = c \\ 0, & \text{if } y_i \neq y_j \end{cases} \qquad \bar{A}_{ij}^{(b)} = \begin{cases} A_{ij}(1/n - 1/n_c), & \text{if } y_i = y_j = c \\ 1/n, & \text{if } y_i \neq y_j \end{cases}$$

2.7 局部线性判别分析

- Demo

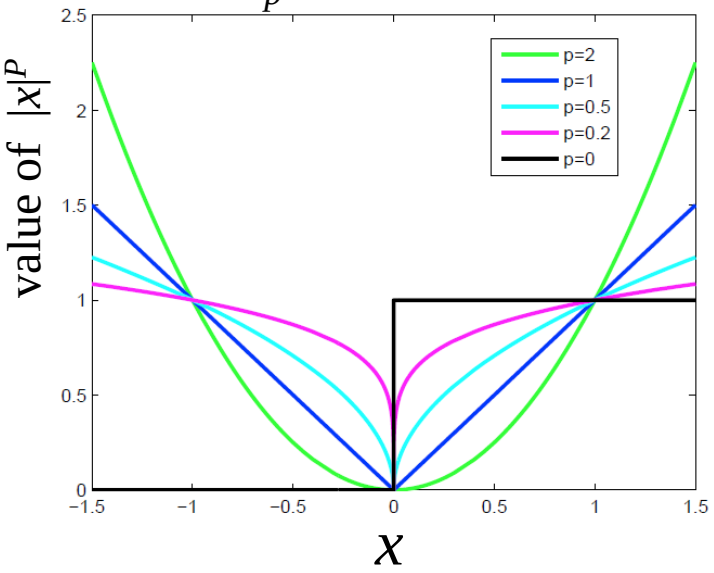


2.8 特征选择

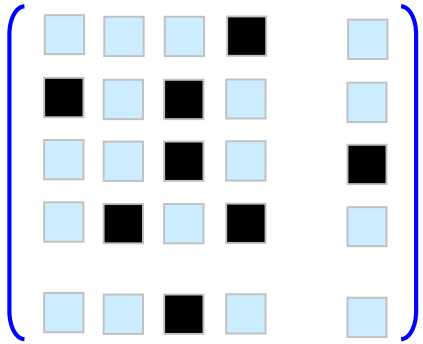
- **稀疏学习**
  - 针对具体的学习问题，可在线性模型中引入恰当的**稀疏约束条件或稀疏性度量。**
    - 稀疏是一种先验（比如：服从拉普拉斯分布）。
    - 稀疏是对某种已知知识的描述。
    - 从结构化风险最小化的角度，**引入稀疏约束条件是增加所学函数在假设空间的简单性**，所学系数向量越稀疏，则函数越简单。
    - **从正则化的角度看，就是为了防止过拟合，提高线性最小二乘法所学模型的泛化能力。**
    - **……**

2.8 特征选择

L_p范数示意图



- **模型扩展：稀疏学习**
  - 向量稀疏性度量：
    - L_0: $\mathbf{v} = [\blacksquare\ \blacksquare\ \blacksquare\ \blacksquare\ \blacksquare\ \blacksquare]^T$, $\|\mathbf{v}\|_0 = 3$
    - L_1: $\|\mathbf{v}\|_1 = \sum_i |v_i|$
    - 采用 L_1 来近似 L_0 — 凸近似
  - 矩阵稀疏性度量：

$\mathbf{W}=$  ⇨ $\|\mathbf{W}\|_1 = \sum_{i,j} |W_{ij}|$

2.8 特征选择

- **模型扩展: 稀疏表示**

$$\mathbf{y} = 0.7\mathbf{a}_1 + 0.0\mathbf{a}_2 + 0.0\mathbf{a}_3 + 0.0\mathbf{a}_4 + 0.2\mathbf{a}_5 + 0.0\mathbf{a}_6 + 0.0\mathbf{a}_7 + 0.1\mathbf{a}_8 + 0.0\mathbf{a}_9$$

L_1-范数松驰 ⇩

$$(\mathrm{p}_0) \quad \min_{\mathbf{w}} \|\mathbf{w}\|_0, \quad \text{subject to } \mathbf{X}\mathbf{w} = \mathbf{y}$$

$$(\mathrm{p}_0) \quad \min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2, \quad \text{subject to } \|\mathbf{w}\|_0 < t$$

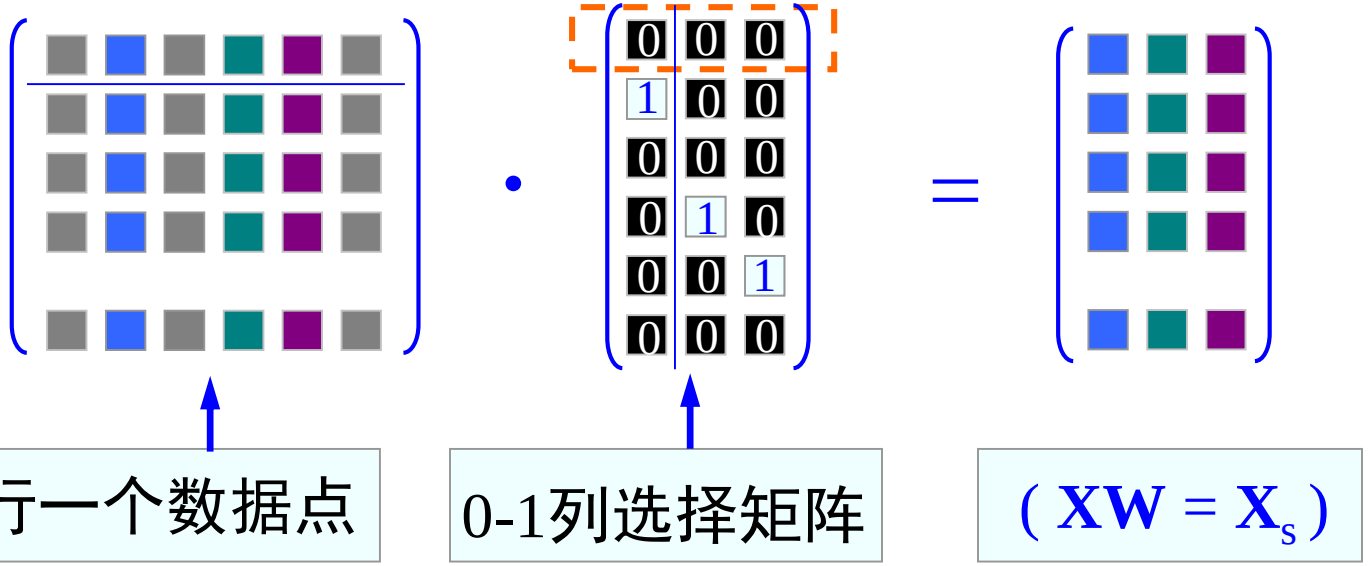
$$(\mathrm{p}_1) \quad \min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2, \quad \text{subject to } \|\mathbf{w}\|_1 < t$$

$$(\mathrm{p}_1) \quad \min_{\mathbf{w}} \|\mathbf{X}\mathbf{w} - \mathbf{y}\|_2^2 + \lambda \|\mathbf{w}\|_1$$

LASSO算法!

2.8 特征选择

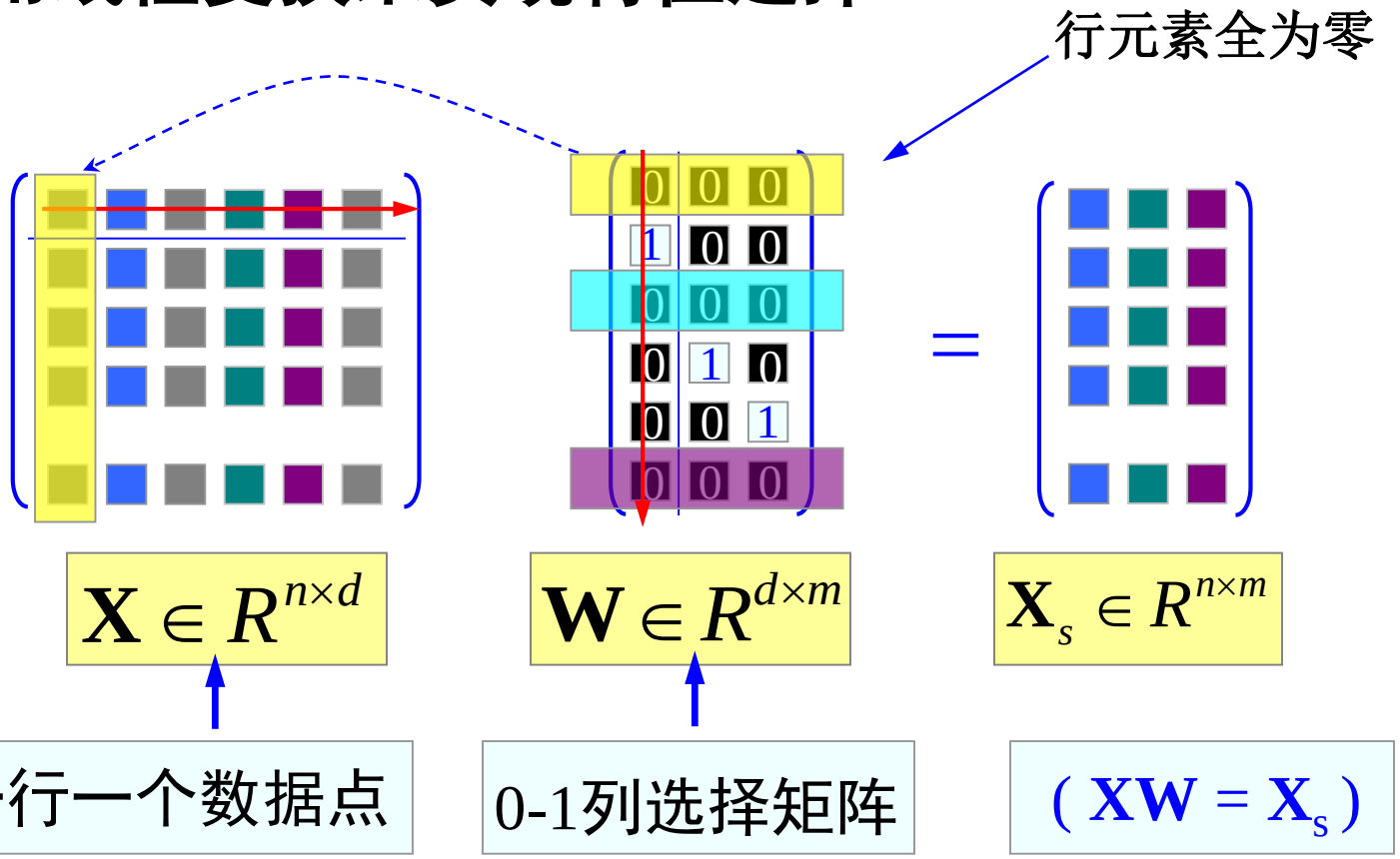
- 采用线性变换来实现特征选择



若列选择矩阵第一行全为零，则第一个特征分量不起作用！

2.8 特征选择

- 采用线性变换来实现特征选择

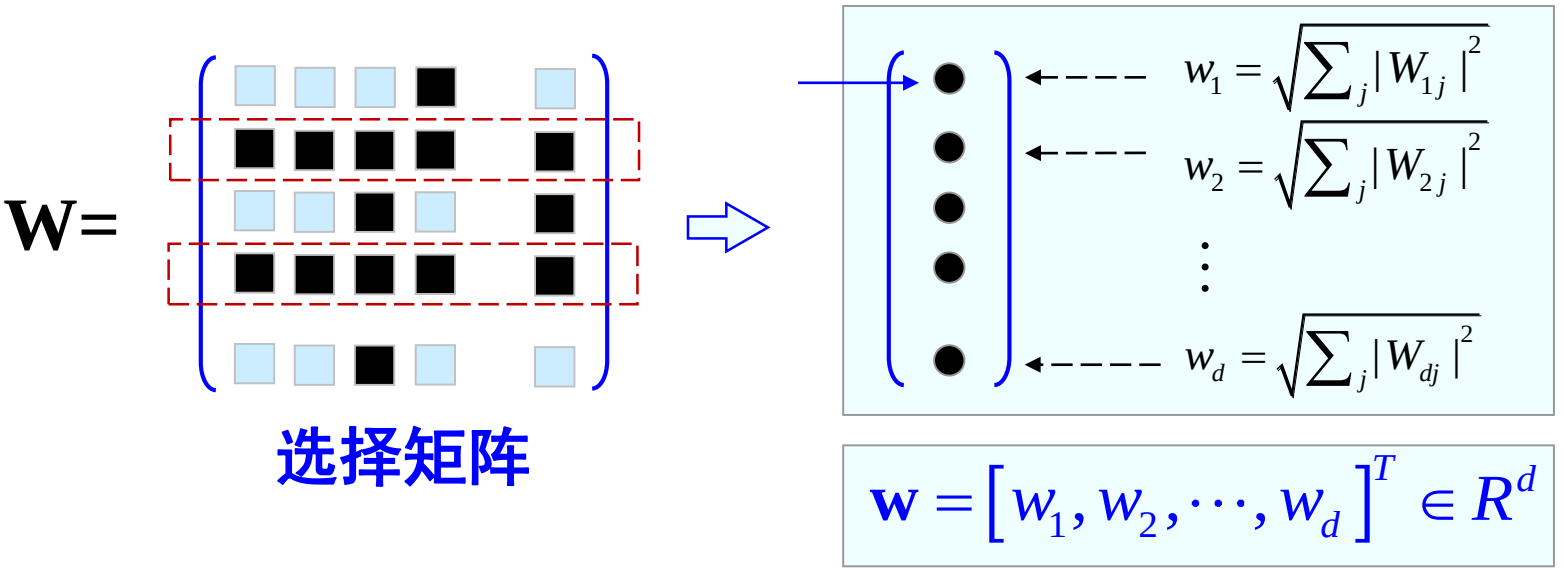


若列选择矩阵第一行全为零，则第一个特征分量不起作用！

2.8 特征选择

$$\mathbf{W}=\begin{pmatrix} W_{11} & \cdots & W_{1m} \\ \vdots & \ddots & \vdots \\ W_{d1} & \cdots & W_{1m} \end{pmatrix}$$

- 矩阵行稀疏性度量：结构化稀疏



要求W的某行为零，只需要该行元素的平方和为零。因此，可以将行平方和开根号收集为一个向量，再考虑其零范数

$\|\mathbf{w}\|_0$ is NP hard！ So we soft it as its L_1 norm $\|\mathbf{w}\|_1$, $\Rightarrow \|\mathbf{W}\|_{2,1}$

2.8 特征选择

$$\mathbf{W}=\begin{pmatrix} W_{11} & \cdots & W_{1m} \\ \vdots & \ddots & \vdots \\ W_{d1} & \cdots & W_{1m} \end{pmatrix}$$

- **矩阵的$L_{2,1}$范数：**

$$\|\mathbf{W}\|_{2,1}=\|\mathbf{w}\|_1=\sum_{i=1}^{d}\sqrt{\sum\nolimits_j |W_{ij}|^2}$$

  - The $L_{2,1}$ norm of matrix is a true norm

$$d(\mathbf{X},\mathbf{Y})=\|\mathbf{X}-\mathbf{Y}\|_{2,1}$$

    - 自反性、非负性、对称性和三角不等式关系

2.8 特征选择

- **正则化线性回归**
  - 线性变换：$\mathbf{y} = \mathbf{W}^T\mathbf{x} + \mathbf{b}$, where $\mathbf{y} \in R^m$, $\mathbf{W} \in R^{d\times m}$, $\mathbf{b} \in R^m$
  - 对n个样本$\{(\mathbf{x}_1, \mathbf{y}_1), (\mathbf{x}_2, \mathbf{y}_2), \ldots, (\mathbf{x}_n, \mathbf{y}_n)\}$，我们期望：

$$\mathbf{XW} - \mathbf{e}_n\mathbf{b}^T \approx \mathbf{Y}, \quad \text{where } \mathbf{X} \in R^{n\times m},\ \mathbf{Y} \in R^{n\times c},\ \mathbf{e}_n \in [1,\ldots,1] \in R^n$$

  - 模型：在最小化“正则化线性回归框架”下，有：

$$\min_{\mathbf{W},\mathbf{b}} \left\| \mathbf{XW} + \mathbf{e}_n\mathbf{b}^T - \mathbf{Y} \right\|_F^2 + \lambda \| \mathbf{W} \|_F^2$$

$$\text{where } \mathbf{X} = \left[\mathbf{x}_1, \cdots, \mathbf{x}_n\right]^T = \begin{pmatrix} x_{11} & \cdots & x_{1m} \\ \vdots & \ddots & \vdots \\ x_{11} & \cdots & x_{nm} \end{pmatrix} \in R^{n\times m}$$

（每一行为一个样本点）

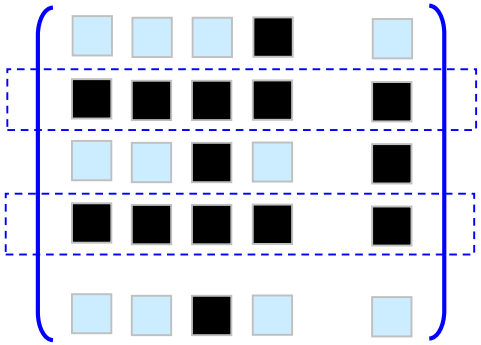
- **学习模型:**

$$\min_{\mathbf{W},\mathbf{b}} \left\| \mathbf{XW} + \mathbf{e}_n \mathbf{b}^T - \mathbf{Y} \right\|_F^2 + \lambda \| \mathbf{W} \|_F^2$$

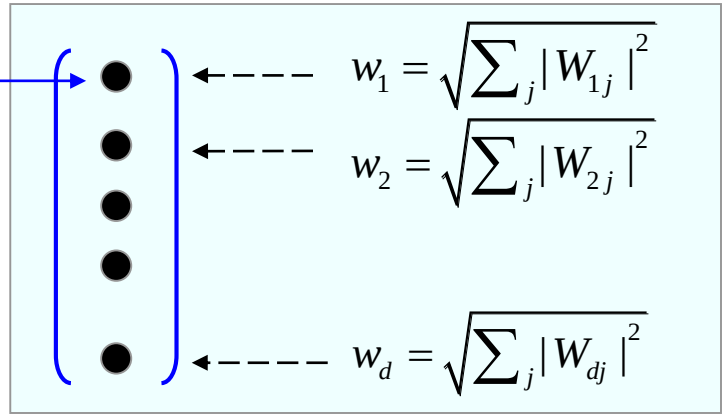
$$\min_{\mathbf{W},\mathbf{b}} \left\| \mathbf{XW} + \mathbf{e}_n \mathbf{b}^T - \mathbf{Y} \right\|_F^2 + \lambda \| \mathbf{W} \|_{2,1}$$

- **最后如何实现特征选择的目标？— 排序**

$\mathbf{W}=$



选择矩阵



$$\mathbf{w} = \left[w_1, w_2, \cdots, w_d \right]^T \in R^d$$

Shiming Xiang, et al., Discriminative Least Squares Regression for Multiclass Classification and Feature Selection. IEEE Transactions on Neural Network and Learning System,2012

2.9 多分类学习

- 引言
  - 现实应用中，我们通常需要处理多分类问题。一些二分类方法可以直接推广到多类分类问题。有此则难以从形式上一次性得到推广。
  - 一个基本策略是：利用二分类学习器的组合来解决多类分问题。其基本思路是“拆解法”：在分类器构造（训练阶段），即将多分类任务拆为多个二分类任务。在测试阶段，这这些分类**器的结果进行集成以获得最终的分类结果。**

2.9 多分类学习

- **拆分策略**
  - 一对一：
    - 假设有c个类别， 每个类别将两两配对，一共产生$c(c-1)/2$个分类器。在测试阶段，每个样本会得到$c(c-1)/2$个分类结果，最后通过“服从多数”原则，采用投票结果来确定其类别归属。
  - 一对多（一对其余）：
    - 对于所有n个训练样本，将其中一个类的样本视为正例，其余所有样本视为负类，训练一个分类器。依次轮换所有类，因此一共将构造c个分类器。对于新样本，采用投票规则。
  - 多对多：
    - 每次将若干个类作为正例，若干个其它类作为反例。
    - 最常用的多对多策略是“纠错输出码”（Error Correcting Output Codes, ECOC）

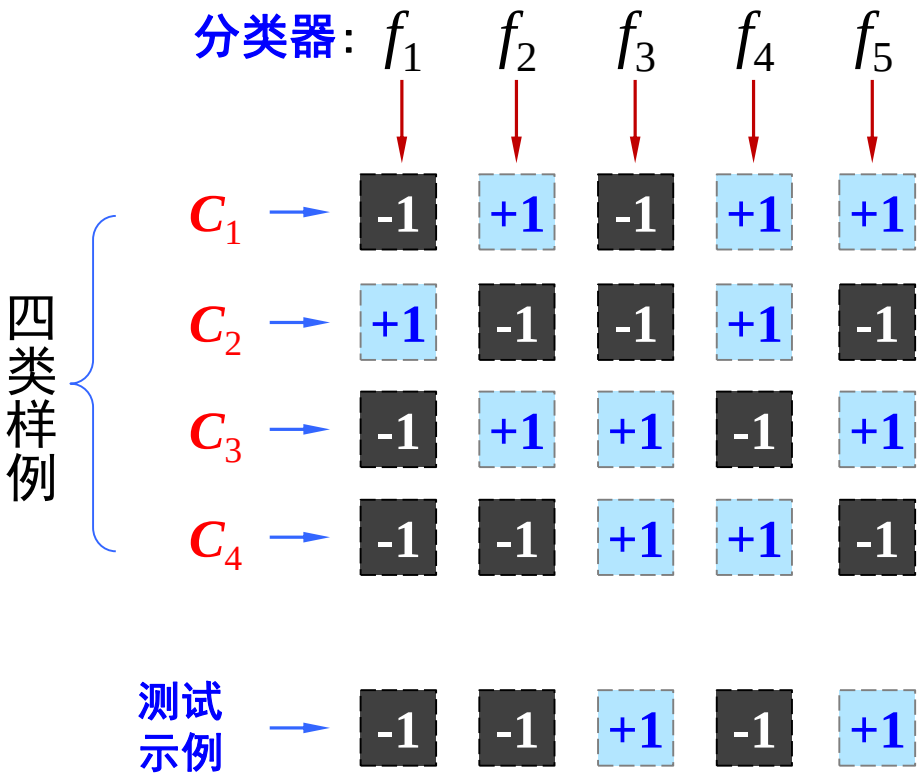
2.9 多分类学习

- **ECOC**
  - 将编码的思想引入至多类类别拆分，并尽可能地在解码过程具有容错性。
    - 编码：对c个类别做M次划分，每次将一部分类别划分为正类，一部分划分为负类，从而形成一个二分类集；按此方式，一共产生M个分类器。
    - 解码：采用M个分类器依次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。

2.9 多分类学习

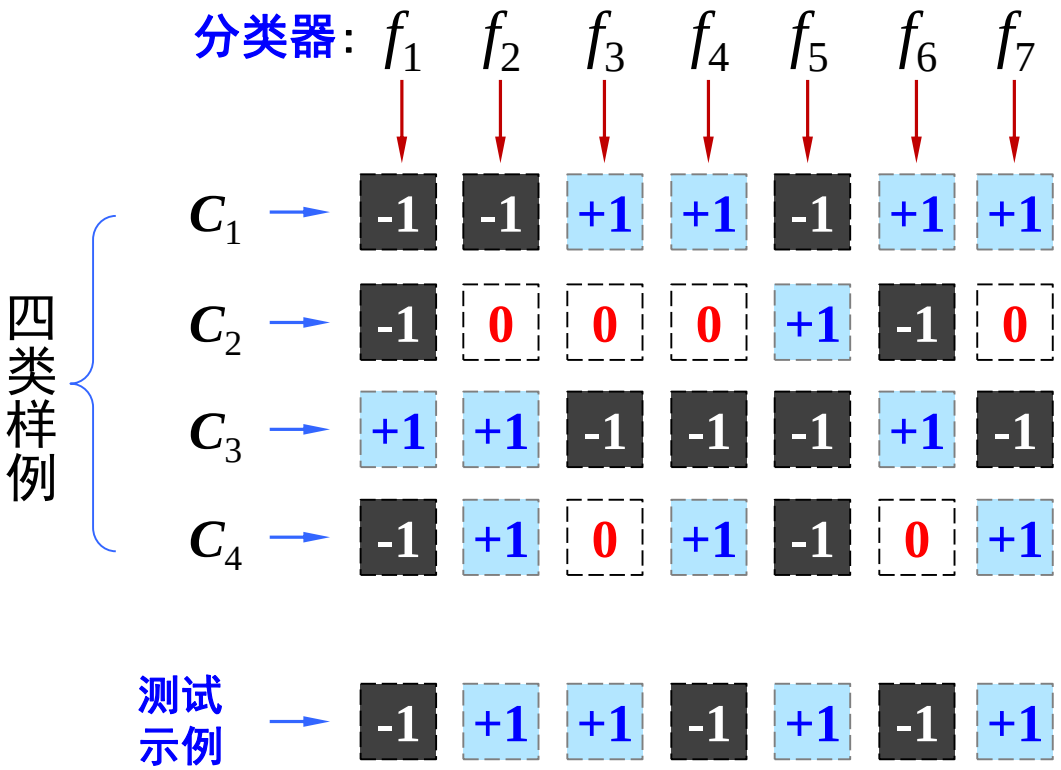
- **ECOC**
  - 类别划分通过编码矩阵（coding matrix）进行来指定。编码矩阵有多种形式。常见的有二元码、三元码。
    - 二元码：将每个类别，要么指定为正类，要么指定为负类；
    - 三元码：对于一个类别，既可以充当正类，也可以充当负类，还可以充当“停用类”。

- **ECOC二元码示意图：**



海明距离	欧氏距离
3	$2\sqrt{3}$
4	4
1	2
2	$2\sqrt{2}$

- **ECOC三元码示意图** (0:表示不考虑此类)：



海明距离	欧氏距离
4	4
4	2
5	$2\sqrt{5}$
3	$\sqrt{10}$

- ECOC为什么会纠错
  - 在测试阶段，ECOC编码对分类器的错误有一定的容忍和修正能力。
    - 假设在上述的二元码示例中，正确的预测编码应该为(-1,+1,+1,-1,+1)，即属于C_3类。但多分类器的实际输出为(-1,-1,+1,-1,+1)。最后仍被分C_3类。
  - 对同一个学习任务，ECOC编码越长，纠错能力越强。
  - 对同等长度的编码，理论上讲，任意两个类别之间的编码越远，则纠错能力越强。
  - 在编码长度较小时，可以据此设计最优编码。但对于长编码，最优编码设计是一个NP难问题。
  - 但并不是编码的理论性能越好，分类性能就越好，因为机器学习问题涉及很多因素。
  - 编码越长，则分类器越多，训练时间越长。

2.10 类不平衡问题

- 引言
  - 前面的分类学习方法都有一个共同的假设，即不同类别的训练样本数目相当。
  - 如果各类别数目相差很大，则会造成一些问题。比如：有990个正例，10个负例，则分类器很容易达到98%的分类精度。然而，它可能没有任何价值，因为它可能很难预测出负类。
  - 类不平衡问题(class-imbalance)是指分类任务中不同类别的样例数目差别很大。
    - 比如：一对多策略、ECOC策略

2.10 类不平衡问题

- **从线性分器的角度来理解**
  - 线性变换 $y = \mathbf{w}^T\mathbf{x}+b$,对新样本进行分类时，事实上是在用预测值$y$与一个阈值进行比较。
  - 例如y大于0.5则判定为正类，否则为反类。 y实际上表达了属于正类的可能性。几率$y/(1-y)$则反映了正例可能性与反例可能性之比。阈值设为0.5表明分类器假定真实正例与反例可能性相同，即分类器决策规则为：

$$\text{if } \frac{y}{1-y} > 1, \text{ then classify it into the positive class}$$

2.10 类不平衡问题

- **从线性分器的角度来理解**
  - 当训练集中正、反例的数目不同相时，令m^+表示正例数目， m^-表示反例数目，则观测几度为$m^+/\ m^-$。由于我们通常假定训练集是真实样本的总体无偏离采样，因此观测几率代表了真实几率。因此，只要分类器的预测几率高于真实几率，则可以判定该样本属于正例：

$$\text{if } \frac{y}{1-y} > \frac{m^+}{m^-}, \text{ then classify it into the positive class}$$

2.10 类不平衡问题

- **类不平衡学习的基本策略—再缩放**
  - 由于在实际决策时，总是采用$y/(1- y) >0.5$来作决策，所以为了实现上述策略，则可以对观测值进行缩放，使y变动到y':

$$\frac{y'}{1-y'} = \frac{y}{1-y}\frac{m^-}{m^+} \quad (>1, \text{then into the positive class})$$

  - 再缩放是代价敏感学习(cost-sensitive learning)的基础。比如：采用 $cost^+/cost^-$ 来代替 m^-/m^+。此处，$cost^+$ 代表将正例误分为反例的代价。
  - 但是，实际操作过程中，该策略也未必十分有效。主要是因为“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，我们未必能根据训练样估计出真实几率。

2.10 类不平衡问题

- **类不平衡学习的技术策略**
  - 第一类：对于样本数目较多的类，进行“欠采样(under-sampling)”，使两类数目基本相同。
  - 第二类：对于样本数目较少的类，进行“过采样(over-sampling)”，使两类数目基本相同。
  - 第三类：采用所有原始数据进行分类器构造，然后采用类似于“再缩放”策略进行阈值移动（threshold-moving）来决策。

Thank All of You!

(Questions?)

向世明

smxiang@nlpr.ia.ac.cn

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

时空数据分析与学习课题组(STDAL)

中科院自动化研究所· 模式识别国家重点实验室