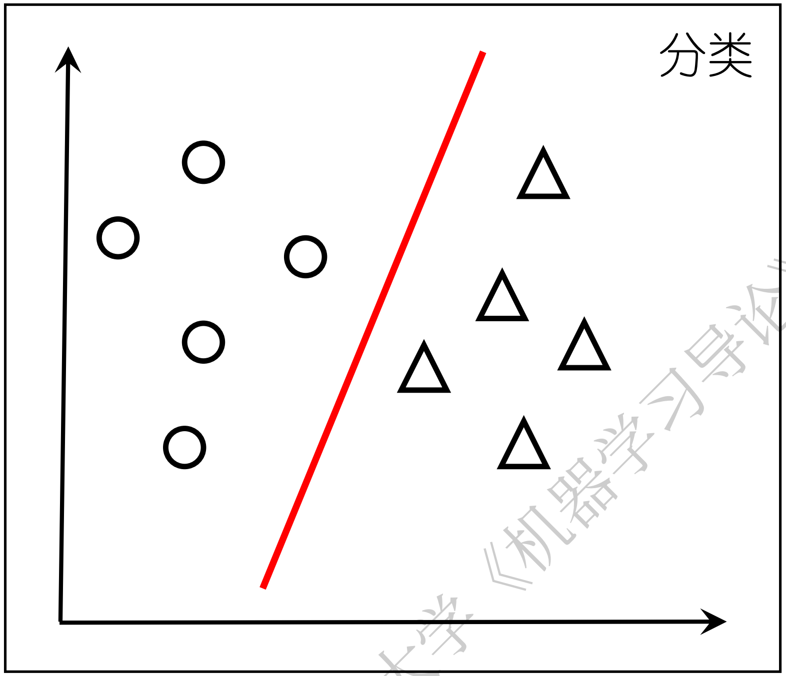
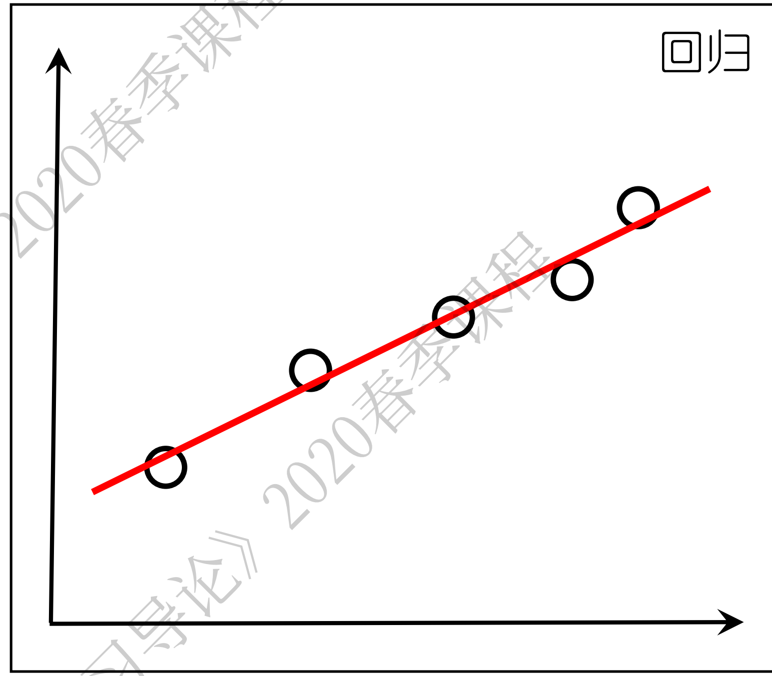
三、线性模型

主讲教师：周志华

线性模型





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

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

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

简单、基本、可理解性好

南京大学《机器学习导论》2020春季课程

线性回归 (linear regression)

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

离散属性的处理：若有“序”(order)，则连续化；
否则，转化为 k 维向量

令均方误差最小化，有
$$(w^*, b^*) = \underset{(w,b)}{\arg\min} \sum_{i=1}^{m} \left(f\left(x_i\right) - y_i\right)^2$$
$$= \underset{(w,b)}{\arg\min} \sum_{i=1}^{m} \left(y_i - wx_i - b\right)^2$$

对 $E_{(w,b)} = \sum_{i=1}^{m} \left(y_i - wx_i - b\right)^2$ 进行最小二乘参数估计

线性回归

分别对 w 和 b 求导：

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

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

令导数为 0，得到闭式(closed-form)解：

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

多元(multi-variate)线性回归

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

$$\boldsymbol{x}_i = (x_{i1}; x_{i2}; \ldots; x_{id}) \qquad y_i \in \mathbb{R}$$

把 $\boldsymbol{w}$ 和 b 吸收入向量形式 $\hat{\boldsymbol{w}} = (\boldsymbol{w}; b)$，数据集表示为

$$\mathbf{X} = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} & 1 \\ x_{21} & x_{22} & \cdots & x_{2d} & 1 \\ \vdots & \vdots & \ddots & \vdots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{md} & 1 \end{pmatrix} = \begin{pmatrix} \boldsymbol{x}_1^{\mathrm{T}} & 1 \\ \boldsymbol{x}_2^{\mathrm{T}} & 1 \\ \vdots & \vdots \\ \boldsymbol{x}_m^{\mathrm{T}} & 1 \end{pmatrix} \qquad \boldsymbol{y} = (y_1; y_2; \ldots; y_m)$$

多元线性回归

同样采用最小二乘法求解，有

$$\hat{\boldsymbol{w}}^* = \underset{\hat{\boldsymbol{w}}}{\arg\min} \left(\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}}\right)^{\mathrm{T}} \left(\boldsymbol{y} - \mathbf{X}\hat{\boldsymbol{w}}\right)$$

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

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

令其为零可得 $\hat{\boldsymbol{w}}$

然而，麻烦来了：涉及矩阵求逆！

- 若 $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ 满秩或正定，则 $\hat{\boldsymbol{w}}^* = \left(\mathbf{X}^{\mathrm{T}}\mathbf{X}\right)^{-1}\mathbf{X}^{\mathrm{T}}\boldsymbol{y}$
- 若 $\mathbf{X}^{\mathrm{T}}\mathbf{X}$ 不满秩，则可解出多个 $\hat{\boldsymbol{w}}$

此时需求助于归纳偏好，或引入 正则化 (regularization) ⟶ 第6、11章

线性模型的变化

对于样例 $(\boldsymbol{x}, y)$, $y \in \mathbb{R}$, 若希望线性模型的预测值逼近真实标记，

则得到线性回归模型 $y = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

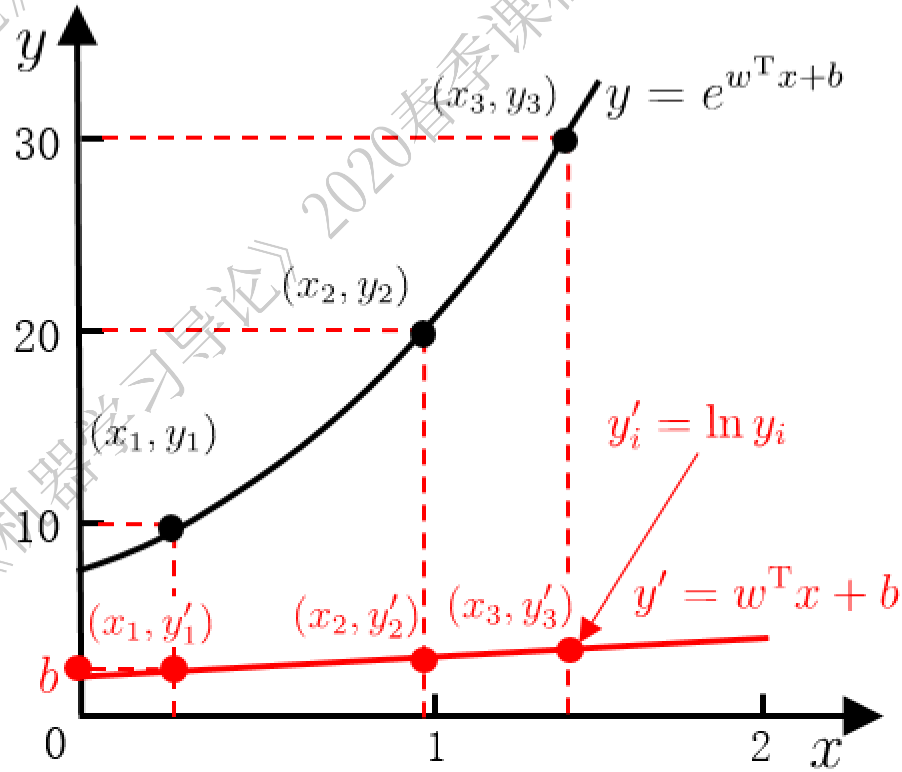
令预测值逼近 y 的衍生物？

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

则得到对数线性回归

(log-linear regression)

实际是在用 $e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}$ 逼近 y



广义(generalized)线性模型

一般形式： $y = \underbrace{g^{-1}}_{\text{单调可微的 联系函数 (link function)}} \left(\boldsymbol{w}^{\mathrm{T}} \boldsymbol{x} + b \right)$

单调可微的 联系函数 (link function)

令 $g(\cdot) = \ln(\cdot)$ 则得到 对数线性回归

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

…… ……

二分类任务

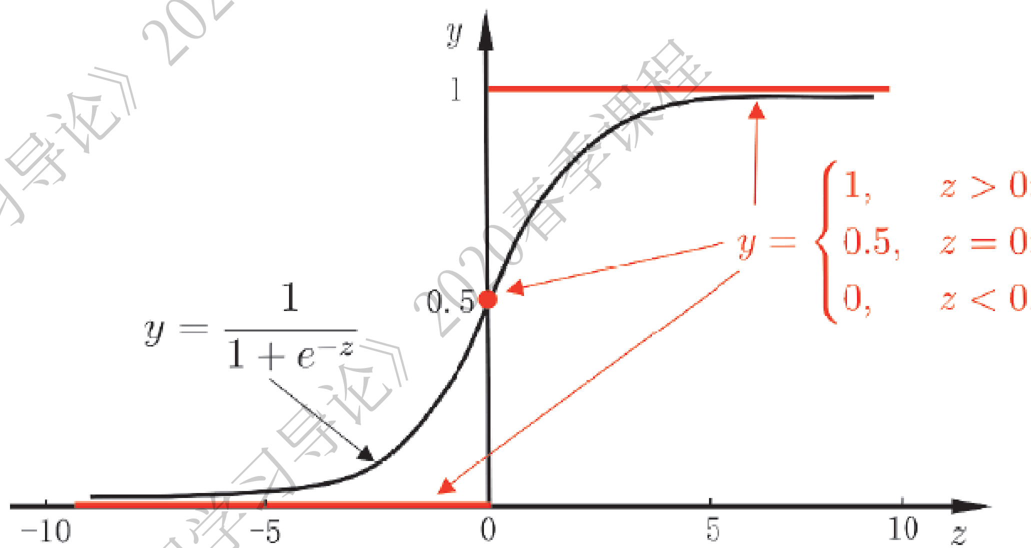
线性回归模型产生的实值输出 $z = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$

期望输出 $y \in \{0, 1\}$

找 z 和 y 的联系函数

理想的“单位阶跃函数”
(unit-step function)

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



性质不好，
需找“替代函数”
(surrogate function)

常用
单调可微、任意阶可导

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

对数几率函数
(logistic function)
简称“对率函数”

注意：Logistic与“逻辑”没有半毛钱关系！
1. Logistic 源自 Logit，不是Logic； 2. 实数值，并非“非0即1”的逻辑值

对率回归

以对率函数为联系函数：

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

即：

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

几率(odds)，反映了 $\boldsymbol{x}$ 作为正例的相对可能性

"对数几率"
(log odds，亦称 logit)

"对数几率回归" (logistic regression)
简称 "对率回归"

- 无需事先假设数据分布
- 可得到 "类别" 的近似概率预测
- 可直接应用现有数值优化算法求取最优解

注意：它是
分类学习算法！

求解思路

若将 y 看作类后验概率估计 $p(y=1 \mid \boldsymbol{x})$，则

$$\ln \frac{y}{1-y} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b \quad \text{可写为} \quad \ln \frac{p(y=1 \mid \boldsymbol{x})}{p(y=0 \mid \boldsymbol{x})} = \boldsymbol{w}^{\mathrm{T}}\boldsymbol{x} + b$$

于是，可使用“极大似然法” ⟶ 第7章
(maximum likelihood method)

给定数据集 $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$

最大化“对数似然”(log-likelihood)函数

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

求解思路

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 $p_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})=p(y=1\mid\hat{\boldsymbol{x}};\boldsymbol{\beta})=\dfrac{e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}{1+e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}$

$p_0(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})=p(y=0\mid\hat{\boldsymbol{x}};\boldsymbol{\beta})=1-p_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})=\dfrac{1}{1+e^{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{x}+b}}$

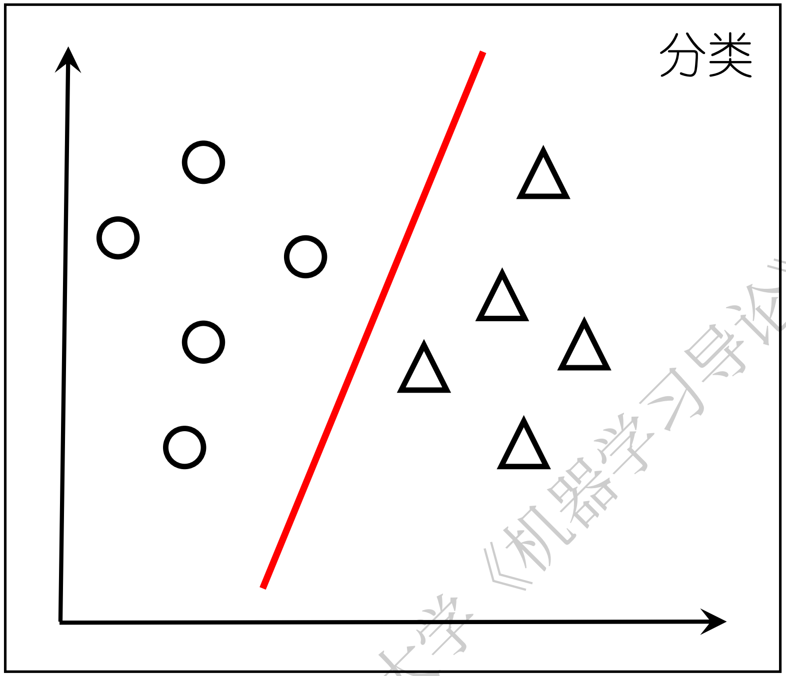
则似然项可重写为 $p(y_i\mid\boldsymbol{x}_i;\boldsymbol{w}_i,b)=y_ip_1(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})+(1-y_i)p_0(\hat{\boldsymbol{x}}_i;\boldsymbol{\beta})$

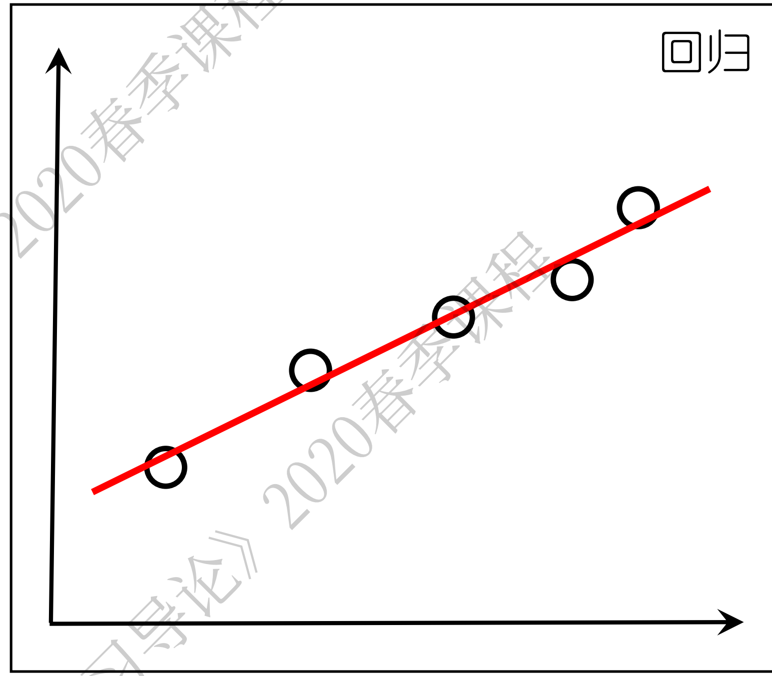
于是，最大化似然函数 $\ell(\boldsymbol{w},b)=\sum_{i=1}^{m}\ln p(y_i\mid\boldsymbol{x}_i;\boldsymbol{w},b)$

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

高阶可导连续凸函数，可用经典的数值优化方法
如梯度下降法/牛顿法 [Boyd and Vandenberghe, 2004]

线性模型做 “分类”



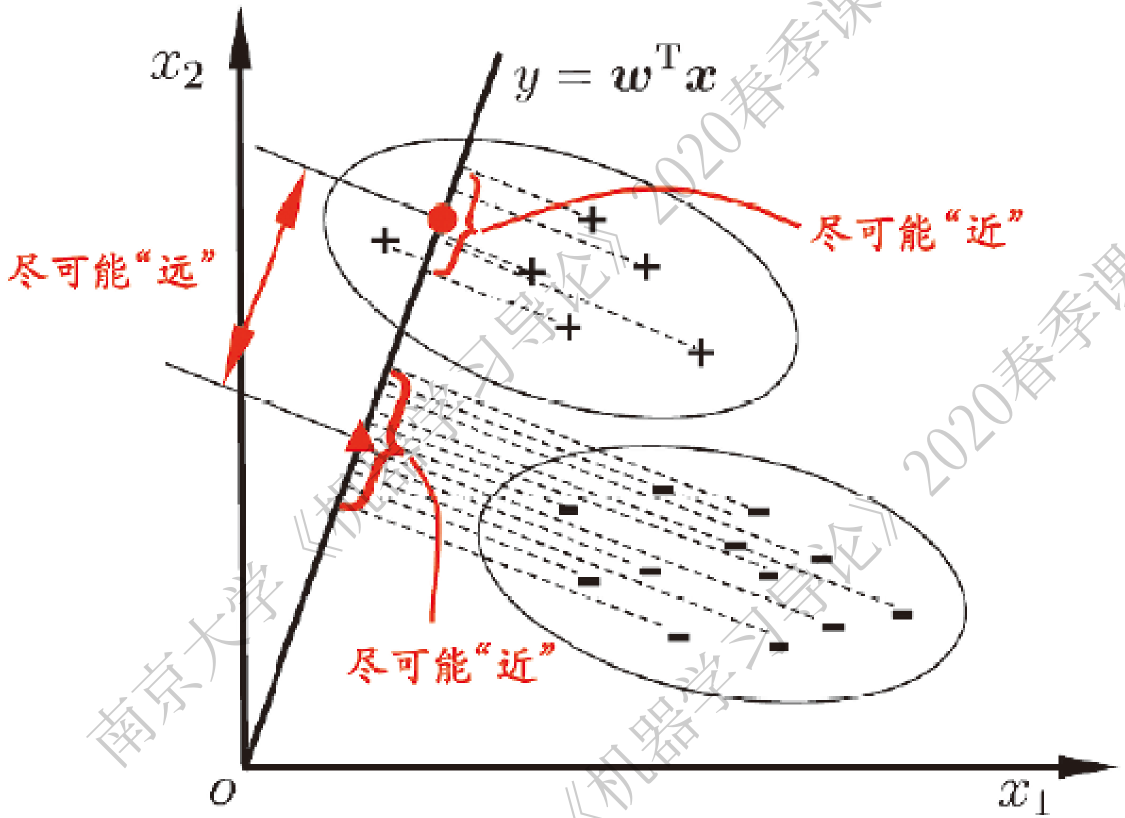


如何 “直接” 做分类？

广义线性模型；
通过 “联系函数”

例如，对率回归

线性判别分析 (Linear Discriminant Analysis)



由于将样例投影到一条直线（低维空间），因此也被视为一种"监督降维"技术　降维 ⟶ 第10章

LDA的目标

给定数据集 $\{(\boldsymbol{x}_i, y_i)\}_{i=1}^{m}$

第 i 类示例的集合 X_i

第 i 类示例的均值向量 $\boldsymbol{\mu}_i$

第 i 类示例的协方差矩阵 $\boldsymbol{\Sigma}_i$

两类样本的中心在直线上的投影：$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_0$ 和 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_1$

两类样本的协方差：$\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_0\boldsymbol{w}$ 和 $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_1\boldsymbol{w}$

同类样例的投影点尽可能接近 → $\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_0\boldsymbol{w} + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_1\boldsymbol{w}$ 尽可能小

异类样例的投影点尽可能远离 → $\|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_0 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_1\|_2^2$ 尽可能大

于是，最大化

$$J = \frac{\|\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_0 - \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\mu}_1\|_2^2}{\boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_0\boldsymbol{w} + \boldsymbol{w}^{\mathrm{T}}\boldsymbol{\Sigma}_1\boldsymbol{w}} = \frac{\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)(\boldsymbol{\mu}_0 - \boldsymbol{\mu}_1)^{\mathrm{T}}\boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}}(\boldsymbol{\Sigma}_0 + \boldsymbol{\Sigma}_1)\boldsymbol{w}}$$

LDA的目标

类内散度矩阵 (within-class scatter matrix)

$$\mathbf{S}_w = \mathbf{\Sigma}_0 + \mathbf{\Sigma}_1$$
$$= \sum_{x \in X_0} (\boldsymbol{x} - \boldsymbol{\mu}_0)(\boldsymbol{x} - \boldsymbol{\mu}_0)^{\mathrm{T}} + \sum_{x \in X_1} (\boldsymbol{x} - \boldsymbol{\mu}_1)(\boldsymbol{x} - \boldsymbol{\mu}_1)^{\mathrm{T}}$$

类间散度矩阵 (between-class scatter matrix)

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

LDA的目标：最大化广义瑞利商 (generalized Rayleigh quotient)

$$\boxed{J = \frac{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_b \boldsymbol{w}}{\boldsymbol{w}^{\mathrm{T}} \mathbf{S}_w \boldsymbol{w}}}$$

$\boldsymbol{w}$ 成倍缩放不影响 J 值
仅考虑方向

求解思路

令 $\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w}=1$，最大化广义瑞利商等价形式为

$$\min_{\boldsymbol{w}} \; -\boldsymbol{w}^{\mathrm{T}}\mathbf{S}_b\boldsymbol{w}$$

$$\text{s.t. } \boldsymbol{w}^{\mathrm{T}}\mathbf{S}_w\boldsymbol{w}=1$$

运用拉格朗日乘子法，有 $\mathbf{S}_b\boldsymbol{w}=\lambda\mathbf{S}_w\boldsymbol{w}$

$\mathbf{S}_b\boldsymbol{w}$ 的方向恒为 $\boldsymbol{\mu}_0-\boldsymbol{\mu}_1$，不妨令 $\mathbf{S}_b\boldsymbol{w}=\lambda(\boldsymbol{\mu}_0-\boldsymbol{\mu}_1)$

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

实践中通常是进行奇异值分解 $\mathbf{S}_w=\mathbf{U}\mathbf{\Sigma}\mathbf{V}^{\mathrm{T}}$ → 附录A

然后 $\mathbf{S}_w^{-1}=\mathbf{V}\mathbf{\Sigma}^{-1}\mathbf{U}^{\mathrm{T}}$

推广到多类

假定有 N 个类

- 全局散度矩阵 $\mathbf{S}_t = \mathbf{S}_b + \mathbf{S}_w = \sum_{i=1}^{m} (\boldsymbol{x}_i - \boldsymbol{\mu})(\boldsymbol{x}_i - \boldsymbol{\mu})^T$
- 类内散度矩阵 $\mathbf{S}_w = \sum_{i=1}^{N} \mathbf{S}_{w_i}$ $\quad \mathbf{S}_{w_i} = \sum_{\boldsymbol{x} \in X_i} (\boldsymbol{x} - \boldsymbol{\mu}_i)(\boldsymbol{x} - \boldsymbol{\mu}_i)^T$
- 类间散度矩阵 $\mathbf{S}_b = \mathbf{S}_t - \mathbf{S}_w = \sum_{i=1}^{N} m_i (\boldsymbol{\mu}_i - \boldsymbol{\mu})(\boldsymbol{\mu}_i - \boldsymbol{\mu})^T$

多分类LDA有多种实现方法：采用 $\mathbf{S}_b$，$\mathbf{S}_w$，$\mathbf{S}_t$ 中的任何两个

例如，$\max\limits_{\mathbf{W}} \frac{\operatorname{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{S}_b\mathbf{W})}{\operatorname{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{S}_w\mathbf{W})} \Longrightarrow \mathbf{S}_b\mathbf{W} = \lambda \mathbf{S}_w \mathbf{W}$

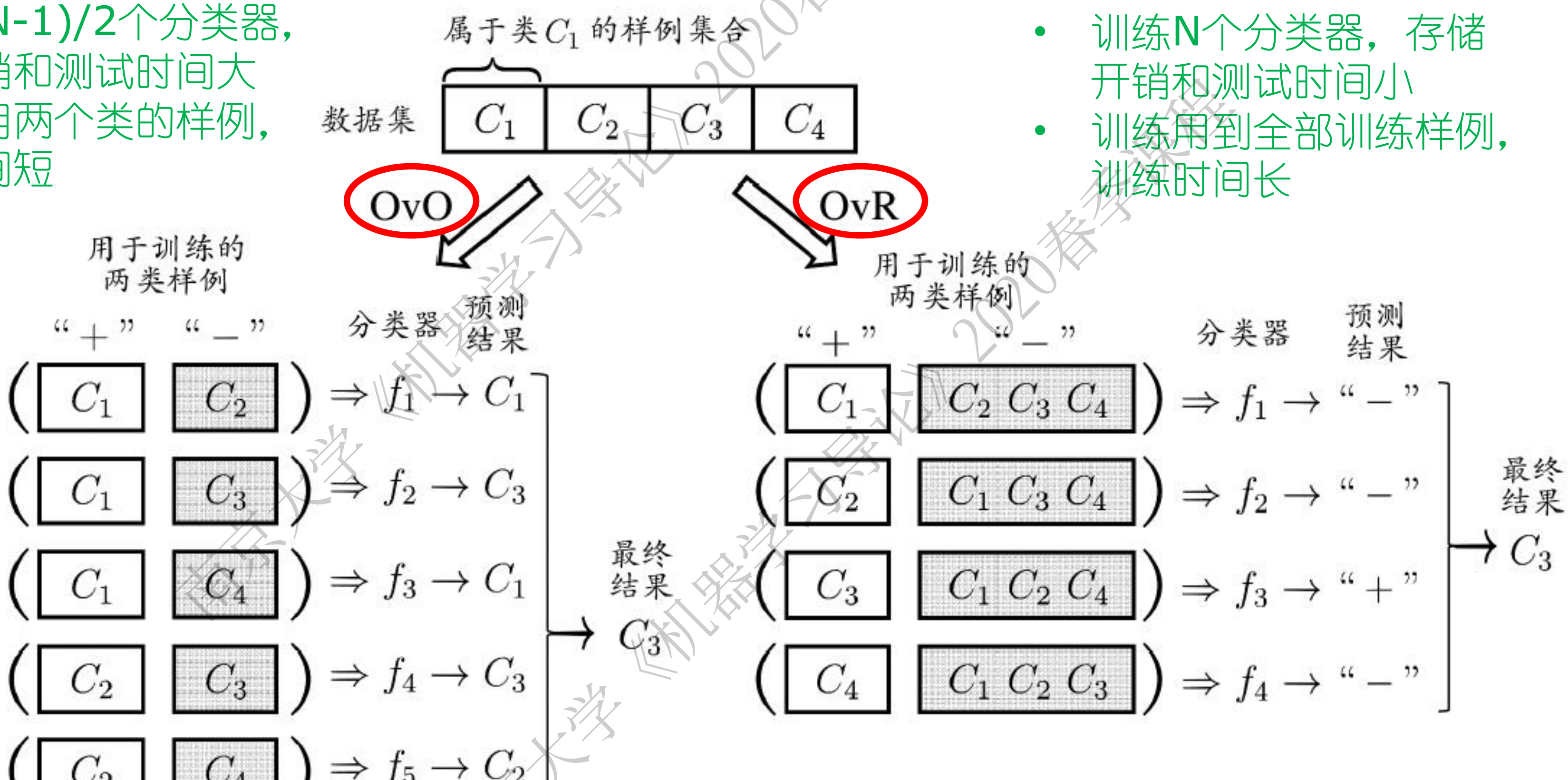
$\mathbf{W} \in \mathbb{R}^{d \times (N-1)}$

$\mathbf{W}$的闭式解是 $\mathbf{S}_w^{-1}\mathbf{S}_b$的 d' ($\leq N$-1) 个最大非零广义特征值对应的特征向量组成的矩阵

多分类学习

拆解法：将一个多分类任务拆分为若干个二分类任务求解

- 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



- 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

类别不平衡 (class-imbalance)

不同类别的样本比例相差很大；“小类”往往更重要

基本思路：

若 $\dfrac{y}{1-y} > 1$ 则 预测为正例.

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

基本策略

—— “再缩放” (rescaling)：

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

然而，精确估计 m^-/m^+ 通常很困难！

常见类别不平衡学习方法：

- 过采样 (oversampling)
  例如：SMOTE
- 欠采样 (undersampling)
  例如：EasyEnsemble
- 阈值移动 (threshold-moving)

纠错输出码 (ECOC)

多对多(Many vs Many, MvM)：将若干类作为正类，若干类作为反类

一种常见方法：纠错输出码 (Error Correcting Output Code)

编码：对 N 个类别做 M 次划分，每次将一部分类别划为正类，一部分划为反类

M 个二类任务；
(原)每类对应一个长为 M 的编码

距离最小的类为最终结果

解码：测试样本交给 M 个分类器预测

长为 M 的预测结果编码

纠错输出码

	f_1	f_2	f_3	f_4	f_5	海明距离	欧氏距离
C_1	−1	+1	−1	+1	+1	3	$2\sqrt{3}$
C_2	+1	−1	−1	+1	−1	4	4
C_3	−1	+1	+1	−1	+1	1	2
C_4	−1	−1	+1	+1	−1	2	$2\sqrt{2}$
测试示例	−1	−1	+1	−1	+1		

(a) 二元 ECOC 码

[Dietterich and Bakiri,1995]

	f_1	f_2	f_3	f_4	f_5	f_6	f_7	海明距离	欧氏距离
C_1	−1	−1	+1	+1	−1	+1	+1	4	4
C_2	−1	0	0	0	+1	−1	0	2	2
C_3	+1	+1	−1	−1	−1	+1	−1	5	$2\sqrt{5}$
C_4	−1	+1	0	+1	−1	0	+1	3	$\sqrt{10}$
测试示例	−1	+1	+1	−1	+1	−1	+1		

(b) 三元 ECOC 码

[Allwein et al. 2000]

- ECOC编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

前往第四站……

主成分分析 (Principal Component Analysis, PCA)

正交属性空间中的样本点，如何使用一个超平面对所有样本进行恰当的表达？

若存在这样的超平面，那么它大概应具有这样的性质：

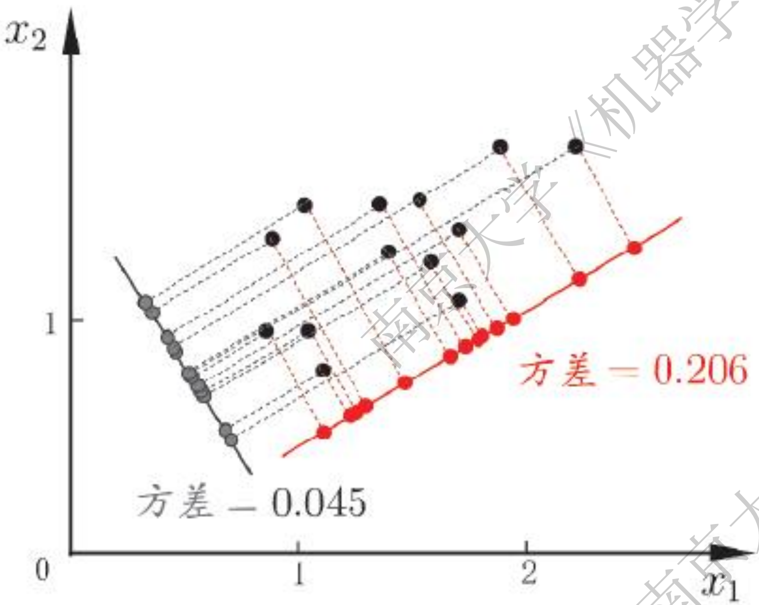
- 最近重构性：样本点到这个超平面的距离都足够近
- 最大可分性：样本点在这个超平面上的投影能尽可能分开

主成分分析的两种等价推导

PCA - 最大可分性

样本点 $\boldsymbol{x}_i$ 在新空间中超平面上的投影是 $\mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i$ ，若所有样本点的投影能尽可能分开，则应该使得投影后样本点的方差最大化

投影后样本点的方差是 $\sum_i \mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\mathbf{W}$



于是：
$$\max_{\mathbf{W}} \quad \mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W})$$
$$\text{s.t.} \quad \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}.$$

等价于：
$$\min_{\mathbf{W}} \quad -\mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W})$$
$$\text{s.t.} \quad \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}.$$

PCA 求解

$$\begin{aligned} \max_{\mathbf{W}} \quad & \mathrm{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W}) \\ \text{s.t.} \quad & \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}. \end{aligned}$$

使用拉格朗日乘子法可得

$$\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W} = \lambda\mathbf{W}.$$

只需对协方差矩阵 $\mathbf{X}\mathbf{X}^{\mathrm{T}}$ 进行特征值分解，并将求得的特征值排序：$\lambda_1 \geq \lambda_2 \geq \cdots \geq \lambda_d$，再取前 d' 个特征值对应的特征向量构成 $\mathbf{W} = (\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_{d'})$，这就是主成分分析的解

关键变量：子空间方差

PCA - 最近重构性

对样本进行中心化：$\sum_i \boldsymbol{x}_i = \mathbf{0}$

假定投影变换后得到的新坐标系为$\{\boldsymbol{w}_1, \boldsymbol{w}_2, \ldots, \boldsymbol{w}_d\}$，其中$\boldsymbol{w}_i$是标准正交基向量

$$||\boldsymbol{w}_i||_2 = 1, \boldsymbol{w}_i^{\mathrm{T}} \boldsymbol{w}_j = 0 (i \neq j).$$

若丢弃新坐标系中的部分坐标，即将维度降低到 $d' < d$，则样本点在低维坐标系中的投影是 $\boldsymbol{z}_i = (z_{i1}; z_{i2}; \ldots; z_{id'})$ $z_{ij} = \boldsymbol{w}_j^{\mathrm{T}} \boldsymbol{x}_i$

若基于$\boldsymbol{z}_i$来重构$\boldsymbol{x}_i$，则会得到 $\hat{\boldsymbol{x}}_i = \sum_{j=1}^{d'} z_{ij} \boldsymbol{w}_j.$

PCA - 最近重构性（续）

原样本点 $\boldsymbol{x}_i$ 与基于投影重构的样本点 $\hat{\boldsymbol{x}}_i$ 之间的距离为

$$\sum_{i=1}^{m}\left\|\sum_{j=1}^{d'} z_{ij}\boldsymbol{w}_j - \boldsymbol{x}_i\right\|_2^2 = \sum_{i=1}^{m}\boldsymbol{z}_i^{\mathrm{T}}\boldsymbol{z}_i - 2\sum_{i=1}^{m}\boldsymbol{z}_i^{\mathrm{T}}\mathbf{W}^{\mathrm{T}}\boldsymbol{x}_i + \text{const}$$
$$\propto -\operatorname{tr}\left(\mathbf{W}^{\mathrm{T}}\left(\sum_{i=1}^{m}\boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}\right)\mathbf{W}\right).$$

$\boldsymbol{w}_j$ 是正交基，$\sum_i \boldsymbol{x}_i\boldsymbol{x}_i^{\mathrm{T}}$ 是协方差矩阵，于是由最近重构性，有：

$$\min_{\mathbf{W}} \quad -\operatorname{tr}(\mathbf{W}^{\mathrm{T}}\mathbf{X}\mathbf{X}^{\mathrm{T}}\mathbf{W})$$
$$\text{s.t.} \quad \mathbf{W}^{\mathrm{T}}\mathbf{W} = \mathbf{I}.$$

关键变量：重构误差

这就是主成分分析的优化目标

PCA 应用

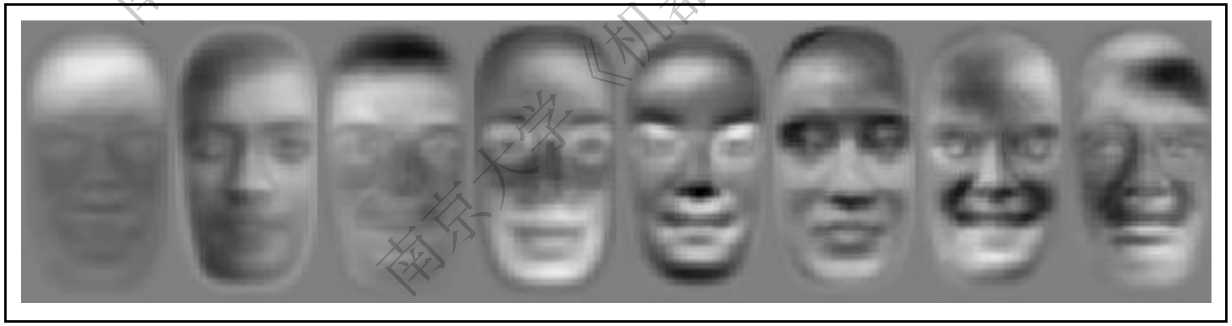
d' 的设置：

- 用户指定
- 在低维空间中对k近邻或其他分类器进行交叉验证
- 设置重构阈值，例如 t =95%，然后选取最小的 d' 使得 $\frac{\sum_{i=1}^{d'} \lambda_i}{\sum_{i=1}^{d} \lambda_i} \geq t.$

PCA 是最常用的降维方法，在不同领域有不同的称谓

例如在人脸识别中该技术被称为“特征脸” (eigenface)

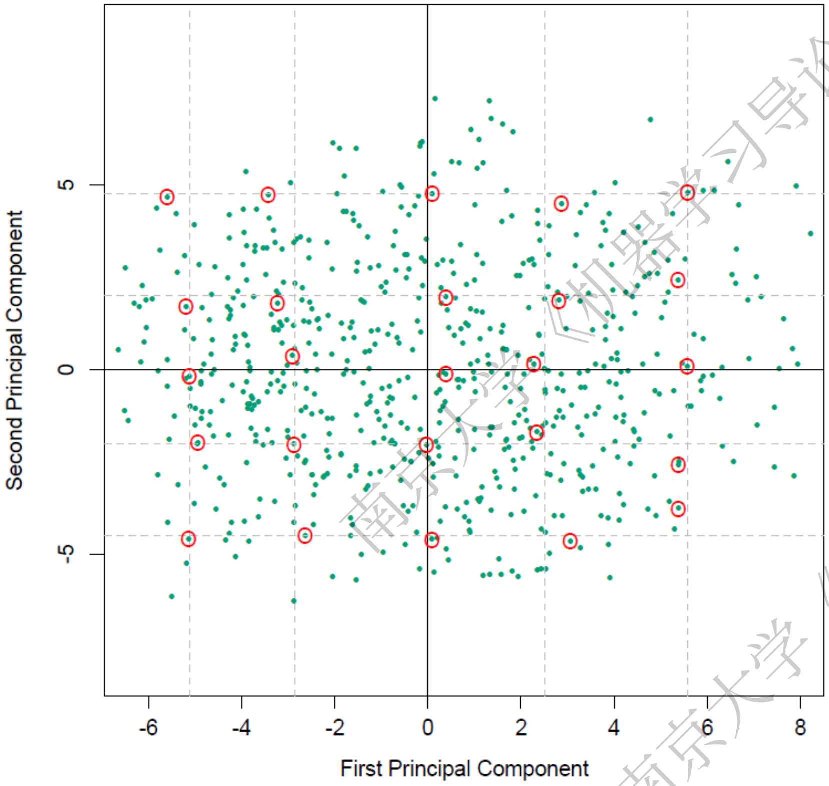
因为若将前 d' 个特征值对应的特征向量还原为图像，则得到

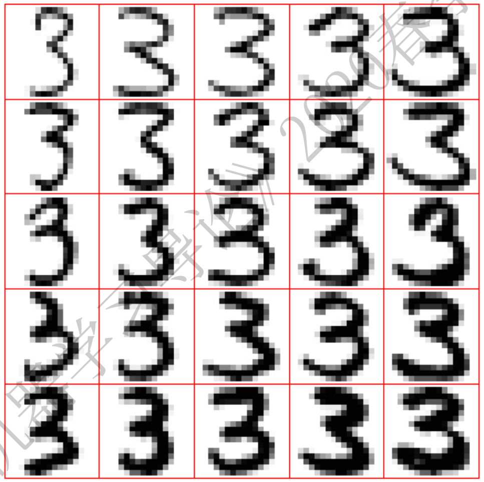


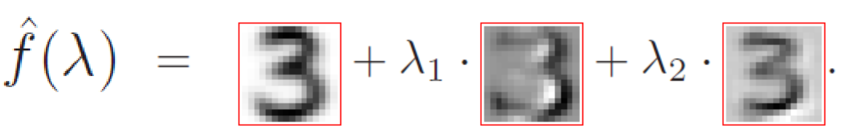
PCA 应用（续）

PCA 是最常用的降维方法，在不同领域有不同的称谓

例如在人脸识别中该技术被称为“特征脸” (eigenface)





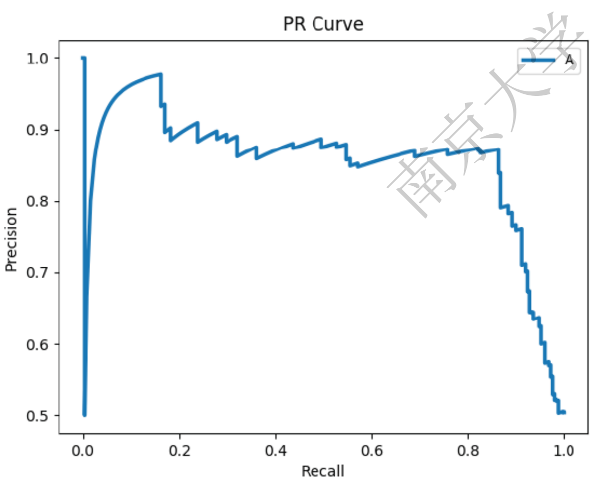


习题2：P-R曲线、ROC曲线

现有 500 个测试样例，其对应的真实标记和学习器的输出值如表1所示 (完整数据见 data.csv 文件)。该任务是一个二分类任务，1 表示正例，0 表示负例。学习器的输出越接近 1 表明学习器认为该样例越可能是正例，越接近 0 表明学习器认为该样例越可能是负例。

表 1: 测试样例表

样本	x_1	x_2	x_3	x_4	x_5	...	x_{496}	x_{497}	x_{498}	x_{499}	x_{500}
标记	1	1	0	0	0	...	0	1	0	1	1
输出值	0.206	0.662	0.219	0.126	0.450	...	0.184	0.505	0.445	0.994	0.602



(a) P-R curve

```
for i, y in enumerate(reversed(label)):
    if y == 1:
        TP += 1.
        FN -= 1.
    else:
        FP += 1.
        TN -= 1.
    if i < len(output) - 1 and output[i+1] == output[i]:
        continue
    precision = TP/(TP + FP)
    recall = TP/(TP+FN)
    TPR = TP/(TP+FN)
    FPR = FP/(TN+FP)
    PR_list.append((recall, precision))
    ROC_list.append((FPR, TPR))
```

习题4：假设检验

在数据集 D_1, D_2, D_3, D_4, D_5 运行了 A, B, C, D, E 五种算法，算法比较序值表如表2所示:

使用 Friedman 检验 ($\alpha = 0.05$) 判断这些算法是否性能都相同。若不相同，进行 Nemenyi 后续检验 ($\alpha = 0.05$)，并说明性能最好的算法与哪些算法有显著差别。本题需编程实现 Friedman 检验和 Nemenyi 后续检验。(预计代码行数小于 50 行)

$k = 5, N = 5$。由 42 页式 (2.34) 与 (2.35) 可得:

```
def f(df: pd.DataFrame):
    k = len(df.columns)
    N = len(df.index)
    S = 0
    for i in range(k):
        performance = df.iloc[:, i].values
        mean_performance = np.mean(performance)
        S += mean_performance ** 2
    forigin = 12 * N / (k * (k+1)) * (S - k * ((k+1)**2)/4)
    F = (N-1) * forigin / (N * (k-1) - forigin)
    return F
```

图 3: Friedman 检验核心代码

$\tau_{\chi^2}2 = 9.92, \tau_F = 3.9365 > 3.007$。因此拒绝 “所有算法性能相同” 假设。

$CD = 2.728, 1.2 + 2.728 = 3.928 < 4$。因此 C 与 D 有显著区别。C 与其余算法之间没有显著区别。

二项检验

- 测试样本：m个
- 泛化错误率：ϵ
- 测试错误率：$\hat{\epsilon}$
- 二项检验：$\epsilon \leq \epsilon_0$

$$\overline{\epsilon} = \min \epsilon \quad \text{s.t.} \quad \sum_{i=\epsilon \times m+1}^{m} \binom{m}{i} \epsilon_0^i (1-\epsilon_0)^{m-i} < \alpha$$

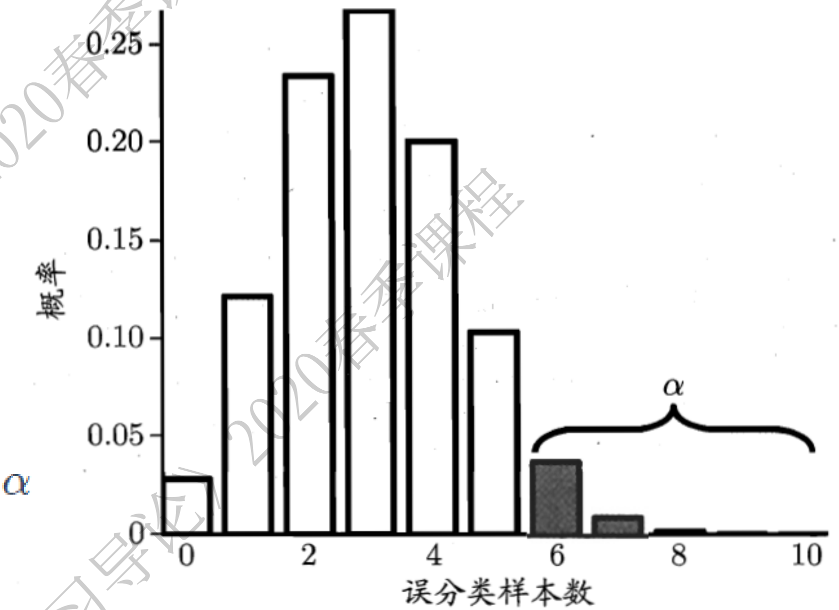


图 2.6 二项分布示意图($m = 10, \epsilon = 0.3$)

- $\hat{\epsilon} < \overline{\epsilon}$: 不能拒绝假设，

  以$1-\alpha$置信度认为$\epsilon \leq \epsilon_0$