

机器学习：无监督学习

教学课程组

2022年

- 参考教材：吴飞，《人工智能导论：模型与算法》，高等教育出版社
- 在线课程(MOOC)：<https://www.icourse163.org/course/ZJU-1003377027>
- 在线实训平台（智海-Mo）：https://mo.zju.edu.cn/classroom/class/zju_ai_2022
- 系列科普读物《走进人工智能》<https://www.ximalaya.com/album/56494803>

提纲

一、K均值聚类

二、主成分分析

三、特征人脸方法

四、潜在语义分析

五、期望最大化算法

K均值聚类 (K-means 聚类)

- 物以类聚，人以群分（《战国策·齐策三》）
- 输入： n 个数据（无任何标注信息）
- 输出： k 个聚类结果
- 目的：将 n 个数据聚类到 k 个集合（也称为类簇）

K均值聚类算法描述

- 若干定义：

- n 个 m -维数据 $\{x_1, x_2, \dots, x_n\}$, $x_i \in R^m (1 \leq i \leq n)$
- 两个 m 维数据之间的欧氏距离为

$$d(x_i, x_j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \dots + (x_{im} - x_{jm})^2}$$

$d(x_i, x_j)$ 值越小，表示 x_i 和 x_j 越相似；反之越不相似

- 聚类集合数目 k
- 问题：如何将 n 个数据依据其相似度大小将它们分别聚类到 k 个集合，使得每个数据仅属于一个聚类集合。

K均值聚类算法：初始化

■ 第一步：初始化聚类质心

- 初始化 k 个聚类质心 $c = \{c_1, c_2, \dots, c_k\}$, $c_j \in R^m (1 \leq j \leq k)$
- 每个聚类质心 c_j 所在集合记为 G_j

K均值聚类算法：对数据进行聚类

■ 第二步：将每个待聚类数据放入唯一一个聚类集合中

- 计算待聚类数据 x_i 和质心 c_j 之间的欧氏距离 $d(x_i, c_j)$ ($1 \leq i \leq n, 1 \leq j \leq k$)
- 将每个 x_i 放入与之距离最近聚类质心所在聚类集合中，

即 $\operatorname{argmin}_{c_j \in C} d(x_i, c_j)$

K均值聚类算法：更新聚类质心

■ 第三步：根据聚类结果、更新聚类质心

- 根据每个聚类集合中所包含的数据，更新该聚类集合质心

值，即：
$$c_j = \frac{1}{|G_j|} \sum_{x_i \in G_j} x_i$$

K均值聚类算法：继续迭代

■ 第四步：算法循环迭代，直到满足条件

■ 在新聚类质心基础上，根据欧氏距离大小，将每个待聚类数据放入唯一一个聚类集合中

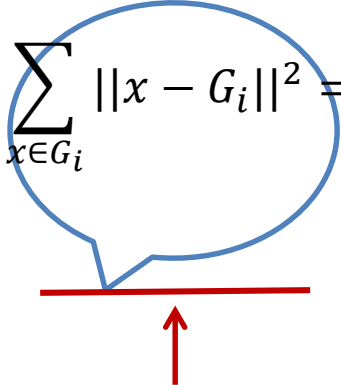
■ 根据新的聚类结果、更新聚类质心

聚类迭代满足如下任意一个条件，则聚类停止：

- 已经达到了迭代次数上限
- 前后两次迭代中，聚类质心基本保持不变

K均值聚类算法的另一个视角：最小化每个类簇的方差

■ 方差：用来计算变量（观察值）与样本平均值之间的差异

$$\arg \min_G \sum_{i=1}^k \sum_{x \in G_i} \|x - G_i\|^2 = \arg \min_G \sum_{i=1}^k |G_i| \text{Var } G_i$$


第*i*个类簇的方差： $\text{var}(G_i) = \frac{1}{|G_i|} \sum_{x \in G_i} \|x - G_i\|^2$

- 欧氏距离与方差量纲相同
- 最小化每个类簇方差将使得最终聚类结果中每个聚类集合中所包含数据呈现出来差异性最小。

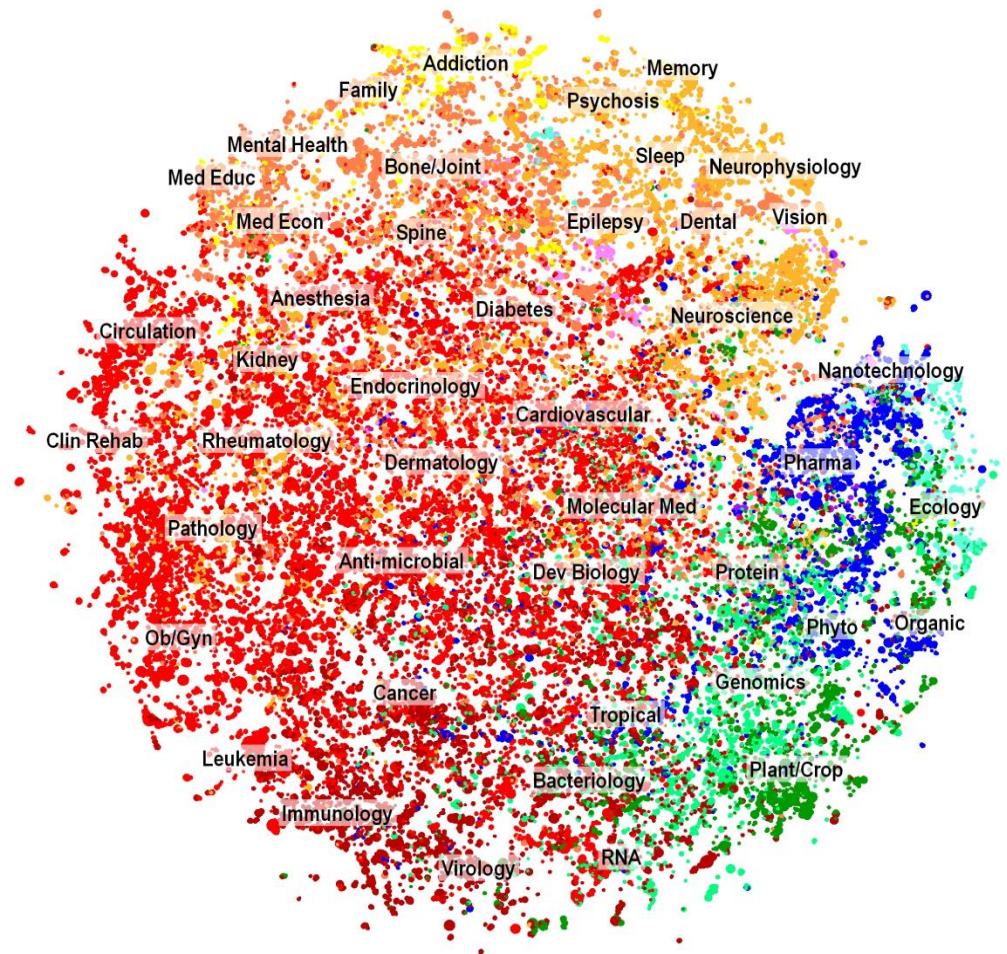
K均值聚类算法的不足

- 需要事先确定聚类数目，很多时候我们并不知道数据应被聚类的数目
- 需要初始化聚类质心，初始化聚类中心对聚类结果有较大的影响
- 算法是迭代执行，时间开销非常大
- 欧氏距离假设数据每个维度之间的重要性是一样的

K均值聚类算法的应用



图像分类



文本分类：将200多万篇论文聚类到29,000个类别，包括化学、工程、生物、传染疾病、生物信息、脑科学、社会科学、计算机科学等及给出了每个类别中的代表单词

提纲

一、K均值聚类

二、主成分分析

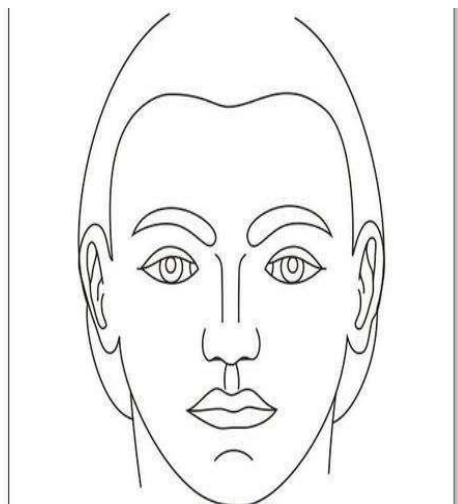
三、特征人脸方法

四、潜在语义分析

五、期望最大化算法

主成分分析: Principle Component Analysis (PCA)

- 主成分分析是一种特征降维方法（与线性区别分析的目的是一样的）。人类在认知过程中会主动“化繁为简”
- 奥卡姆剃刀定律（Occam's Razor）：“如无必要，勿增实体”，即“简单有效原理”



主成分分析: 若干概念-方差与协方差

数据样本的方差 variance

假设有 n 个数据, 记为 $X = \{x_i\} \ (i = 1, \dots, n)$

- 方差等于各个数据与样本均值之差的平方和之平均数
- 方差描述了样本数据的波动程度

$$\text{Var}(X) = \frac{1}{n} \sum_{i=1}^n (x_i - u)^2$$

其中 u 是样本均值, $u = \frac{1}{n} \sum_{i=1}^n x_i$

主成分分析: 若干概念-方差与协方差

数据样本的协方差 covariance

假设有 n 个两维变量数据, 记为 $(X, Y) = \{(x_i, y_i)\} \ (i = 1, \dots, n)$

- 衡量两个变量之间的相关度

$$\text{cov}(X, Y) = \frac{1}{n} \sum_{i=1}^n (x_i - E(X))(y_i - E(Y))$$

其中 $E(X)$ 和 $E(Y)$ 分别是 X 和 Y 的样本均值, 分别定义如下

$$E(X) = \frac{1}{n} \sum_{i=1}^n x_i, \quad E(Y) = \frac{1}{n} \sum_{i=1}^n y_i$$

主成分分析: 协方差例子

表5.1 方差与协方差的计算例子（无偏估计）

表5.1 方差与协方差的计算例子（无偏估计）					
编号	x_i	y_i	$x_i - E(X)$	$y_i - E(Y)$	$[x_i - E(X)][y_i - E(Y)]$
1	1	7	-8.33	-16.67	138.89
2	3	11	-6.33	-12.67	80.22
3	6	17	-3.33	-6.67	22.22
4	10	25	0.67	1.33	0.89
5	15	35	5.67	11.33	64.22
6	21	47	11.67	23.33	272.22
$E(X) = 9.33$		$E(Y) = 23.67$	$\text{var}(X) = 57.87$	$\text{var}(Y) = 231.47$	$\text{cov}(X, Y) = 115.73$

$$X = \{x_i\}, Y = \{y_i\}$$

主成分分析: 协方差例子

- 对于一组两维变量（如广告投入-商品销售、天气状况-旅游出行等），可通过计算它们之间的协方差值来判断这组数据给出的两维变量是否存在关联关系：
- 当协方差 $cov(X, Y) > 0$ 时，称 X 与 Y 正相关
- 当协方差 $cov(X, Y) < 0$ 时，称 X 与 Y 负相关
- 当协方差 $cov(X, Y) = 0$ 时，称 X 与 Y 不相关（线性意义下）

主成分分析: 从协方差到相关系数

可通过皮尔逊相关系数（Pearson Correlation coefficient）将两组变量之间的关联度规整到一定的取值范围内。皮尔逊相关系数定义如下：

$$\text{corr}(X, Y) = \frac{\text{Cov}(X, Y)}{\sqrt{\text{Var}(X)\text{Var}(Y)}} = \frac{\text{Cov}(X, Y)}{\sigma_x \sigma_y}$$

表5.1 方差与协方差的计算例子（无偏估计）

$$\text{corr}(X, Y) = 1$$

表5.1 方差与协方差的计算例子（无偏估计）

编号	x_i	y_i	$x_i - E(X)$	$y_i - E(Y)$	$[x_i - E(X)][y_i - E(Y)]$
1	1	7	-8.33	-16.67	138.89
2	3	11	-6.33	-12.67	80.22
3	6	17	-3.33	-6.67	22.22
4	10	25	0.67	1.33	0.89
5	15	35	5.67	11.33	64.22
6	21	47	11.67	23.33	272.22
$E(X) = 9.33$		$E(Y) = 23.67$	$\text{var}(X) = 57.87$	$\text{var}(Y) = 231.47$	$\text{cov}(X, Y) = 115.73$

主成分分析: 从协方差到相关系数

皮尔逊相关系数所具有的性质如下:

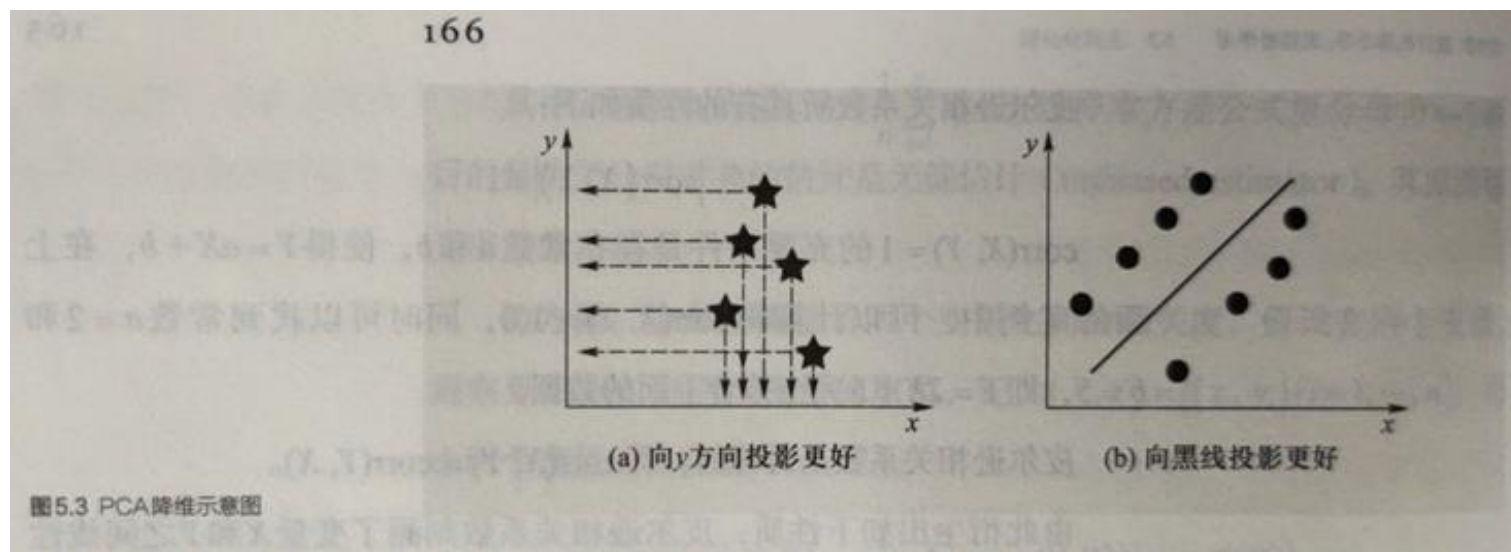
- $|\text{corr}(X, Y)| \leq 1$
- $\text{corr}(X, Y) = 1$ 的充要条件是存在常数 a 和 b , 使得 $Y = aX + b$
- 皮尔逊相关系数是对称的, 即 $\text{corr}(X, Y) = \text{corr}(Y, X)$
- 由此衍生出如下性质: 皮尔逊相关系数刻画了变量 X 和 Y 之间线性相关程度, 如果 $|\text{corr}(X, Y)|$ 的取值越大, 则两者在线性相关的意义下相关程度越大。 $|\text{corr}(X, Y)| = 0$ 表示两者不存在线性相关关系 (可能存在其他非线性相关的关系)。
- 正线性相关意味着变量 X 增加的情况下, 变量 Y 也随之增加; 负线性相关意味着变量 X 减少的情况下, 变量 Y 随之增加。

主成分分析: 从协方差到相关系数

- 相关性(correlation)与独立性(independence)
 - 如果 X 和 Y 的线性不相关, 则 $|corr(X, Y)| = 0$
 - 如果 X 和 Y 的彼此独立, 则一定 $|corr(X, Y)| = 0$, 且 X 和 Y 不存在任何线性或非线性关系
 - “不相关”是一个比“独立”要弱的概念, 即独立一定不相关, 但是不相关不一定相互独立 (可能存在其他复杂的关联关系)。独立指两个变量彼此之间不相互影响。

- 在数理统计中，方差被经常用来度量数据和其数学期望（即均值）之间偏离程度，这个偏离程度反映了数据分布结构。
- 在许多实际问题中，研究数据和其均值之间的偏离程度有着很重要的意义。
- 在降维之中，需要尽可能将数据向方差最大方向进行投影，使得数据所蕴含信息没有丢失，彰显个性。如左下图所示，向y方向投影（使得二维数据映射为一维）就比向x方向投影结果在降维这个意义上而言要好；右下图则是向黑斜线方向投影要好。

图5.3 PCA降维示意图



主成分分析: 算法动机

- 主成分分析思想是将 n 维特征数据映射到 l 维空间 ($n \gg l$)，去除原始数据之间的冗余性（通过去除相关性手段达到这一目的）。
- 将原始数据向这些数据方差最大的方向进行投影。一旦发现了方差最大的投影方向，则继续寻找保持方差第二的方向且进行投影。
- 将每个数据从 n 维高维空间映射到 l 维低维空间，每个数据所得到最好的 k 维特征就是使得每一维上样本方差都尽可能大。

主成分分析: 算法描述

- 假设有 n 个 d 维样本数据所构成的集合 $D = \{\mathbf{x}_1, \mathbf{x}_2, \dots, \mathbf{x}_n\}$, 其中 $\mathbf{x}_i (1 \leq i \leq n) \in R^d$ 。
- 集合 D 可以表示成一个 $n \times d$ 的矩阵 \mathbf{X} 。
- 假定每一维度的特征均值均为零（已经标准化）。
- 主成分分析的目的是求取一个且使用一个 $d \times l$ 的映射矩阵 \mathbf{W} 。
- 给定一个样本 \mathbf{x}_i , 可将 \mathbf{x}_i 从 d 维空间如下映射到 l 维空间: $(\mathbf{x}_i)_{1 \times d} (\mathbf{W})_{d \times l}$
- 将所有降维后数据用 \mathbf{Y} 表示, 有 $\mathbf{Y} = \mathbf{X} \mathbf{W}$

? 如何求取
映射矩阵 \mathbf{W}

降维 原始 映射
结果 数据 矩阵

- $\mathbf{Y} = n \times l$
- $\mathbf{X} = n \times d$
- $\mathbf{W} = d \times l$

主成分分析: 算法描述

$$\mathbf{Y} = n \times l \quad \mathbf{X} = n \times d \quad \mathbf{W} = d \times l$$

降维后 n 个 l 维样本数据 \mathbf{Y} 的方差为:

$$\text{var}(\mathbf{Y}) = \frac{1}{n-1} \text{trace}(\mathbf{Y}^T \mathbf{Y})$$

$$= \frac{1}{n-1} \text{trace}(\mathbf{W}^T \mathbf{X}^T \mathbf{X} \mathbf{W})$$

$$= \text{trace}(\mathbf{W}^T \frac{1}{n-1} \mathbf{X}^T \mathbf{X} \mathbf{W})$$

降维前 n 个 d 维样本数据 \mathbf{X} 的协方差矩阵记为:

$$\Sigma = \frac{1}{n-1} \mathbf{X}^T \mathbf{X}$$

主成份分析的求解目标函数为

$$\max_{\mathbf{W}} \text{trace}(\mathbf{W}^T \Sigma \mathbf{W})$$

满足约束条件

$$\mathbf{w}_i^T \mathbf{w}_i = 1 \quad i \in \{1, 2, \dots, l\}$$

主成分分析: 算法描述

所有带约束的最优化问题，可通过拉格朗日乘子法将其转化为无约束最优化问题

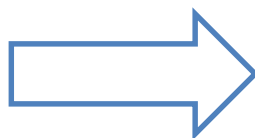
主成份分析求解目标函数为

$$\max_{\mathbf{W}} \text{trace}(\mathbf{W}^T \Sigma \mathbf{W})$$

满足约束条件

$$\mathbf{w}_i^T \mathbf{w}_i = 1 \quad i \in \{1, 2, \dots, l\}$$

拉格朗日
函数



$$\mathbf{Y} = n \times l \quad \mathbf{X} = n \times d \quad \mathbf{W} = d \times l$$

$$L(\mathbf{W}, \lambda) = \text{trace}(\mathbf{W}^T \Sigma \mathbf{W}) - \sum_{i=1}^l \lambda_i (\mathbf{w}_i^T \mathbf{w}_i - 1)$$

其中 $\lambda_i (1 \leq i \leq l)$ 为拉格朗日乘子， \mathbf{w}_i 为矩阵 \mathbf{W} 第 i 列。

对上述拉格朗日函数中变量 \mathbf{w}_i 求偏导并令导数为零，得到

$$\Sigma \mathbf{w}_i = \lambda_i \mathbf{w}_i$$

上式表明：每一个 \mathbf{w}_i 都是 n 个 d 维样本数据 \mathbf{X} 的协方差矩阵 Σ 的一个特征向量， λ_i 是这个特征向量所对应的特征值。

保证降维后结果正交以
去除相关性（即冗余度）

主成分分析: 算法描述

$$\mathbf{Y} = n \times l \quad \mathbf{X} = n \times d \quad \mathbf{W} = d \times l$$

$$\Sigma \mathbf{w}_i = \lambda_i \mathbf{w}_i, \text{ 且 } \text{trace}(\mathbf{W}^T \Sigma \mathbf{W}) = \sum_{i=1}^l \mathbf{w}_i^T \Sigma \mathbf{w}_i = \sum_{i=1}^l \lambda_i$$

- 可见，在主成份分析中，最优化的方差等于原始样本数据 \mathbf{X} 的协方差矩阵 Σ 的特征根之和。
- 要使方差最大，我们可以求得协方差矩阵 Σ 的特征向量和特征根，然后取前 l 个最大特征根所对应的特征向量组成映射矩阵 \mathbf{W} 即可。
- 注意，每个特征向量 \mathbf{w}_i 与原始数据 x_i 的维数是一样的，均为 d 。

其他常用降维方法

- 非负矩阵分解 非负矩阵分解 (non-negative matrix factorization, NMF)
- 多维尺度法(Metric multidimensional scaling, MDS)
- 局部线性嵌入 (Locally Linear Embedding, LLE)

其他常用降维方法

- 非负矩阵分解 非负矩阵分解 (non-negative matrix factorization, NMF)
 - 该方法将非负的大矩阵分解成两个非负的小矩阵。
 - 矩阵分解的一般目标为将原始矩阵 D 表示为矩阵 W 和矩阵 H 相乘的形式, 即 $D = WH$ 。
 - 主成分分析方法即可以实现这一点, 不过主成分分析不要求原始矩阵 D 中所有元素为正数 (即非负)。
 - 在很多实际情况中, 如图像所组成的像素矩阵或自然语言中常用的单词-文档矩阵中并不存在为负数的元素, 因此, 对非负矩阵的分解是很有意义的问题。

其他常用降维方法

- 非负矩阵分解 非负矩阵分解 (non-negative matrix factorization, NMF)

非负矩阵分解将给定的非负矩阵 $D \in \mathbb{R}^{n \times d}$ 分解为两个非负矩阵 $W \in \mathbb{R}^{n \times l}$ 和 $H \in \mathbb{R}^{l \times d}$, 并满足下面的条件:

$$D \approx WH \quad D_{i,j}, W_{i,\mu}, H_{\mu,j} \geq 0$$

其中, $0 \leq i < n - 1, 0 \leq j < d - 1, 0 \leq \mu < l - 1$ 。

为了得到原始矩阵 D 的分解矩阵 W 和 H , 首先定义原始矩阵 D 与 WH 之间的误差, 即代价函数 (cost function)。代价函数被用来度量原始矩阵 D 与其近似矩阵 WH 之间的误差, 这样求取原始矩阵的分解矩阵 W 和 H 的问题可以转变为一个误差最小化问题。

其他常用降维方法

● 非负矩阵分解 非负矩阵分解 (non-negative matrix factorization, NMF)

两种常用的代价函数如下

(1) 欧式距离平方函数

$$\|D - WH\|_F^2 = \sum_{0 \leq i < n, 0 \leq j < d} (D_{i,j} - (WH)_{i,j})^2$$

(2) KL散度 (Kullback–Leibler divergence) 函数

$$D_{KL}(D||WH) = \sum_{0 \leq i < n, 0 \leq j < d} D_{i,j} \log \frac{D_{i,j}}{(WH)_{i,j}} - D_{i,j} + (WH)_{i,j}$$

当代价函数是欧式距离平方函数时，矩阵 W 和 H 可使用如下的迭代方法求解：

$$H_{a,\mu} \leftarrow H_{a,\mu} \frac{(W^T D)_{a,\mu}}{(W^T WH)_{a,\mu}} \quad W_{ia} \leftarrow W_{ia} \frac{(DH^T)_{i,a}}{(WHH^T)_{i,a}}$$

其中， $0 \leq a < l, 0 \leq \mu < d, 0 \leq i < n$ 。

当代价函数是KL散度函数时，矩阵 W 和 H 可使用如下的迭代方法求解：

$$H_{a,\mu} \leftarrow H_{a,\mu} \frac{\sum_i W_{ia} D_{iu} / (WH)_{iu}}{\sum_k W_{ka}} \quad W_{ia} \leftarrow W_{ia} \frac{\sum_u H_{au} D_{iu} / (WH)_{iu}}{\sum_v W_{av}}$$

其他常用降维方法

- 非负矩阵分解 非负矩阵分解 (non-negative matrix factorization, NMF)

非负矩阵分解依据了“对整体的感知是由对组成整体的部分进行感知而构成的(纯加性的)”这一观点，即“部分组成整体 (part-of-the-whole) ”。这也符合直观的理解：整体是由部分组成的。其要求分解后的所有分量均为非负值（即纯加性描述），并且同时实现非线性的维数降维。

其他常用降维方法

- 多维尺度法(Metric multidimensional scaling, MDS)

多维尺度法(Metric multidimensional scaling, MDS)保持原始数据之间两两距离不变。MDS算法会计算原始数据两两之间的距离，形成一个距离矩阵，以此来保证降维之前距离近的数据在降维之后也能很接近。不过，也正因为MDS需要知道这样的距离矩阵，所以无法对新数据集合进行降维（比如测试集），这被称为“out-of-sample”问题。

其他常用降维方法

● 多维尺度法(Metric multidimensional scaling, MDS)

令 N 是原始数据数量, 每个数据由 d 维特征表示, 即 $x_i = (x_{i1}, x_{i2}, \dots, x_{id})$, 可以将所有原始数据表示为 X , $X = [x_1, \dots, x_i, \dots, x_n]$ 。多维尺度法首先计算任意两个数据 x_i 和 x_j ($1 \leq i, j \leq n$) 之间的距离 s_{ij} , 得到距离矩阵 A , 即 $A = (s_{ij})$ 。 A 是一个 $n \times n$ 大小的矩阵, x_i 和 x_j 之间的距离度量函数由具体应用来定 (如可选取余弦距离等)。

接着计算中心化矩阵 $H = I - \frac{1}{n} \mathbf{1}_n \mathbf{1}_n^T$, 其中 I 是单位矩阵, $\mathbf{1}_n$ 是全 1 的 n 维列向量, 可以得到如下 $n \times n$ 大小的矩阵 B :

$$B = -\frac{1}{2} H A H$$

假设矩阵 B 的秩为 p , 对矩阵 B 进行奇异值分解, 可得:

$$B = \Gamma \Lambda \Gamma^T$$

其中 $\Lambda = \text{diag}(\lambda_1, \dots, \lambda_l)$ 是一个对角矩阵, 对角线上的元素为矩阵 B 的非零特征根; $\Gamma = (\gamma_1, \dots, \gamma_l)$ 是一个 $n \times l$ 矩阵, 其中 γ_j 为特征根 λ_j ($1 \leq j \leq l$) 所对应的特征向量。

于是 $X = [x_1, \dots, x_i, \dots, x_n]$ 中 d 维原始数据可通过如下变换转变为 l 维数据, 从而实现降维:

$$\bar{X}^T = [\bar{x}_1, \bar{x}_2, \dots, \bar{x}_n]^T = \Gamma \Lambda^{1/2}$$

其中 \bar{x}_i ($1 \leq i \leq n$) 表示原始数据 x_i 降维后的结果。

其他常用降维方法

- 局部线性嵌入 (Locally Linear Embedding, LLE)

上述的PCA和MDS都属于线性降维方法，LLE是一种非线性降维方法。LLE的基本假设是：一个流形的局部可以近似于一个欧式空间，每个样本均可以利用其邻居进行线性重构。换句话说，如果假设数据是局部线性的（即使数据的原始高维空间是非线性流形嵌入），于是可以用和邻居数据的线性关系进行局部重构。

局部线性嵌入保留了每个数据的局部性质，利用局部线性来逼近全局非线性，通过相互重叠的局部邻域提供全局结构信息，最终保持数据整体几何信息。

其他常用降维方法

● 局部线性嵌入 (Locally Linear Embedding, LLE)

局部线性嵌入算法的基本步骤如下：

(1) 假设 n 个原始数据 $x_i (1 \leq i \leq n)$ 的维数为 d ，计算任意数据 $x_i (1 \leq i \leq n)$ 的 k 近邻。

(2) 如下计算数据点的局部重建权值，使样本点的重建误差最小：

$$\min_{w_{ij}} \sum_{i=1}^n \left\| x_i - \sum_{x_j \in N_k(x_i)} w_{ij} x_j \right\|^2$$

$$\text{s.t.} \quad \sum_{x_j \in N_k(x_i)} w_{ij} = 1, i = 1, \dots, n$$

其中， $N_k(x_i)$ 表示 x_i 的 k 个近邻， w_{ij} 表示使用近邻 x_j 来重建 x_i 时的权重， k 个近邻的权重和为 1。

(3) 将每个数据点重建权重组成矩阵 W ，实际上是要寻找原始数据 $x_i (1 \leq i \leq n)$ 的低维嵌入 $y_i (1 \leq i \leq n)$ ，使得重建误差最小，即 $\min \Phi(Y) = \sum_{i=1}^n \|y_i - \sum_{j=1}^n w_{ij} y_j\|^2$ 。

(4) 在具体计算中，需要对降维结果加以一定约束，即 $\sum_{i=1}^n y_i = 0, \frac{1}{n} \sum_{i=1}^n y_i y_i^T = I$ ，其中 I 是 n 维单位矩阵。这样，局部线性嵌入就变为如下优化函数：

$$\min \Phi(Y) = \sum_{i=1}^n \|Y I_i - Y W_i\|^2 = \sum_{i=1}^n \|Y(I_i - W_i)\|^2 = \min \text{tr}(Y M Y^T)$$

$$\text{s.t. } Y Y^T = I$$

其中 $M = (I - W)^T(I - W)$ 。使用拉格朗日乘子，可得 $M Y^T = \lambda Y^T$ ，取矩阵

M 最小的前 k 个非零特征根所对应特征向量构成的矩阵即可将原始数据 $x_i (1 \leq i \leq n)$ 从 d 维空间转变为 k 维空间中的 $y_i (1 \leq i \leq n)$ 。在实际计算中，矩阵 M 最小特征值几乎为零，因此通常取第 $2 \sim (k+1)$ 之间的特征值所对应特征向量。

提纲

一、K均值聚类

二、主成分分析

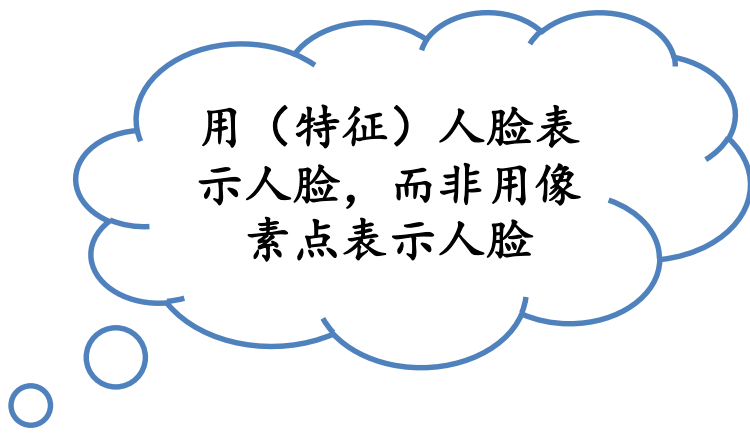
三、特征人脸方法

四、潜在语义分析

五、期望最大化算法

特征人脸方法: 动机

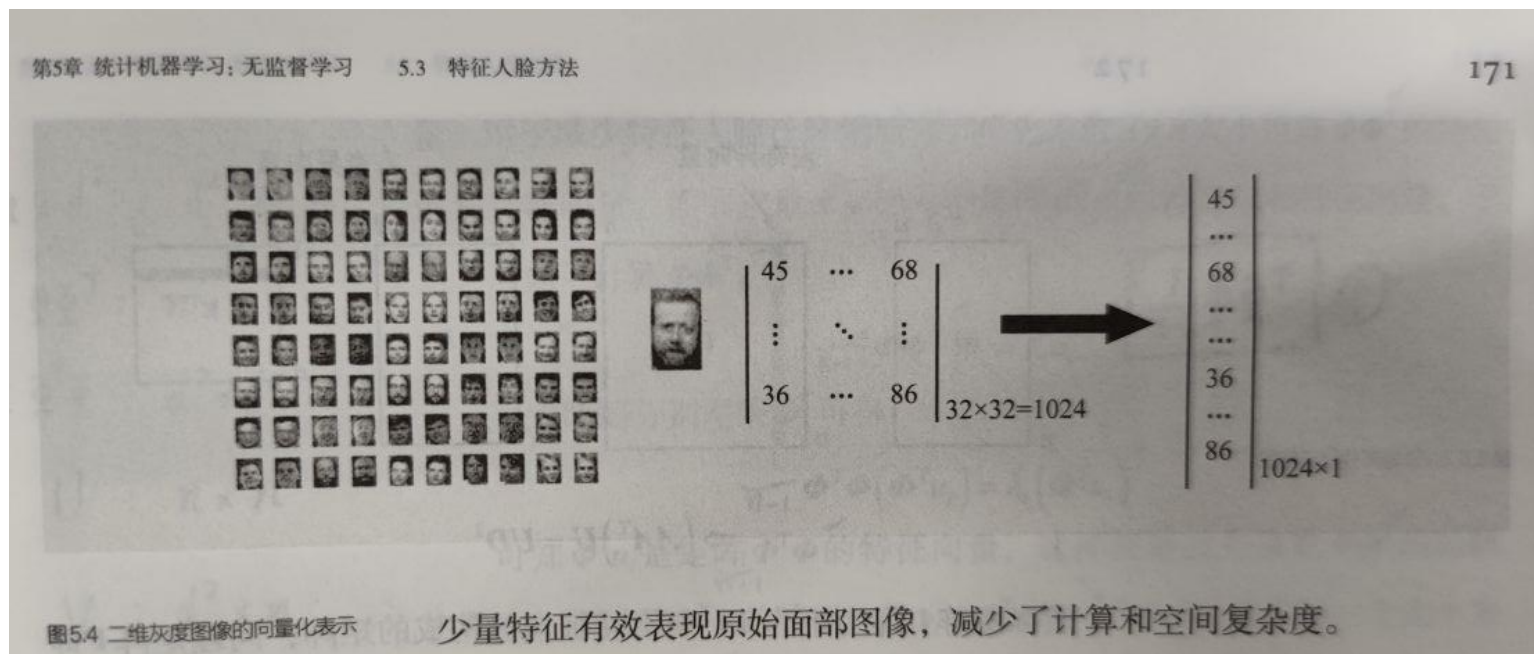
- 特征人脸方法是一种应用主成份分析来实现人脸图像降维的方法，其本质是用一种称为“特征人脸(eigenface)”的特征向量按照线性组合形式来表达每一张原始人脸图像，进而实现人脸识别。
- 由此可见，这一方法的关键之处在于如何得到特征人脸。



用（特征）人脸表示人脸，而非用像素点表示人脸

特征人脸方法: 算法描述

图5.4 二维灰度图像的向量化表示



- 将每幅人脸图像转换成列向量
- 如将一幅 32×32 的人脸图像转成 1024×1 的列向量

特征人脸: 算法描述

$$Y = n \times l \quad X = n \times d \quad W = d \times l$$

- 输入: n 个1024维人脸样本数据所构成的矩阵 X , 降维后的维数 l
- 输出: 映射矩阵 $W = \{w_1, w_2, \dots, w_l\}$ (其中每个 $w_j (1 \leq j \leq l)$ 是一个特征人脸)
- 算法步骤:

1: 对于每个人脸样本数据 x_i 进行中心化处理: $x_i = x_i - \mu, \mu = \frac{1}{n} \sum_{j=1}^n x_j$

2: 计算原始人脸样本数据的协方差矩阵: $\Sigma = \frac{1}{n-1} X^T X$

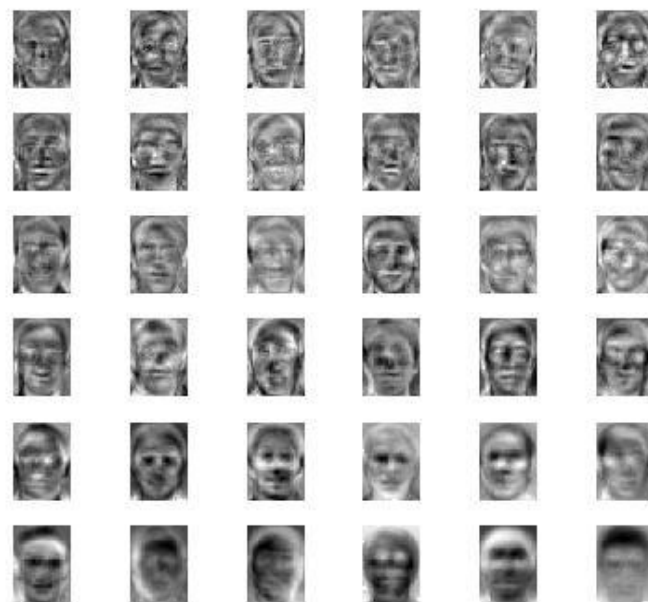
3: 对协方差矩阵 Σ 进行特征值分解, 对所得特征根从小到大排序 $\lambda_1 \geq \lambda_2 \geq \dots \geq \lambda_d$

4: 取前 l 个最大特征根所对应特征向量 w_1, w_2, \dots, w_l 组成映射矩阵 W

5: 将每个人脸图像 x_i 按照如下方法降维: $(x_i)_{1 \times d} (W)_{d \times l} = 1 \times l$

特征人脸: 算法描述

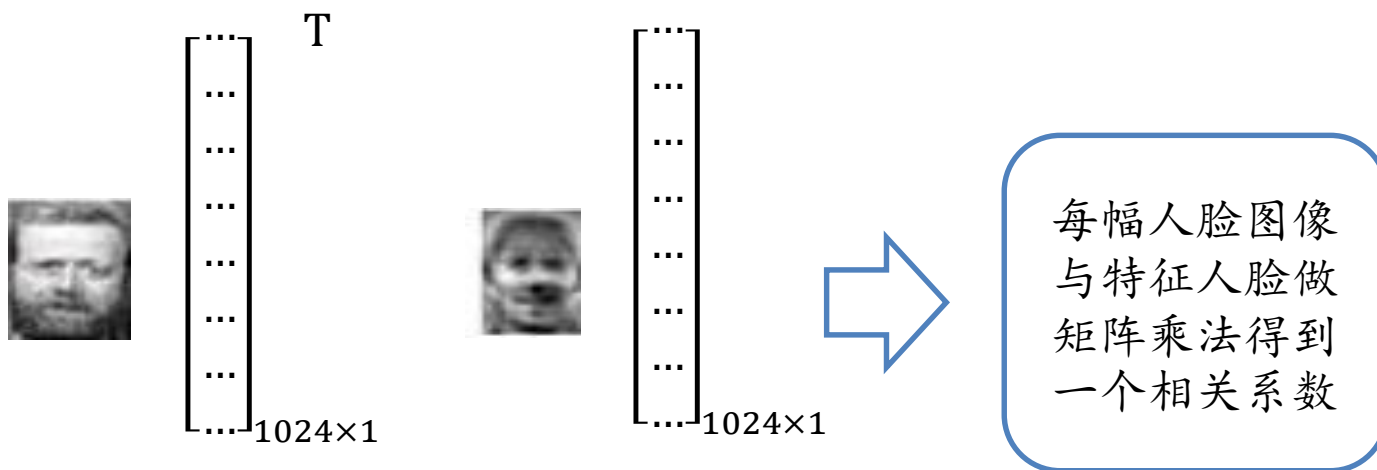
- 每个人脸特征向量 w_i 与原始人脸数据 x_i 的维数是一样的，均为1024。
- 可将每个特征向量还原为 32×32 的人脸图像，称之为特征人脸，因此可得到 l 个特征人脸。



400个人脸（左）和与之对应的36个特征人脸

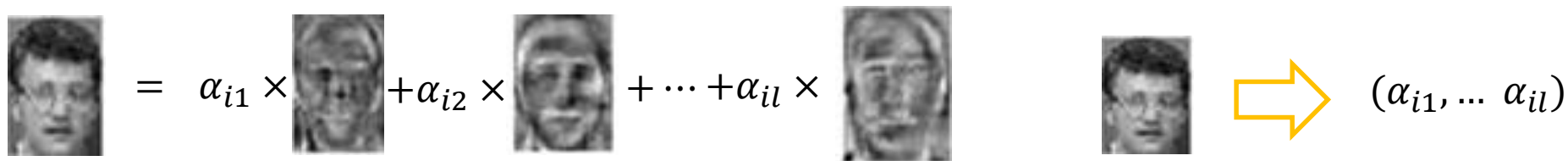
基于特征人脸的降维

- 将每幅人脸分别与每个特征人脸做矩阵乘法，得到一个相关系数
- 每幅人脸得到 l 个相关系数 \Rightarrow 每幅人脸从1024维约减到 l 维



基于特征人脸的降维

- 由于每幅人脸是所有特征人脸的线性组合，因此就实现人脸从“像素点表达”到“特征人脸表达”的转变。每幅人脸从1024维约减到 l 维。


$$x_i = \alpha_{i1} \times \text{feature}_1 + \alpha_{i2} \times \text{feature}_2 + \dots + \alpha_{il} \times \text{feature}_l \rightarrow (\alpha_{i1}, \dots, \alpha_{il})$$

x_i

使用 l 个特征人脸的线性组合来表达原始人脸数据 x_i

x_i 的像素点
空间表达
 32×32

x_i 的人脸子
空间的 l 个系
数表达

在后续人脸识别分类中，就使用这 l 个系数来表示原始人脸图像。即计算两张人脸是否相似，不是去计算两个 32×32 矩阵是否相似，而是计算两个人脸所对应的 l 个系数是否相似

人脸表达的方法对比：聚类、主成份分析、非负矩阵分解



x_i

聚类表示：
用待表示人脸最相似的
聚类质心来
表示



x_i

$$x_i = \alpha_{i1} \times \text{[face 1]} + \alpha_{i2} \times \text{[face 2]} + \dots + \alpha_{il} \times \text{[face l]}$$

特征人脸(主成分分析) 表示：使用 l 个特征人脸的线性组合来表达原始人脸数据 x_i



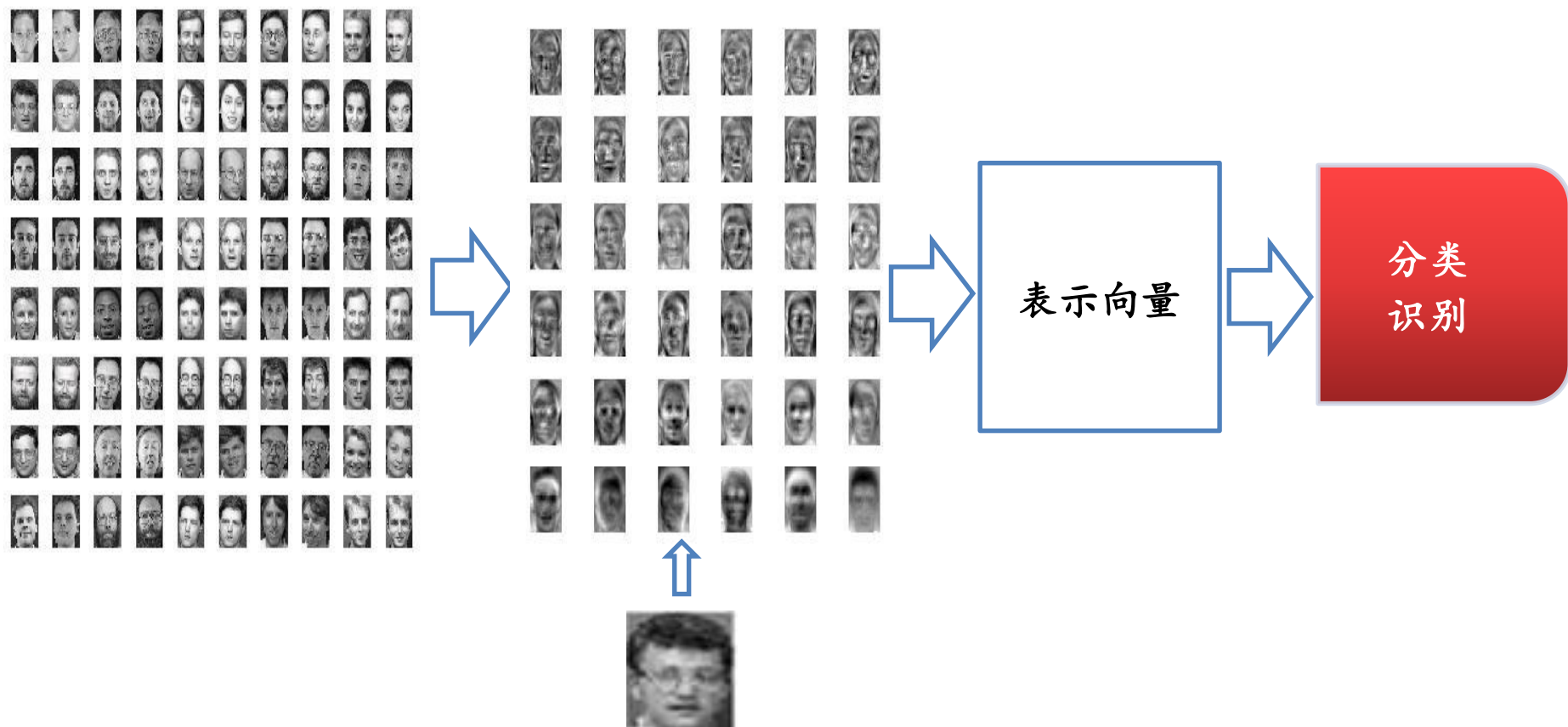
x_i



非负矩阵人脸分解方法表示：通过若干个特征人脸的线性组合来表达原始人脸数据 x_i ，体现了“部分组成整体”

Daniel D. Lee & H. Sebastian Seung, Learning the parts of objects by non-negative matrix factorization, 1999, [Nature](#)

人脸表达后的分析与处理



提纲

一、K均值聚类

二、主成分分析

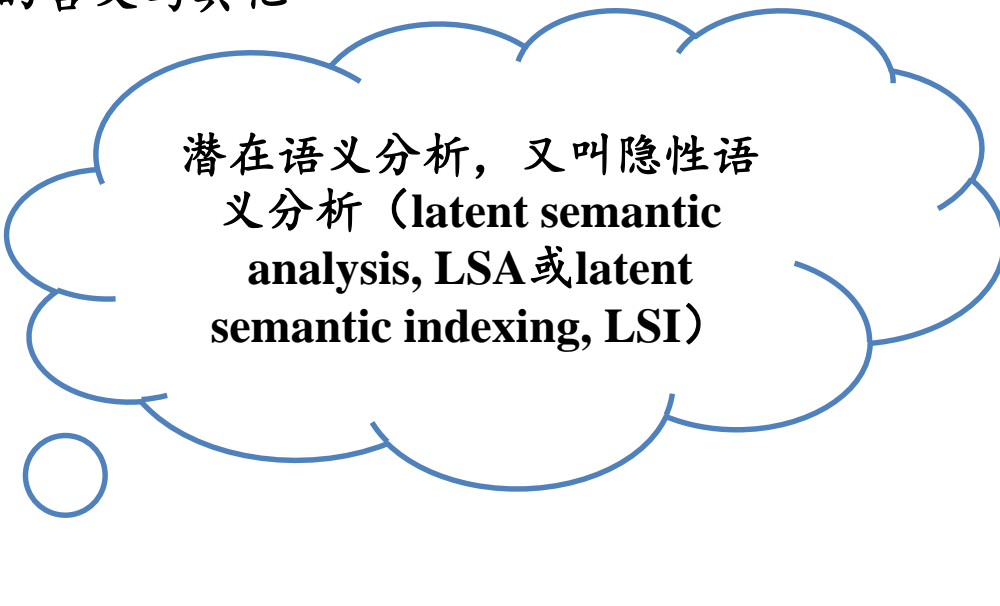
三、特征人脸方法

四、潜在语义分析

五、期望最大化算法

潜在语义分析: 动机

- 潜在语义分析是一种从海量文本数据中学习单词-单词、单词-文档以及文档-文档之间隐性关系，进而得到文档和单词表达特征的方法。
- 该方法的基本思想是综合考虑某些单词在哪些文档中同时出现，以此来决定该词语的含义与其他词语的相似度。



潜在语义分析，又叫隐性语义分析 (latent semantic analysis, LSA 或 latent semantic indexing, LSI)

潜在语义分析思想

潜在语义分析先构建一个单词-文档 (term-document) 矩阵A, 进而寻找该矩阵的低秩逼近 (low rank approximation) [Markovsky 2012], 来挖掘单词-单词、单词-文档以及文档-文档之间的关联关系。

机器学习 (Machine Learning) :

- a1: Efficient Algorithms for Non-convex Isotonic Regression through Submodular Optimization
- a2: Combinatorial Optimization with Graph Convolutional Networks and Guided Tree Search
- a3: An Improved Analysis of Alternating Minimization for Structured Multi-Response Regression
- a4: Analysis of Krylov Subspace Solutions of Regularized Non-Convex Quadratic Problems
- a5: Post: Device Placement with Cross-Entropy Minimization and Proximal Policy Optimization

基因编辑 (gene editing) 类别:

- b1: CRISPR/Cas9 and TALENs generate heritable mutations for genes involved in small RNA processing of Glycine max and Medicago truncatula
- b2: Generation of D1-1 TALEN isogenic control cell line from Dravet syndrome patient iPSCs using TALEN-mediated editing of the SCN1A gene
- b3: Genome-Scale CRISPR Screening Identifies Novel Human Pluripotent Gene Networks
- b4: Champions: A phase 1/2 clinical trial with dose escalation of SB-913 ZFN-mediated in vivo human genome editing for treatment of MPS II (Hunter syndrome)

潜在语义分析

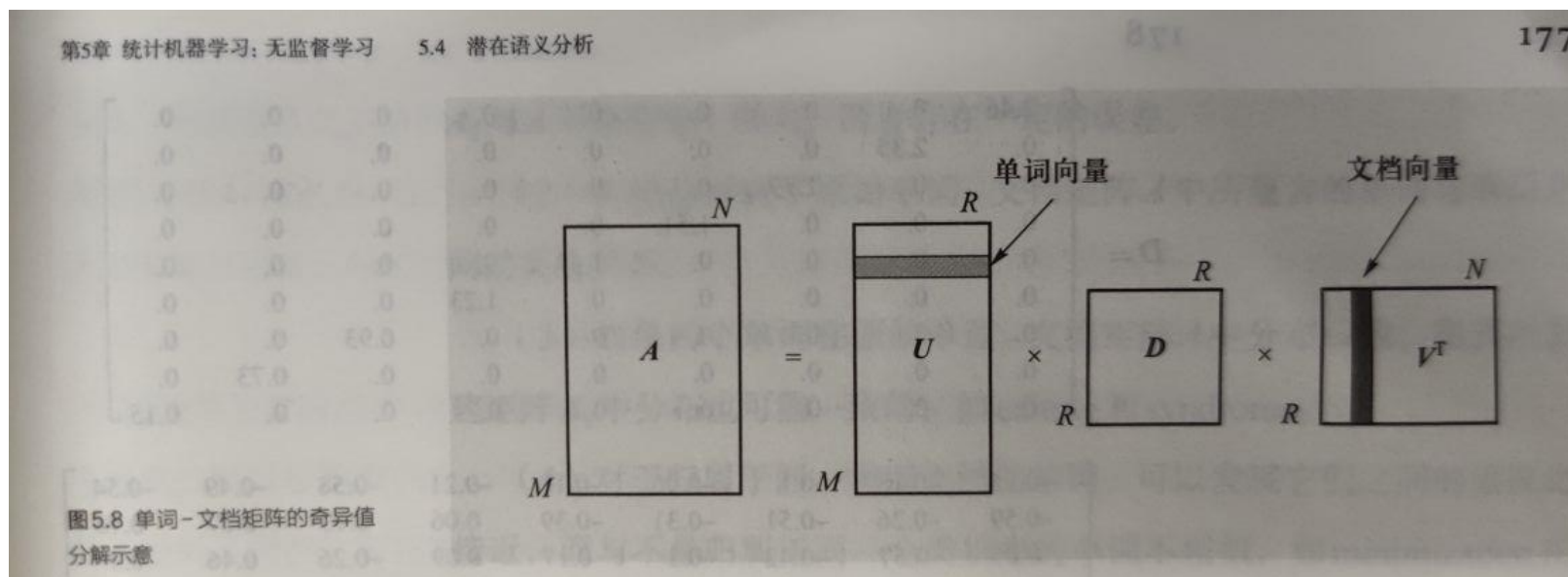
第一步：计算单词-文档矩阵

表5.2 单词-文档 (term-document) 矩阵

[illegible]

■ 第二步：奇异值分解

图5.8 单词-文档矩阵的奇异值分解示意



潜在语义分析

■ 第三步：重建矩阵

当 $k=2$ ，即选取最大的前两个特征根及其对应的特征向量对矩阵 A 进行重建。下面给出了选取矩阵 U 、矩阵 D 和矩阵 V 的子部分重建所得矩阵 A_2 ：

$$A_2 = \left\{ \begin{array}{lcccccccccc} & a1 & a2 & a3 & a4 & a5 & b1 & b2 & b3 & b4 \\ nonconvex & 0.56 & 0.27 & 0.49 & 0.3 & 0.37 & -0.01 & -0.05 & 0.05 & -0.05 \\ regression & 0.68 & 0.33 & 0.59 & 0.36 & 0.45 & -0.01 & -0.06 & 0.05 & -0.06 \\ optimization & 0.79 & 0.4 & 0.68 & 0.41 & 0.53 & 0.02 & 0.02 & 0.14 & 0.02 \\ network & 0.22 & 0.18 & 0.17 & 0.1 & 0.15 & 0.13 & 0.36 & 0.32 & 0.33 \\ analysis & 0.51 & 0.25 & 0.44 & 0.27 & 0.34 & -0.01 & -0.06 & 0.03 & -0.06 \\ minimization & 0.56 & 0.27 & 0.49 & 0.3 & 0.37 & -0.01 & -0.05 & 0.05 & -0.05 \\ gene & 0.01 & 0.16 & -0.03 & -0.02 & 0.02 & 0.29 & 0.81 & 0.66 & 0.75 \\ syndrome & -0.05 & 0.11 & -0.07 & -0.05 & -0.02 & 0.26 & 0.72 & 0.58 & 0.67 \\ editing & -0.05 & 0.11 & -0.07 & -0.05 & -0.02 & 0.26 & 0.72 & 0.58 & 0.67 \\ human & 0.01 & 0.13 & -0.02 & -0.01 & 0.02 & 0.23 & 0.65 & 0.53 & 0.6 \end{array} \right\}$$

潜在语义分析

■ 第四步：挖掘语义关系

- 基于单词-文档矩阵A，可以计算任意两个文档之间的皮尔逊相关系数，从而得到如下文档-文档相关系数矩阵：

$$\begin{pmatrix} & a1 & a2 & a3 & a4 & a5 & b1 & b2 & b3 & b4 \\ a1 & 1. & 0.22 & 0.05 & 0.22 & 0.22 & -0.22 & -0.43 & -0.43 & -0.43 \\ a2 & 0.22 & 1. & -0.33 & -0.25 & 0.37 & -0.17 & -0.33 & 0.22 & -0.33 \\ a3 & 0.05 & -0.33 & 1. & 0.22 & 0.22 & -0.22 & -0.43 & -0.43 & -0.43 \\ a4 & 0.22 & -0.25 & 0.22 & 1. & -0.25 & -0.17 & -0.33 & -0.33 & -0.33 \\ a5 & 0.22 & 0.37 & 0.22 & -0.25 & 1. & -0.17 & -0.33 & -0.33 & -0.33 \\ b1 & -0.22 & -0.17 & -0.22 & -0.17 & -0.17 & 1. & 0.51 & 0.51 & -0.22 \\ b2 & -0.43 & -0.33 & -0.43 & -0.33 & -0.33 & 0.51 & 1. & 0.05 & 0.52 \\ b3 & -0.43 & 0.22 & -0.43 & -0.33 & -0.33 & 0.51 & 0.05 & 1. & 0.05 \\ b4 & -0.43 & -0.33 & -0.43 & -0.33 & -0.33 & -0.22 & 0.52 & 0.05 & 1. \end{pmatrix}$$

- 基于重建单词-文档矩阵A₂，可以计算任意两个文档之间的皮尔逊相关系数，从而得到如下文档-文档相关系数矩阵：

$$\begin{pmatrix} & a1 & a2 & a3 & a4 & a5 & b1 & b2 & b3 & b4 \\ a1 & 1. & 0.97 & 1. & 1. & 1. & -0.95 & -0.95 & -0.93 & -0.95 \\ a2 & 0.97 & 1. & 0.97 & 0.97 & 0.98 & -0.85 & -0.86 & -0.83 & -0.86 \\ a3 & 1. & 0.97 & 1. & 1. & 1. & -0.95 & -0.96 & -0.94 & -0.96 \\ a4 & 1. & 0.97 & 1. & 1. & 1. & -0.95 & -0.96 & -0.94 & -0.96 \\ a5 & 1. & 0.98 & 1. & 1. & 1. & -0.94 & -0.95 & -0.93 & -0.95 \\ b1 & -0.95 & -0.85 & -0.95 & -0.95 & -0.94 & 1. & 1. & 1. & 1. \\ b2 & -0.95 & -0.86 & -0.96 & -0.96 & -0.95 & 1. & 1. & 1. & 1. \\ b3 & -0.93 & -0.83 & -0.94 & -0.94 & -0.93 & 1. & 1. & 1. & 1. \\ b4 & -0.95 & -0.86 & -0.96 & -0.96 & -0.95 & 1. & 1. & 1. & 1. \end{pmatrix}$$

潜在语义分析

- 通过观察两个相关系数矩阵可以发现，基于重建单词-文档矩阵 A_2 得到的相关系数矩阵中单词-单词之间的相关关系更加明确，在同一文档中出现过的单词之间相关系数趋近为1，没有同时出现在同一文档中的单词之间相关系数趋近为-1。单词network相比于其他单词更加“中立”一些，这是因为单词network既出现在了机器学习这一类别文档标题a2中，也出现在基因编辑这一类别文档b3中。不过单词network与基因编辑这一类别单词相关度更高，这是因为单词network与gene和human这两个具有明显基因编辑特点的单词同时出现在文档b3中，只与optimization这一明显具有机器学习特点的单词同时出现在文档a2中。
- 可以看出，隐性语义分析对原始单词-文档矩阵中所蕴含关系进行了有效挖掘，能刻画单词-单词、单词-文档以及文档-文档之间语义关系。

提纲

一、K均值聚类

二、主成分分析

三、特征人脸方法

四、潜在语义分析

五、期望最大化算法

模型参数估计

- 最大似然估计 (maximum likelihood estimation, MLE)

假设由 n 个数据样本构成的集合 $\mathcal{D} = \{x_1, x_2, \dots, x_n\}$ 从参数为 θ 的某个模型（如高斯模型等）以一定概率独立采样得到。于是，可以通过最大似然估计算法（maximum likelihood estimation, MLE）来求取参数 θ ，使得在参数为 θ 的模型下数据集 \mathcal{D} 出现的可能性最大，即 $\hat{\theta} = \underset{\theta}{\operatorname{argmax}} P(\mathcal{D}|\theta)$ 。

- 最大后验估计 (maximum a posteriori estimation, MAP)

或者也可利用最大后验估计（maximum a posteriori estimation, MAP）从数据集 \mathcal{D} 来如下估计参数 θ ： $\hat{\theta} = \underset{\theta}{\operatorname{argmax}} P(\theta|\mathcal{D}) = \underset{\theta}{\operatorname{argmax}} \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})}$ 。由于 $P(\mathcal{D})$ 与 θ 无关，则可得 $\underset{\theta}{\operatorname{argmax}} \frac{P(\mathcal{D}|\theta)P(\theta)}{P(\mathcal{D})} = \underset{\theta}{\operatorname{argmax}} P(\mathcal{D}|\theta)P(\theta)$ ，对这个式子取对数，得到 $\underset{\theta}{\operatorname{argmax}} \log P(\mathcal{D}|\theta) + \log P(\theta)$ 。可见，最大后验估计与最大似然估计相比，增加了一项与 θ 相关的先验概率 $P(\theta)$ 。

模型参数估计

- 无论是最大似然估计算法或者是最大后验估计算法，都是充分利用已有数据，在参数模型确定（只是参数值未知）情况下，对所优化目标中的参数求导，令导数为0，求取模型的参数值。
- 在解决一些具体问题时，难以事先就将模型确定下来，然后利用数据来求取模型中的参数值。在这样情况下，无法直接利用最大似然估计算法或者最大后验估计算法来求取模型参数。

期望最大化算法

期望最大化 (expectation maximization, EM)

- EM算法是一种重要的用于解决含有隐变量 (latent variable) 问题的参数估计方法。
- EM算法分为求取期望 (E步骤, expectation) 和期望最大化 (M步骤, maximization) 两个步骤。
- 在EM算法的**E步骤**时, 先假设模型参数的初始值, 估计隐变量取值; 在EM算法的**M步骤**时, 基于观测数据、模型参数和隐变量取值一起来最大化“拟合”数据, 更新模型参数。基于所更新的模型参数, 得到新的隐变量取值 (EM算法的 E 步), 然后继续极大化“拟合”数据, 更新模型参数 (EM算法的M步)。以此类推迭代, 直到算法收敛, 得到合适的模型参数。

期望最大化算法：二硬币投掷例子

- 假设有A和B两个硬币，进行五轮掷币实验：在每一轮实验中，先随机选择一个硬币，然后用所选择的硬币投掷十次，将投掷结果作为本轮实验观测结果。H代表硬币正面朝上、T代表硬币反面朝上。

表5.3 两个硬币投掷5轮（每轮10次）的结果

表5.3 两个硬币投掷5轮（每轮10次）的结果

1	H	T	T	T	H	H	T	H	T	H
2	H	H	H	H	T	H	H	H	H	H
3	H	T	H	H	H	H	H	T	H	H
4	H	T	H	T	T	T	H	H	T	T
5	T	H	H	H	T	H	H	H	T	H

- 从这十轮观测数据出发，计算硬币A或硬币B被投掷为正面的概率。记硬币A或硬币B被投掷为正面的概率为 $\theta = \{\theta_A, \theta_B\}$

期望最大化算法：二硬币投掷例子

求取期望（E步骤，Expectation）：

初始化每一轮中硬币A和硬币B投掷为正面的概率为 $\hat{\theta}_A^{(0)} = 0.60$ 和 $\hat{\theta}_B^{(0)} = 0.50$ 。
基于“HTTTHHTHTH”这10次投掷结果，由硬币A投掷所得概率为：

$$\begin{aligned} & P(\text{选择硬币A投掷} | \text{硬币投掷结果}, \theta) \\ &= \frac{P(\text{选择硬币A投掷}, \text{硬币投掷结果} | \theta)}{P(\text{选择硬币A投掷}, \text{硬币投掷结果} | \theta) + P(\text{选择硬币B投掷}, \text{硬币投掷结果} | \theta)} \\ &= \frac{(0.6)^5 \times (0.4)^5}{(0.6)^5 \times (0.4)^5 + (0.5)^{10}} = 0.45 \end{aligned}$$

这10次结果由硬币B投掷所得概率为：

$$\begin{aligned} & P(\text{选择硬币B投掷} | \text{硬币投掷结果}, \theta) \\ &= 1 - P(\text{选择硬币A投掷} | \text{硬币投掷结果}, \theta) \\ &= 0.55 \end{aligned}$$

期望最大化算法：二硬币投掷例子

表5.4 两个硬币在每一轮中被选择概率以及投掷正面/反面的次数

184						
表5.4 两个硬币在每一轮中被选择概率以及投掷正面/反面的次数						
轮次	选硬币 A 概率	选硬币 B 概率	硬币 A 为正面期望次数	硬币 A 为反面期望次数	硬币 B 为正面期望次数	硬币 B 为反面期望次数
1	0.45	0.55	2.25	2.25	2.75	2.75
2	0.80	0.20	7.24	0.80	1.76	0.20
3	0.73	0.27	5.87	1.47	2.13	0.53
4	0.35	0.65	1.41	2.11	2.59	3.89
5	0.65	0.35	4.53	1.94	2.47	1.07
合计			21.30	8.57	11.70	8.43

一旦得到了在第一轮试验中选择硬币A或硬币B投掷的概率，则可以根据这个概率来计算第一轮中硬币A和硬币B投掷正面的期望次数。比如，硬币A为正面期望次数为 $N_{\text{正面总次数}} \times P_{\text{选硬币A概率}} = 5 \times 0.45 = 2.25$ ，其他项计算类似。

期望最大化算法：二硬币投掷例子

期望最大化（M步骤，Maximization）：

在上面的计算中，通过初始化硬币A和硬币B投掷得到正面概率 $\hat{\theta}_A^{(0)}$ 和 $\hat{\theta}_B^{(0)}$ ，得到每一轮中选择硬币A和选择硬币B概率这一“隐变量”，进而可计算得到每一轮中硬币A和硬币B投掷正面次数。

在这些信息基础上，可更新得到硬币A和硬币B投掷为正面的概率，从而得到新的模型参数：

$$\hat{\theta}_A^{(1)} = \frac{21.30}{21.30+8.57} = 0.713 \quad \hat{\theta}_B^{(1)} = \frac{11.70}{11.70+8.43} = 0.581$$

接下来，可在新的概率值基础上继续计算每一轮投掷中选择硬币A或硬币B的概率，进而计算得到五轮中硬币A和硬币B投掷正面的总次数，从而得到硬币A和硬币B投掷为正面的更新概率值 $\hat{\theta}_A^{(2)}$ 和 $\hat{\theta}_B^{(2)}$ 。上述算法不断迭代，直至算法收敛，最终得到硬币A和硬币B投掷为正面的概率 $\theta = \{\theta_A, \theta_B\}$ 。

期望最大化算法：二硬币投掷例子

表5.5 两个硬币投掷结果为正面的概率迭代计算结果

表5.5 两个硬币投掷结果为正面的概率迭代计算结果

迭代次数	硬币A为正面次数	硬币A为反面次数	硬币B为正面次数	硬币B为反面次数	硬币A投掷正面概率 θ_A	硬币B投掷正面概率 θ_B
1	21.30	8.57	11.70	8.43	$\hat{\theta}_A^{(1)} = 0.713$	$\hat{\theta}_B^{(1)} = 0.581$
2	19.21	6.56	13.79	10.44	$\hat{\theta}_A^{(2)} = 0.745$	$\hat{\theta}_B^{(2)} = 0.569$
3	19.41	5.86	13.59	11.14	$\hat{\theta}_A^{(3)} = 0.768$	$\hat{\theta}_B^{(3)} = 0.550$
4	19.75	5.47	13.25	11.53	$\hat{\theta}_A^{(4)} = 0.783$	$\hat{\theta}_B^{(4)} = 0.535$
5	19.98	5.28	13.02	11.72	$\hat{\theta}_A^{(5)} = 0.791$	$\hat{\theta}_B^{(5)} = 0.526$
6	20.09	5.19	12.91	11.81	$\hat{\theta}_A^{(6)} = 0.795$	$\hat{\theta}_B^{(6)} = 0.522$
7	20.14	5.16	12.86	11.84	$\hat{\theta}_A^{(7)} = 0.796$	$\hat{\theta}_B^{(7)} = 0.521$
8	20.16	5.15	12.84	11.85	$\hat{\theta}_A^{(8)} = 0.796$	$\hat{\theta}_B^{(8)} = 0.520$
9	20.17	5.15	12.83	11.85	$\hat{\theta}_A^{(9)} = 0.797$	$\hat{\theta}_B^{(9)} = 0.520$
10	20.18	5.15	12.82	11.85	$\hat{\theta}_A^{(10)} = 0.797$	$\hat{\theta}_B^{(10)} = 0.520$

期望最大化算法：二硬币投掷例子

- 隐变量：

每一轮选择硬币A还是选择硬币B来完成10次投掷是一个隐变量，硬币A和硬币B投掷结果为正面的概率 $\theta = \{\theta_A, \theta_B\}$ 称为模型参数。

- 计算隐变量（EM 算法的 E 步）、最大化似然函数和更新模型参数（EM算法的M步）：

EM 算法使用迭代方法来求解模型参数 $\theta = \{\theta_A, \theta_B\}$ ：先初始化模型参数，然后计算得到隐变量（EM 算法的 E 步），接着基于观测投掷结果和当前隐变量值一起来最大化似然函数（即使得模型参数能够更好拟合观测结果），更新模型参数（EM算法的M步）。基于当前得到的模型参数，继续更新隐变量（EM算法的 E 步），然后继续最大化似然函数，更新模型参数（EM算法的M步）。以此类推，不断迭代下去，直到模型参数基本无变化，算法收敛。

期望最大化算法：三硬币投掷例子

- 假设有三枚质地材料不均匀的硬币（即每枚硬币投掷出现正反两面的概率不一定相等），这三枚硬币分别被标记为0, 1, 2。约定出现正面记为H（Head, 头），出现反面记为T（Tail, 尾），一次试验的过程如下：首先掷硬币0，如果硬币0投掷结果为H，则选择硬币1投掷三次，如果硬币0投掷结果为T，则选择硬币2投掷三次。观测结果中仅记录硬币1和硬币2的投掷结果，不出现硬币0的投掷结果。因为硬币0的投掷结果没有被记录，所以是未观测到的数据（隐变量）。
- 未观测数据取值集合记为 $C_Z = \{H, T\}$ ，观测数据取值集合记为 $C_X = \{HHH, TTT, HTT, THH, HHT, TTH, HTH, THT\}$ ，模型参数集合（三枚硬币分别出现正面的概率）为 $\Theta = \{\lambda, p_1, p_2\}$ 。在n次试验中，未观测到的数据序列记为z，观测到的数据序列记为x，观测到的数据序列中有h次为正面朝上，t次为反面朝上。

$$P(x, z|\Theta) = P(z|\Theta)P(x|z, \Theta)$$

$$P(z|\Theta) = \begin{cases} \lambda & \text{if } z = H \\ 1 - \lambda & \text{if } z = T \end{cases} \quad P(x|z, \Theta) = \begin{cases} p_1^h (1 - p_1)^t & \text{if } z = H \\ p_2^h (1 - p_2)^t & \text{if } z = T \end{cases}$$

期望最大化算法：三硬币投掷例子

- 在硬币0掷出正面后，选择硬币1投掷三次所得“反正反”这一结果的概率如下计算：

$$P(x = THT, z = H|\theta) = \lambda p_1(1 - p_1)^2$$

- 在硬币0掷出反面后，选择硬币2投掷三次所得“反正反”这一结果的概率如下计算：

$$P(x = THT, z = T|\theta) = (1 - \lambda)p_2(1 - p_2)^2$$

如果某次观测得到“反正反”这一投掷结果，则该投掷结果发生的概率如下计算：

$$\begin{aligned} P(x = THT|\theta) &= P(x = THT, z = H|\theta) + P(x = THT, z = T|\theta) \\ &= \lambda p_1(1 - p_1)^2 + (1 - \lambda)p_2(1 - p_2)^2 \end{aligned}$$

如果某次观测得到“反正反”这一投掷结果，这一结果是由硬币0投掷为正面（未观测的数据）所促发的概率计算如下：

$$\begin{aligned} P(z = H|x = THT, \theta) &= \frac{P(x = THT, z = H|\theta)}{P(x = THT|\theta)} \\ &= \frac{\lambda p_1(1 - p_1)^2}{\lambda p_1(1 - p_1)^2 + (1 - \lambda)p_2(1 - p_2)^2} \end{aligned}$$

这样，可以从观测数据来推测未观测数据的概率分布，即从硬币正面和反面观测结果来推测硬币0投掷为正面或反面这一隐变量。

期望最大化算法：三硬币投掷例子

情况1 为了了解未观测数据对模型参数求解带来的影响，这里先假设一个简单情况，在知道全部的数据（即包括上述的观测数据和未观测数据）情况下，来计算模型参数。

假设所观测的数据如下：($\langle HHH \rangle, H$), ($\langle TTT \rangle, T$), ($\langle HHH \rangle, H$), ($\langle TTT \rangle, T$), ($\langle HHH \rangle, H$)

则可如下来估计参数 θ 取值，其中

$$\hat{\lambda} = \frac{\text{count}(\text{硬币0掷出正面})}{\text{count}(\text{试验次数})} = \frac{3}{5}$$

$$\hat{p}_1 = \frac{\text{count}(\text{硬币1掷出正面})}{\text{count}(\text{硬币1掷出次数})} = \frac{9}{3 * 3} = 1$$

$$\hat{p}_2 = \frac{\text{count}(\text{硬币2掷出正面})}{\text{count}(\text{硬币2掷出次数})} = \frac{0}{2 * 3} = 0$$

期望最大化算法：三硬币投掷例子

情况2 在硬币0所掷出结果无法观测情况下，假设得到了如下部分不完全观测数据：

$(\langle HHH \rangle, _), (\langle TTT, _ \rangle), (\langle HHH \rangle, _), (\langle TTT \rangle, _), (\langle HHH \rangle, _)$

与前面的推导类似，如下计算这一观测结果是由硬币0投掷为正面或反面所引起的概率（求取期望步骤）。这一过程也就是计算隐变量取值（硬币0所投掷的结果）。

$$P(z = H|x = HHH, \theta) = \frac{\lambda p_1^3}{\lambda p_1^3 + (1 - \lambda)p_2^3}$$

$$P(z = T|x = HHH, \theta) = 1 - P(z = H|x = HHH, \theta)$$

$$P(z = H|x = TTT, \theta) = \frac{\lambda(1 - p_1)^3}{\lambda(1 - p_1)^3 + (1 - \lambda)(1 - p_2)^3}$$

$$P(z = T|x = TTT, \theta) = 1 - P(z = H|x = TTT, \theta)$$

在上述对隐变量概率估计基础上，则可估计参数 θ 的更新值（期望最大化步骤）：

$$\hat{\lambda} = \frac{3 * P(z = H|x = HHH, \theta) + 2 * P(z = H|x = TTT, \theta)}{5}$$

$$\hat{p}_1 = \frac{3 * 3 * P(z = H|x = HHH, \theta) + 0 * 2 * P(z = H|x = TTT, \theta)}{3 * 3 * P(z = H|x = HHH, \theta) + 3 * 2 * P(z = H|x = TTT, \theta)}$$

$$\hat{p}_2 = \frac{3 * 3 * P(z = T|x = HHH, \theta) + 0 * 2 * P(z = T|x = TTT, \theta)}{3 * 3 * P(z = T|x = HHH, \theta) + 3 * 2 * P(z = T|x = TTT, \theta)}$$

期望最大化算法：三硬币投掷例子

表5.6 三个硬币投掷中参数迭代更新过程

表5.6 三个硬币投掷中参数迭代更新过程					
迭代次数	λ	p_1	p_2	$P(z=H x=HHH, \theta)$	$P(z=H x=TTT, \theta)$
1	0.5000	0.4000	0.8000	0.1111	0.9643
2	0.4524	0.1474	0.9739	0.0029	1.0000
3	0.4017	0.0043	1.0000	0.0000	1.0000
4	0.4000	0.0000	1.0000	0.0000	1.0000

表5.7 三个硬币投掷中参数初始值被更改后的迭代更新过程

表5.7 三个硬币投掷中参数初始值被更改后的迭代更新过程					
迭代次数	λ	p_1	p_2	$P(z=H x=HHH, \theta)$	$P(z=H x=TTT, \theta)$
1	0.5000	0.5100	0.5000	0.5148	0.4849
2	0.5028	0.6143	0.5855	0.5388	0.449
3	0.5029	0.6428	0.5567	0.609	0.346
4	0.5038	0.7253	0.4728	0.7857	0.1255
5	0.5217	0.9037	0.2688	0.9765	0.0025
6	0.5869	0.9983	0.0342	1.0000	0.0000
7	0.6000	1.0000	0.0000	1.0000	0.0000

期望最大化算法：EM算法一般形式

对于 n 个相互独立的样本 $X = \{x_1, x_2, \dots, x_n\}$ 及其对应的隐变量 $Z = \{z_1, z_2, \dots, z_n\}$ ，在假设样本的模型参数为 θ 前提下，观测数据 x_i 的概率为 $P(x_i|\theta)$ ，完全数据 (x_i, z_i) 的似然函数为 $P(x_i, z_i|\theta)$ 。

在上面的表示基础上，优化目标为求解合适的 θ 和 Z 使得对数似然函数最大：

$$(\theta, Z) = \operatorname{argmax}_{\theta, Z} L(\theta, Z) = \operatorname{argmax}_{\theta, Z} \sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i|\theta)$$

但是，优化求解含有未观测数据 Z 的对数似然函数 $L(\theta, Z)$ 十分困难，EM算法不断构造对数似然函数 $L(\theta, Z)$ 的一个下界（E步骤），然后最大化这个下界（M步骤），以迭代方式逼近模型参数所能取得极大似然值。

期望最大化算法：EM算法一般形式

求解目标： $(\theta, Z) = \operatorname{argmax}_{\theta, Z} L(\theta, Z) = \operatorname{argmax}_{\theta, Z} \sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i | \theta)$

$$\sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i | \theta) = \sum_{i=1}^n \log \sum_{z_i} Q_i(z_i) \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$$

$$\geq \underbrace{\sum_{i=1}^n \sum_{z_i} Q_i(z_i) \log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}}_{\text{对数似然函数下界}}$$

在上式中， $Q_i(z_i)$ 是隐变量分布，满足： $\sum_{z_i} Q_i(z_i) = 1$ ($0 \leq Q_i(z_i)$)。

上述不等式中使用了Jensen 不等式 (Jensen's inequality)。对于凹函数 f ，Jensen 不等式使得下面不等式成立：

$$\log(E(f)) \geq E(\log(f)), \text{ 其中 } E(f) = \sum_i \lambda_i f_i, \lambda_i \geq 0, \sum_i \lambda_i = 1$$

期望最大化算法：EM算法一般形式

求解目标： $(\theta, Z) = \operatorname{argmax}_{\theta, Z} L(\theta, Z) = \operatorname{argmax}_{\theta, Z} \sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i | \theta)$

令 $f_i = \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 和 $\lambda_i = Q_i(z_i)$ ，则根据Jensen 不等式的定义，可将 $\frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 视为第 i 个样本，

$Q_i(z_i)$ 为第 i 个样本的权重。按照这样的约定，可得到如下式子：

$$E \left(\log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)} \right) = \sum_{z_i} \underbrace{Q_i(z_i)}_{\text{权重}} \underbrace{\log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}}_{\text{样本值}}$$

于是，为了最大化 $\sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i | \theta)$ 这一对数似然函数，只需最大化其下界

$\sum_{i=1}^n \sum_{z_i} Q_i(z_i) \log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 。这个下界实际上是 $\log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 的加权求和。由于权重 $Q_i(z_i)$ 累加之

和为1，因此 $\sum_{z_i} Q_i(z_i) \log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 就是 $\log \frac{P(x_i, z_i | \theta)}{Q_i(z_i)}$ 的加权平均，也就是所谓的期望，**这就是EM**

算法中Expectation这一单词的来源。于是，EM算法就是不断最大化这一下界（M步骤），从而通过迭代的方式逼近模型参数的极大似然估计值。

期望最大化算法：EM算法一般形式

求解目标： $(\theta, Z) = \operatorname{argmax}_{\theta, Z} L(\theta, Z) = \operatorname{argmax}_{\theta, Z} \sum_{i=1}^n \log \sum_{z_i} P(x_i, z_i | \theta)$

显然，当 θ 取值给定后，对数似然函数的下界只与 $P(x_i, z_i)$ 和 $Q_i(z_i)$ 相关。于是，通过调整 $P(x_i, z_i)$ 和 $Q_i(z_i)$ 取值，使得似然函数下界不断逼近似然函数真实值。那么，当不等式取等式时，调整后的似然函数下界等于似然函数真实值。当每个样本取值均相等时（也就是每个样本取值为同一个常数），Jensen 不等式中的等式成立。于是令 $\frac{P(x_i, z_i | \theta)}{Q_i(z_i)} = c$ （ c 为常数），得到 $P(x_i, z_i | \theta) = c Q_i(z_i)$ 。由于 $\sum_{z_i} Q_i(z_i) = 1$ ，可知 $\sum_{z_i} P(x_i, z_i | \theta) = c$ 。

于是， $Q_i(z_i) = \frac{P(x_i, z_i | \theta)}{c} = \frac{P(x_i, z_i | \theta)}{\sum_{z_i} P(x_i, z_i | \theta)} = \frac{P(x_i, z_i | \theta)}{P(x_i | \theta)} = P(z_i | x_i, \theta)$ 。也就是说，只要 $Q_i(z_i) = P(z_i | x_i, \theta)$ ，就能够保证对数似然函数最大值与其下界相等。

从上面的阐述可知，固定参数 θ 后，只要从观测数据 x_i 和参数 θ 出发，令 $Q_i(z_i) = P(z_i | x_i, \theta)$ ，则可以得到对数似然函数最大值的下界，这就是EM算法中的E步骤。然后，固定 $Q_i(z_i)$ ，调整 θ ，再去极大化对数似然函数最大值的下界，这就是EM算法的M步骤。

期望最大化算法：EM算法一般形式

算法5.3 EM算法

算法5.3 EM算法

输入：观测所得样本数据 X 及其对应的隐变量 Z （即无法观测数据）、联合分布 $P(X, Z|\theta)$ 。

输出：模型参数 θ 。

算法步骤：

1. 初始化参数取值 θ^0 。

2. 求取期望步骤（E步骤）：计算 $Q(\theta|\theta') = \sum_{i=1}^n \sum_{z_i} P(z_i|x_i, \theta) \log P(x_i, z_i|\theta)$ 。

其含义是对数似然函数 $\log P(x_i, z_i|\theta)$ 在已观测数据 X 和当前参数 θ' 下去估计隐变量 Z 的条件概率分布 $P(z_i|x_i, \theta)$ 。

3. 期望最大化步骤（M步骤）： $\theta^{t+1} = \arg\max_{\theta} Q(\theta|\theta')$ 。

4. 重复第2步和第3步，直到收敛。

在算法5.3中，固定上一步所得参数 θ^t 情况下，通过计算 Z 的后验概率来得到能够逼近真实最大似然估计的下界，与算法中的求取期望步骤一致。在期望最大化步骤中，通过最大化对数似然函数，可以得到新的参数 θ^{t+1} ，即：

$$\theta^{t+1} = \arg\max_{\theta} \sum_{i=1}^n \sum_{z_i} Q_i(z_i) \log \frac{P(x_i, z_i|\theta)}{Q_i(z_i)}$$

因为 $Q_i(z_i) = p(z_i|x_i, \theta)$ 是由上一步的 θ^t 估计出的，与下一步要优化的 θ^{t+1} 无关，所以上式等价于：

$$\begin{aligned} & \theta^{t+1} \\ &= \arg\max_{\theta} \sum_i \sum_{z_i} Q_i(z_i) \log p(x_i, z_i|\theta) - Q_i(z_i) \log Q_i(z_i) \\ &= \arg\max_{\theta} \sum_i \sum_{z_i} Q_i(z_i) \log p(x_i, z_i|\theta) \end{aligned}$$

广义EM算法中，E步骤是固定参数来优化隐变量分布，M步骤是固定隐变量分布来优化参数，两者不同交替迭代。至此，证明了EM算法能够通过不断最大化下界来逼近最大似然估计值。

期望最大化算法：EM算法一般形式

算法5.3 EM算法

算法5.3 EM算法

输入：观测所得样本数据 X 及其对应的隐变量 Z （即无法观测数据）、联合分布 $P(X, Z|\theta)$ 。
输出：模型参数 θ 。

算法步骤：

1. 初始化参数取值 θ^0 。

2. 求取期望步骤（E步骤）：计算 $Q(\theta|\theta') = \sum_{i=1}^n \sum_{z_i} P(z_i|x_i, \theta) \log P(x_i, z_i|\theta)$ 。

其含义是对数似然函数 $\log P(x_i, z_i|\theta)$ 在已观测数据 X 和当前参数 θ' 下去估计隐变量 Z 的条件概率分布 $P(z_i|x_i, \theta)$ 。

3. 期望最大化步骤（M步骤）： $\theta^{t+1} = \arg\max_{\theta} Q(\theta|\theta')$ 。

4. 重复第2步和第3步，直到收敛。

假设 θ^{t+1} 和 θ^t 是两个相邻的迭代更新得到的参数值，如果能证明 $L(\theta^t) \leq L(\theta^{t+1})$ ，即可认为EM算法能够让似然单调增长。下面给出简略的证明：

$$L(\theta^{t+1}) \geq \sum_i \sum_{z_i} Q_i(z_i) \log \frac{p(x_i, z_i|\theta^{t+1})}{Q_i(z_i)}$$
$$\Rightarrow l(\theta^{t+1}) \geq \sum_i \sum_{z_i} Q_i^t(z_i) \log \frac{p(x_i, z_i|\theta^{t+1})}{Q_i^t(z_i)} \quad (\text{特殊化})$$

因为 $\theta^{t+1} = \arg\max_{\theta} \sum_i \sum_{z_i} Q_i(z_i) \log \frac{p(x_i, z_i|\theta)}{Q_i(z_i)}$ ，即对于 θ^t （事实上可以是任意的 θ ）都有下面的式子成立：

$$\sum_i \sum_{z_i} Q_i^t(z_i) \log \frac{p(x_i, z_i|\theta^{t+1})}{Q_i^t(z_i)}$$
$$\geq \sum_i \sum_{z_i} Q_i^t(z_i) \log \frac{p(x_i, z_i|\theta^t)}{Q_i^t(z_i)}$$
$$= L(\theta^t)$$

至此，证明了 $L(\theta^{t+1}) \geq L(\theta^t)$ ，即EM算法能够保证似然度取值单调增长，**即EM算法能够稳定收敛**

无监督学习

无监督学习从非标注样本出发来学习数据的分布，这是一个异常困难的工作。由于无法利用标注信息，因此在无监督学习只能利用假设数据具有某些结构来进行学习。正如拉普拉斯所言“概率论只不过是把常识用数学公式表达了出来”，无监督学习就是把预设数据具有某种结构作为一种“知识”来指导模型的学习。

作业内容

K-means异常检测

异常值检测（outlier detection）是一种数据挖掘过程，用于发现数据集中的异常值并确定异常值的详细信息。

当前数据容量大、数据类型多样、获取数据速度快；但是数据也比较复杂，数据的质量有待商榷；而数据容量大意味着手动标记异常值成本高、效率低下；因此能够自动检测异常值至关重要。

自动异常检测具有广泛的应用，例如信用卡欺诈检测、系统健康监测、故障检测以及传感器网络中的事件检测系统等。

实验要求

- (1)了解 KMeans、PCA 算法，了解算法的基本原理
- (2)使用Numpy自行编写 KMeans 算法完成异常点检测
- (3)并将完成的KMeans 算法代码实现以及实现思路贴在实验报告中提交

布置日期：2022年4月11日；提交作业：2022年4月22日晚上12点前