数据科学导论

第三章：数据预处理

周帆

统计与数据科学学院

数据预处理

- 数据预处理：概述
- 数据清洗
- 数据集成
- 数据规约与变换
- 降维
- 总结

什么是数据预处理？——主要任务

- **数据清洗**
  - 处理缺失数据，平滑噪声数据，识别或删除异常值，以及解决不一致问题
- **数据集成**
  - 多个数据库, 数据立方或文件的集成
- **数据规约**
  - 维度规约（降维）
  - 数量规约
  - 数据压缩
- **数据变换和数据离散化**
  - 标准化
  - 概念层次生成

为什么要数据预处理？——数据质量问题

- **数据质量的度量：多维度视角**
  - 准确性：正确或错误，准确与否
  - 完整性：未记录，不可用，……
  - 一致性：有些修改了而有些没有，……
  - 时效性：及时更新?
  - 可信度：数据的正确性有多可靠?
  - 可解释性：数据是否易于理解?

数据预处理

- 数据预处理：概述
- 数据清洗
- 数据集成
- 数据规约与变换
- 降维
- 总结

数据清洗

- 真实世界中的数据很**脏**：大量潜在的不正确数据，例如仪器故障, 人为或计算机错误以及传输错误
  - 不完整：缺乏属性值，缺乏感兴趣的特定属性，或者只包含汇总数据
    - 例如，*职业=“”*（缺失数据）
  - 有噪声：包含噪声, 错误值或离群点
    - 例如，*薪资=“-10”*（错误）
  - 不一致：包含代码或名称中的矛盾，例如，
    - *年龄=“42”，生日=“03/07/2010”*
    - 过去的评级“1，2，3”，现在的评级“A，B，C”
    - 重复记录之间的矛盾
  - 蓄意的（例如，*伪装丢失数据*）
    - 1月1日是每个人的生日?

数据不完整(缺失)

- 数据并不总是可用的
  - 例如，许多元组对若干属性没有记录值，比如销售数据中的客户收入
- 数据缺失可能的原因：
  - 设备故障
  - 与其他记录数据不一致而被删除
  - 因误解未录入数据
  - 某些数据在输入时可能被认为是不重要的
  - 没有记录数据的历史或变化
- 缺失的数据可能需要进行推断

如何处理缺失的数据?

- 忽略元组：通常在类标签缺失时执行(在进行分类时) -当每个属性缺失值的百分比变化很大时无效
- 手动填写缺失值：繁琐+不可行?
- 自动填写
- 一个全局常量：例如，“未知”，一个新的类?!
- 属性均值
- 属于同一类的所有样本的属性均值：更聪明
- **更好的做法：基于推理，如决策树**

噪声数据

- **噪声：**被测变量的随机误差或方差
- **不正确属性值**可能的原因：
  - 数据采集仪器故障
  - 数据输入问题
  - 数据传输问题
  - 技术限制
  - 命名习惯不一致
- **其他数据问题**
  - 重复记录
  - 不完整数据
  - 不一致数据

如何处理有噪声的数据?

- 分箱
  - 首先对数据进行排序，并将其划分到(等频的)箱
  - 然后可以**按箱均值平滑，按箱中值平滑，按箱边界平滑**等
- 回归
  - 通过将数据拟合到回归函数中平滑
- 聚类
  - 检测和去除异常值
- 半监督：结合计算机检查和人工检查
  - 检测可疑值并由人工检查(例如，处理可能的异常值)

数据预处理

- 数据预处理：概述
- 数据清洗
- 数据集成
- 数据规约与变换
- 降维
- 总结

数据集成

- 数据集成
  - 将来自多个数据源的数据组合到一个一致的存储中
- 模式集成：例如，A.cust-id ≡ B.cust-#
  - 集成来自不同来源的元数据
- **实体标识：**
  - 从多个数据源中识别真实世界的实体；例如，Bill Clinton =William Clinton
- 检测和解决数据值冲突
  - 对于同一个现实世界的实体，来自不同来源的属性值是不同的
  - 可能的原因：不同的表示，不同的尺度，例如，公制单位与英制单位

数据集成中的冗余处理

- 在集成多个数据库时，经常会出现冗余数据
  - 对象标识：相同的属性或对象在不同的数据库中可能有不同的名称
  - 可派生数据：一个属性可能是另一个表中的“派生”属性；例如，年收入
- **通过相关分析和协方差分析或许可以检测出冗余属性**
- 仔细整合来自多个来源的数据可能有助于减少/避免冗余和不一致，并提高数据挖掘速度和质量

相关分析(分类数据)

- X^2(卡方)检验：

观测数

$$\chi^2 = \sum_{i=1}^{n} \frac{(O_i - E_i)^2}{E_i}$$

期望数

- 原假设：两个分布是相互独立的
- 对X^2值贡献最大的单元是那些实际数与期望数相差很大的单元
  - X^2值越大，变量之间的相关性越强
- 注：相关性并不意味着因果关系
  - 一个城市的医院数量和汽车盗窃数量是相关的
  - 与两者都有因果关系的第三变量： 人口数

卡方计算：一个例子

	下棋	不下棋	总和（行）
喜欢科幻小说	250（90）	200（360）	450
不喜欢科幻小说	50（210）	1000（840）	1050
总和（列）	300	1200	1500

如何推导出90?

450/1500 * 300 = 90

观测数

$$\chi^2 = \sum_{i}^{n} \frac{(O_i - E_i)^2}{E_i}$$

期望数

卡方计算：一个例子

	下棋	不下棋	总和（行）
喜欢科幻小说	250（90）	200（360）	450
不喜欢科幻小说	50（210）	1000（840）	1050
总和（列）	300	1200	1500

如何推导出90?

450/1500 * 300 = 90

- X^2(卡方)计算（括号内的数字是根据两类数据分布计算的期望数）

$$\chi^2 = \frac{(250-90)^2}{90} + \frac{(50-210)^2}{210} + \frac{(200-360)^2}{360} + \frac{(1000-840)^2}{840} = 507.93$$

我们可以在置信水平0.001上拒绝独立的原假设

- 结果表明，“喜欢科幻小说”和“下棋”在组中具有相关性

两变量的协方差

- 两个变量 X_1 和 X_2 之间的协方差

$$\sigma_{12} = E[(X_1 - \mu_1)(X_2 - \mu_2)] = E[X_1X_2] - \mu_1\mu_2 = E[X_1X_2] - E[X_1]E[X_2]$$

其中 $\mu_1 = \mathrm{E}[X_1]$ 是X_1的均值或**期望值**； μ_2 类似

- X_1 和 X_2 之间的样本协方差：$\hat{\sigma}_{12} = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)(x_{i2} - \hat{\mu}_2)$
- 样本协方差是样本方差的推广：

$$\hat{\sigma}_{11} = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)(x_{i1} - \hat{\mu}_1) = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)^2 = \hat{\sigma}_1^2$$

- 正协方差：如果 $\sigma_{12} > 0$
- 负协方差：如果 $\sigma_{12} < 0$
- 独立性：如果 X_1 和 X_2 是独立的，则 $\sigma_{12} = 0$，反之则不成立
  - 有些随机变量间可能协方差为0，但是并不独立

两变量的协方差

- 两个变量 X_1 和 X_2 之间的协方差

$$\sigma_{12} = E[(X_1 - \mu_1)(X_2 - \mu_2)] = E[X_1X_2] - \mu_1\mu_2 = E[X_1X_2] - E[X_1]E[X_2]$$

其中 $\mu_1 = \mathrm{E}[X_1]$ 是X_1的均值或**期望值**； μ_2 类似

- X_1 和 X_2 之间的样本协方差：$\hat{\sigma}_{12} = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)(x_{i2} - \hat{\mu}_2)$
- 样本协方差是样本方差的推广：

$$\hat{\sigma}_{11} = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)(x_{i1} - \hat{\mu}_1) = \frac{1}{n}\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)^2 = \hat{\sigma}_1^2$$

- 正协方差：如果 $\sigma_{12} > 0$
- 负协方差：如果 $\sigma_{12} < 0$
- 独立性：如果 X_1 和 X_2 是独立的，则 $\sigma_{12} = 0$，反之则不成立
  - 有些随机变量间可能协方差为0，但是并不独立
  - 只有在一些额外的假设下（例如，数据服从多变量正态分布），协方差为0才意味着独立性

示例：协方差计算

- 假设两只股票 X_1 和 X_2 在一周内的值如下：
  - (2, 5), (3, 8), (5, 10), (4, 11), (6, 14)
- 问题：如果这些股票受到相同行业趋势的影响，它们的价格会一起上涨还是一起下跌？

示例：协方差计算

- 假设两只股票 X_1 和 X_2 在一周内的值如下：
  - (2, 5), (3, 8), (5, 10), (4, 11), (6, 14)
- 问题：如果这些股票受到相同行业趋势的影响，它们的价格会一起上涨还是一起下跌?
- 协方差公式：

$$\sigma_{12} = E[(X_1 - \mu_1)(X_2 - \mu_2)] = E[X_1X_2] - \mu_1\mu_2 = E[X_1X_2] - E[X_1]E[X_2]$$

- 其计算可简化为：
  - $E(X_1) = (2 + 3 + 5 + 4 + 6)/5 = 20/5 = 4$
  - $E(X_2) = (5 + 8 + 10 + 11 + 14)/5 = 48/5 = 9.6$
  - $\sigma_{12} = (2 \times 5 + 3 \times 8 + 5 \times 10 + 4 \times 11 + 6 \times 14)/5 - 4 \times 9.6 = 4$
- 因此，如果 $\sigma_{12} > 0$ ， X_1 和 X_2 一起上升

两个数值变量之间的相关性

- 两个变量 X_1 和 X_2 之间的**相关性**是标准协方差，是用每个变量的标准差对协方差进行归一化得到的

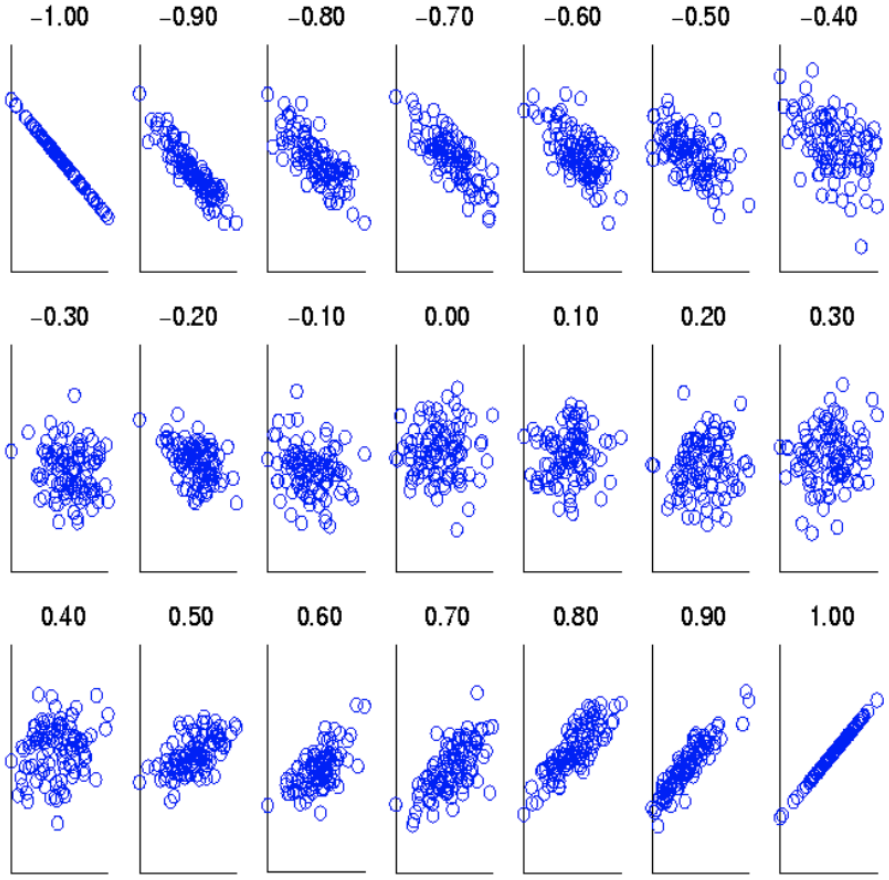
$$\rho_{12} = \frac{\sigma_{12}}{\sigma_1 \sigma_2} = \frac{\sigma_{12}}{\sqrt{\sigma_1^2 \sigma_2^2}}$$

- 两个属性 X_1 和 X_2 的样本相关性：$\hat{\rho}_{12} = \frac{\hat{\sigma}_{12}}{\hat{\sigma}_1 \hat{\sigma}_2} = \frac{\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)(x_{i2} - \hat{\mu}_2)}{\sqrt{\sum_{i=1}^{n}(x_{i1} - \hat{\mu}_1)^2 \sum_{i=1}^{n}(x_{i2} - \hat{\mu}_2)^2}}$

  其中，n 为元组个数， μ_1 和 μ_2分别为 X_1 和 X_2 的均值， σ_1 和 σ_2分别是 X_1 和 X_2 的标准差

- 若$\rho_{12} > 0$：A 和 B 正相关（ X_1的值随 X_2的值的增加而增加 ）
  - 值越大，相关性越强
- 若$\rho_{12} = 0$：独立（与协方差讨论中的假设相同）
- 若$\rho_{12} < 0$：负相关

相关系数变化的可视化



- 相关系数取值范围：[-1, 1]
- 一组散点图显示了一组点及其相关系数从 -1 到 1 的变化

协方差矩阵

- 两个变量 X_1 和 X_2 的方差和协方差信息可以概括为 2×2 的协方差矩阵：

$$\sum = E[(X - \mu)(X - \mu)^T] = E[\begin{pmatrix} X_1 - \mu_1 \\ X_2 - \mu_2 \end{pmatrix}(X_1 - \mu_1 \quad X_2 - \mu_2)]$$

$$= \begin{pmatrix} E[(X_1 - \mu_1)(X_1 - \mu_1)] & E[(X_1 - \mu_1)(X_2 - \mu_2)] \\ E[(X_2 - \mu_2)(X_1 - \mu_1)] & E[(X_2 - \mu_2)(X_2 - \mu_2)] \end{pmatrix}$$

$$= \begin{pmatrix} \sigma_1^2 & \sigma_{12} \\ \sigma_{21} & \sigma_2^2 \end{pmatrix}$$

- 将之推广到 d-维，有

$$D = \begin{pmatrix} x_{11} & x_{12} & \cdots & x_{1d} \\ x_{21} & x_{22} & & x_{2d} \\ \vdots & & \ddots & \vdots \\ x_{d1} & x_{d2} & \cdots & x_{dd} \end{pmatrix} \quad \sum = E[(X - \mu)(X - \mu)^T] = \begin{pmatrix} \sigma_1^2 & \sigma_{12} & \cdots & \sigma_{1d} \\ \sigma_{21} & \sigma_2^2 & & \sigma_{2d} \\ \vdots & & \ddots & \vdots \\ \sigma_{d1} & \sigma_{d2} & \cdots & \sigma_d^2 \end{pmatrix}$$

数据预处理

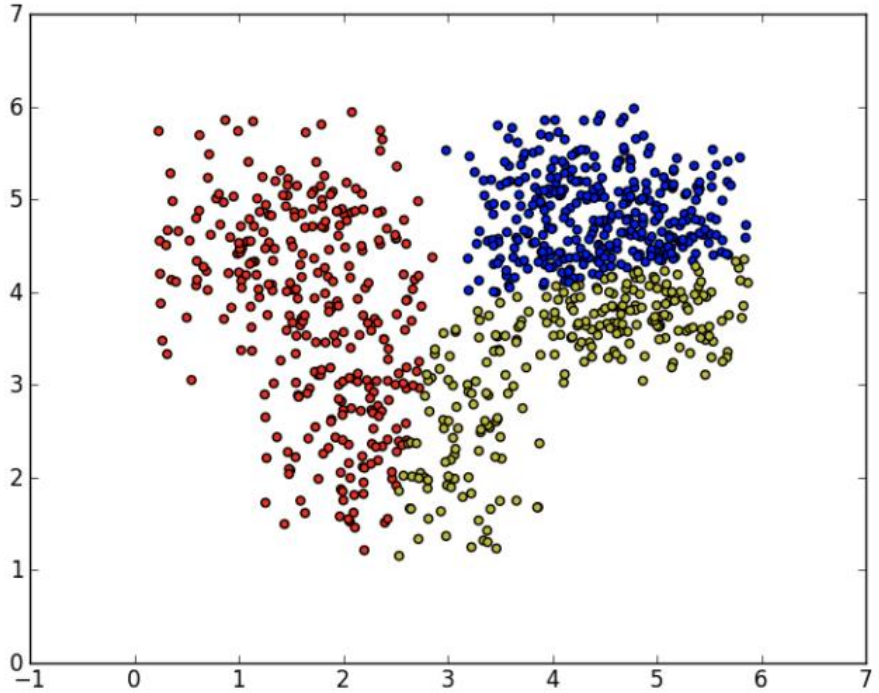
- 数据预处理：概述
- 数据清洗
- 数据集成
- 数据规约与变换
- 降维
- 总结

数据规约

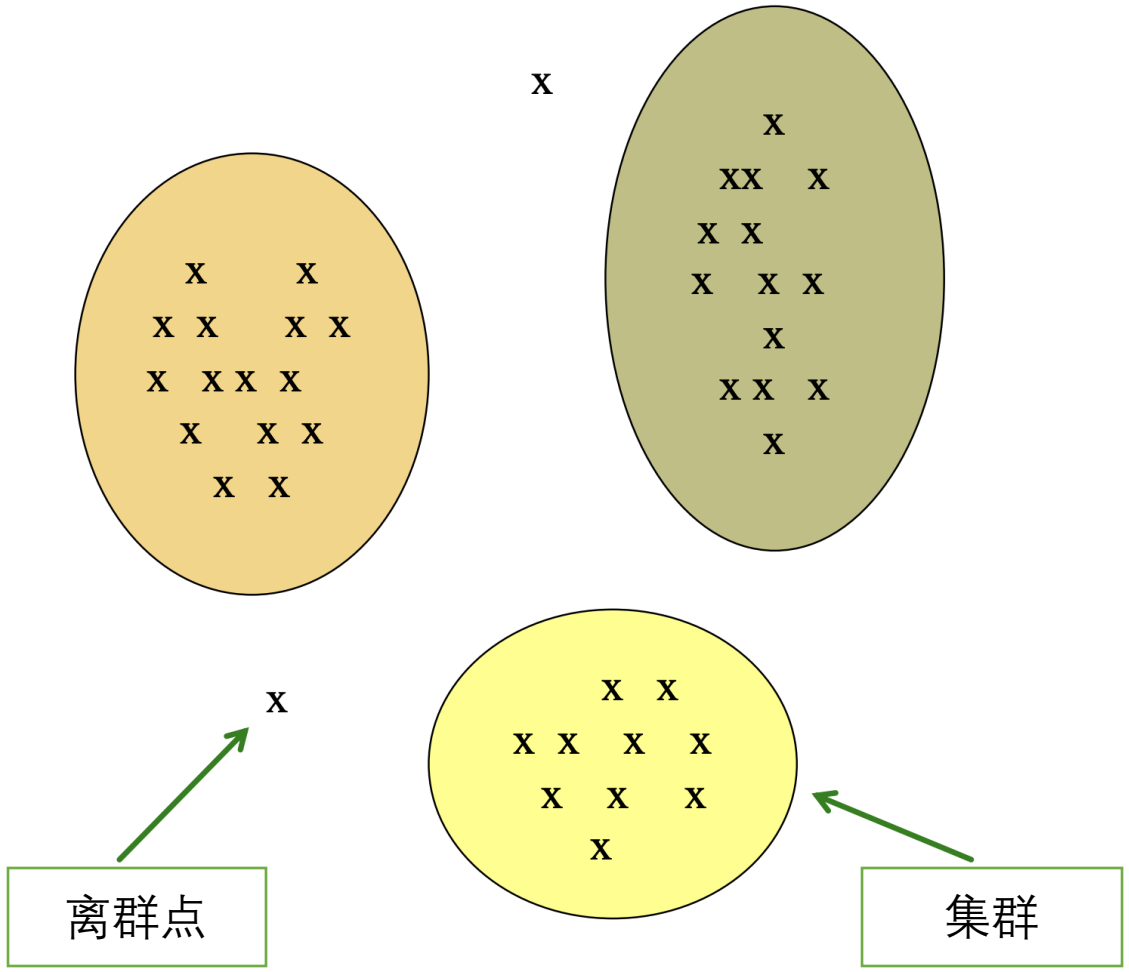
- **数据规约：**
  - 获得数据集的简化表示
    - 体量小得多，但却产生 *几乎* 相同的分析结果
  - 为什么要进行数据规约？——数据库/数据仓库可能可以存储 TB 级的数据
  - 在完整的数据集上进行复杂的分析可能需要很长的时间
    - **数据规约的方法**（也称为*数据规模规约* 或 *数量规约*）
  - 回归和对数线性模型
  - 直方图，聚类，抽样
  - 数据立方合计
  - 数据压缩

聚类

- 基于相似性将数据划分为簇，仅存储簇的表示(例如，质心和直径)
- 如果数据本身是成簇的（或成群的），那么聚类可能非常有效，否则聚类可能无效
- 可以分层聚类并且存储在多维索引树结构中
- 聚类定义和聚类算法有很多选择



示例: 群集和离群点



示例: 群集和离群点

- 一个由20亿个“天空物体”组成的集合，通过7个维度的特征来表示天体
- 问题：将类似的天体聚集在一起，可以得到例如星系，附近的恒星，类星体等

聚类问题：音乐CD

- 直观地说：音乐可以分成不同的类别，顾客偏好一部分流派
  - 但是类别到底是什么?
- 用一群购买该CD的客户表示CD
- 类似的CD有类似的客户群，反之亦然

聚类问题：音乐CD

包含所有CDs的空间：

- 针对每个客户，考虑一个一维空间
  - 一个维度中的值只可能为 0 或 1
  - 一个CD就是这个空间 $(x_1, x_2, \ldots, x_d)$ 中的一个“点”，其中 $x_i = 1$ 当且仅当第 i 个客户买了这个CD
- 对于亚马逊来说，维度数是数千万的
- **任务**：查找类似 CD 的集群

聚类问题：文档

寻找主题：

- 用向量 $(x_1, x_2, \ldots, x_d)$ 表示文档，其中 $x_i = 1$ 当且仅当第 i 个单词(以某种顺序)出现在这个文档中

- **具有相似词集的文档可能是关于相同主题的**

不同的距离

- **当我们把文档看作是一组单词或词条时，我们有一个选择：**
  - **作为向量：** 用**余弦距离**度量相似性
  - **作为集合：** 用**Jaccard距离**度量相似性
  - **作为点：** 用**欧几里得距离**度量相似性

为什么聚类很难?

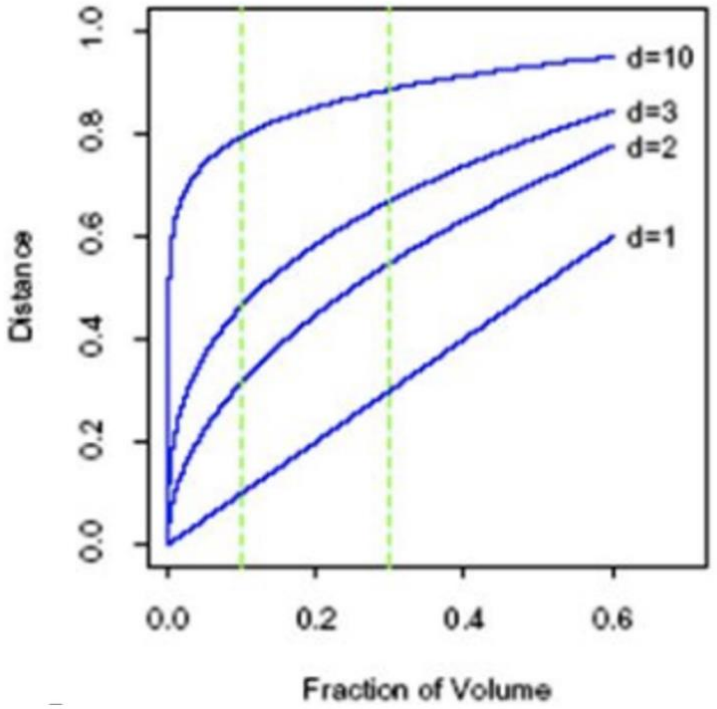
- 二维聚类看起来很简单
- 对少量数据进行聚类看起来很容易

- 许多应用程序涉及的不是 2 个维度，而是 10 个或者 10,000 个维度
- **高维空间看起来不一样：**

  几乎所有的点与彼此都相距很远 → **维度诅咒!**

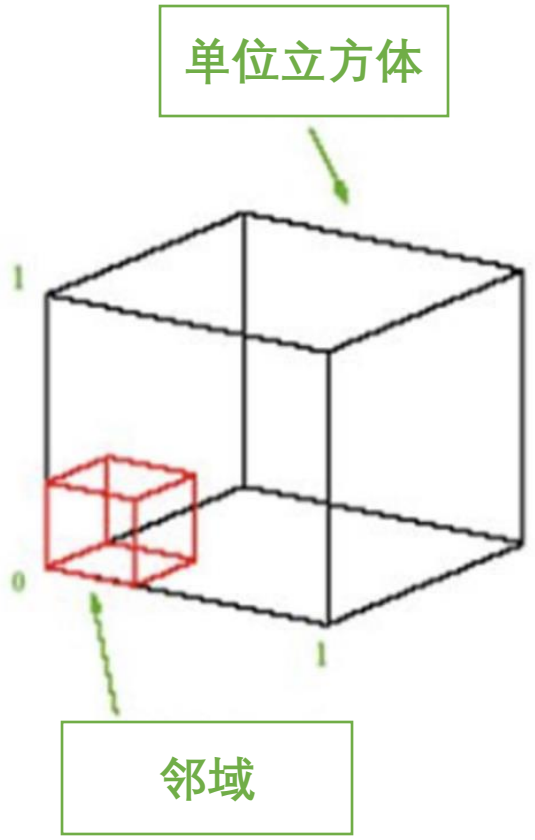
维度诅咒

- 假设数据位于 $[0,1]^d$ 中
- 为了捕获代表超立方体体积的一小部分 s 的邻域，我们需要边缘长度为 $s^{1/d}$
  - $s = 0.1, d = 10, s^{1/d} = 0.8$
  - $s = 0.01, d = 10, s^{1/d} = 0.63$



维度诅咒

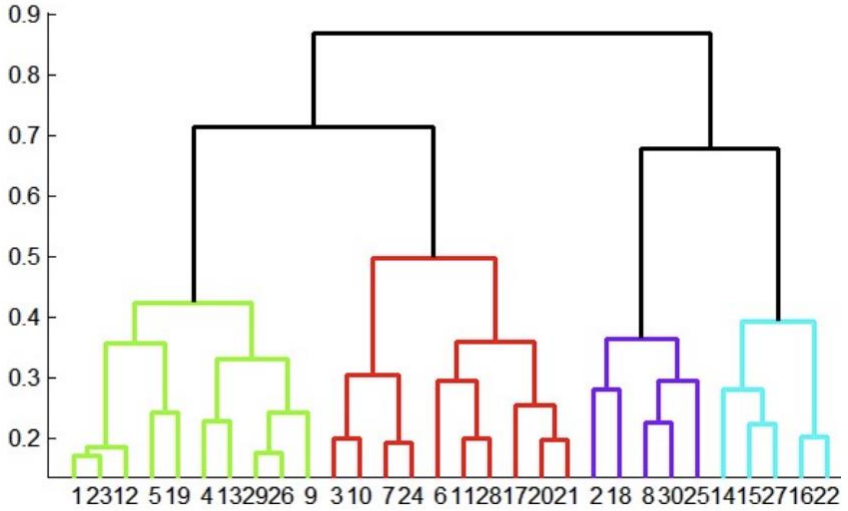
- 邻域不再是局部的
- 所有的点彼此距离都很远
- 高维空间中的点都是孤立的
  - 边长 r=0.1 的超立方体体积为 0.1^d
  - 当 d 增长时，它很快变得很小以至于捕获该点的概率很小很小



概述：聚类策略

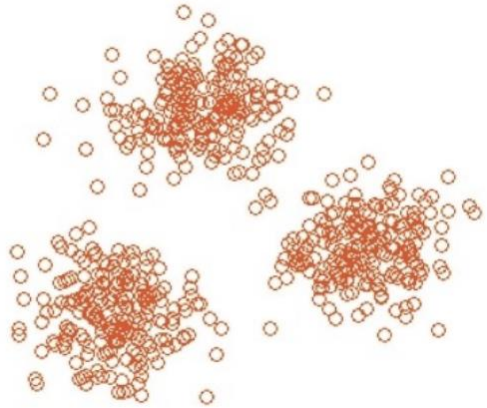
层次聚类：

- **聚合**(自下而上)：
  - 最初，每个点都是一个簇
  - 反复将两个“最近”的簇组合成一个簇
- **分裂**(自上而下)：
  - 从一个集群开始，然后递归地拆分它



点分配：

- 维护一组集群
- 点属于“最近的”聚类



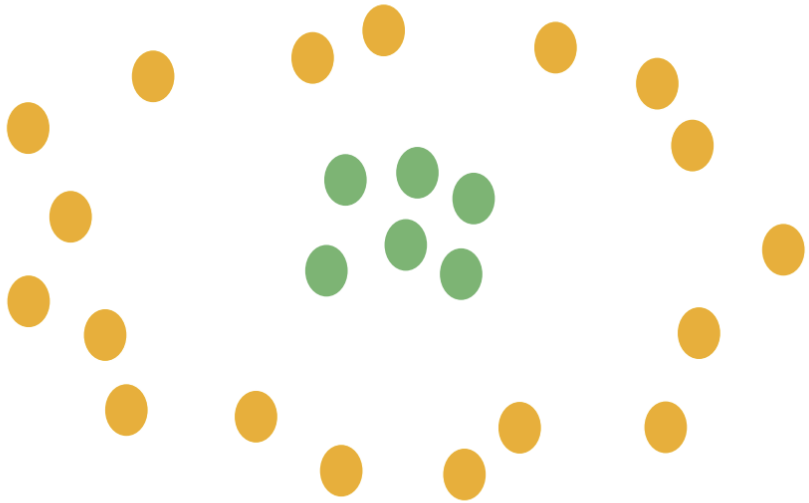
概述：聚类策略

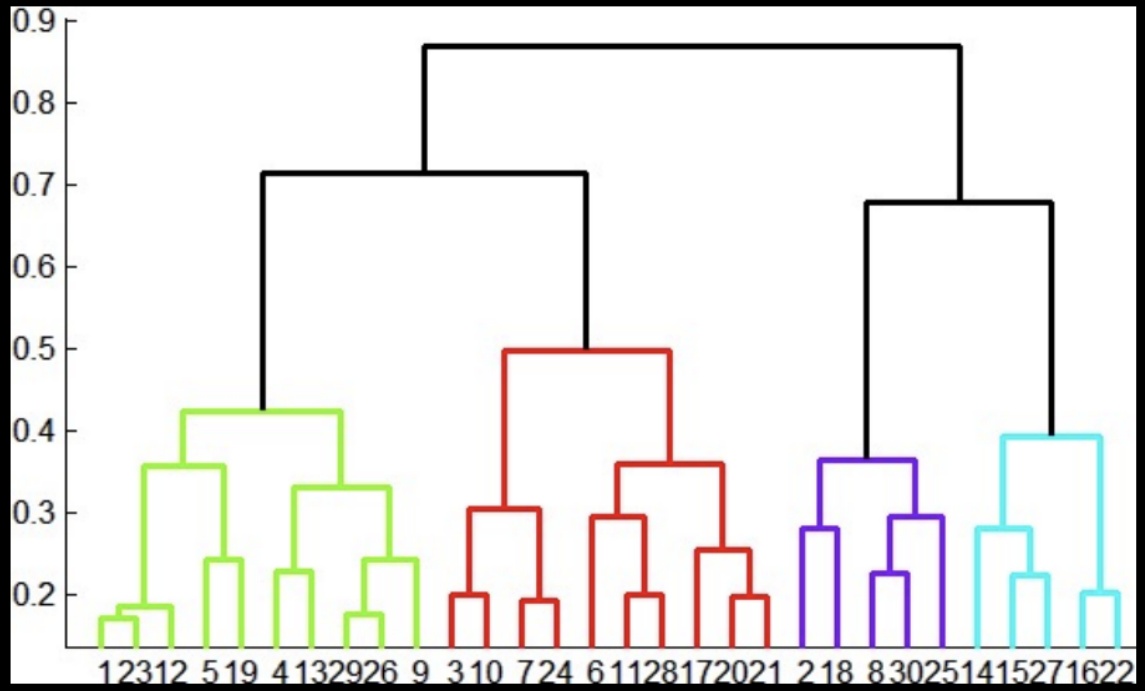
- 空间是欧式的还是非欧的?
- 在欧几里得空间中:
  - 点是实数的向量，也就是坐标
  - 我们可以把一组点汇总为它们的平均值，我们称之为*质心*。
  - 距离测量: L2范数，L1范数
- 在非欧空间中:
  - 没有位置和质心的概念
  - 我们以不同的方式进行点汇总
  - 距离测量: Jaccard，Hamming，cosine

层次聚类 VS 点分配

- **当簇是漂亮的凸形状时，点分配是好的:**

- **当形状很奇怪的时候，层次聚类更好:**
  - 注意：两个簇本质上有相同的质心。



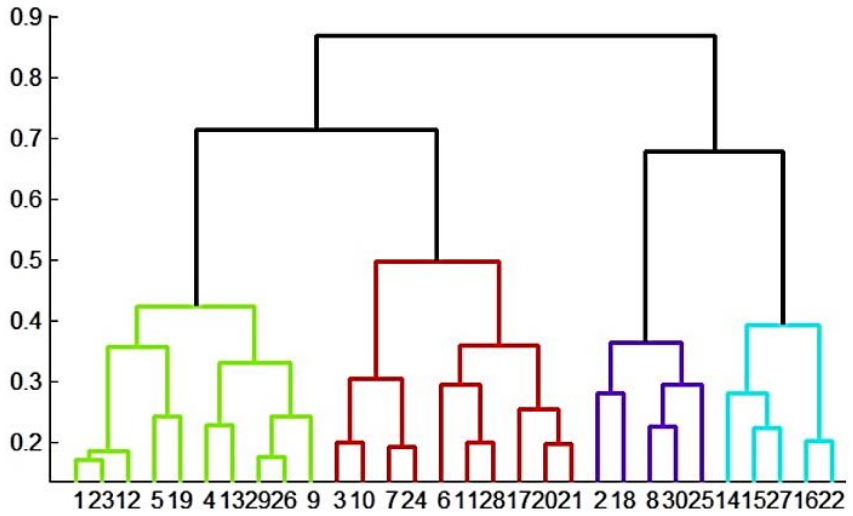

0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
12312 5 19 4 13 29 26 9 3 10 7 24 6 11 28 17 20 21 2 18 8 30 25 14 15 27 16 22

层次聚类

层次聚类

- **关键操作：**

  重复合并两个“最近”的集群



- **三个重要问题：**
  - 1)如何表示集群?
  - 2)如何确定聚类的接近度?
  - 3)何时停止合并集群?

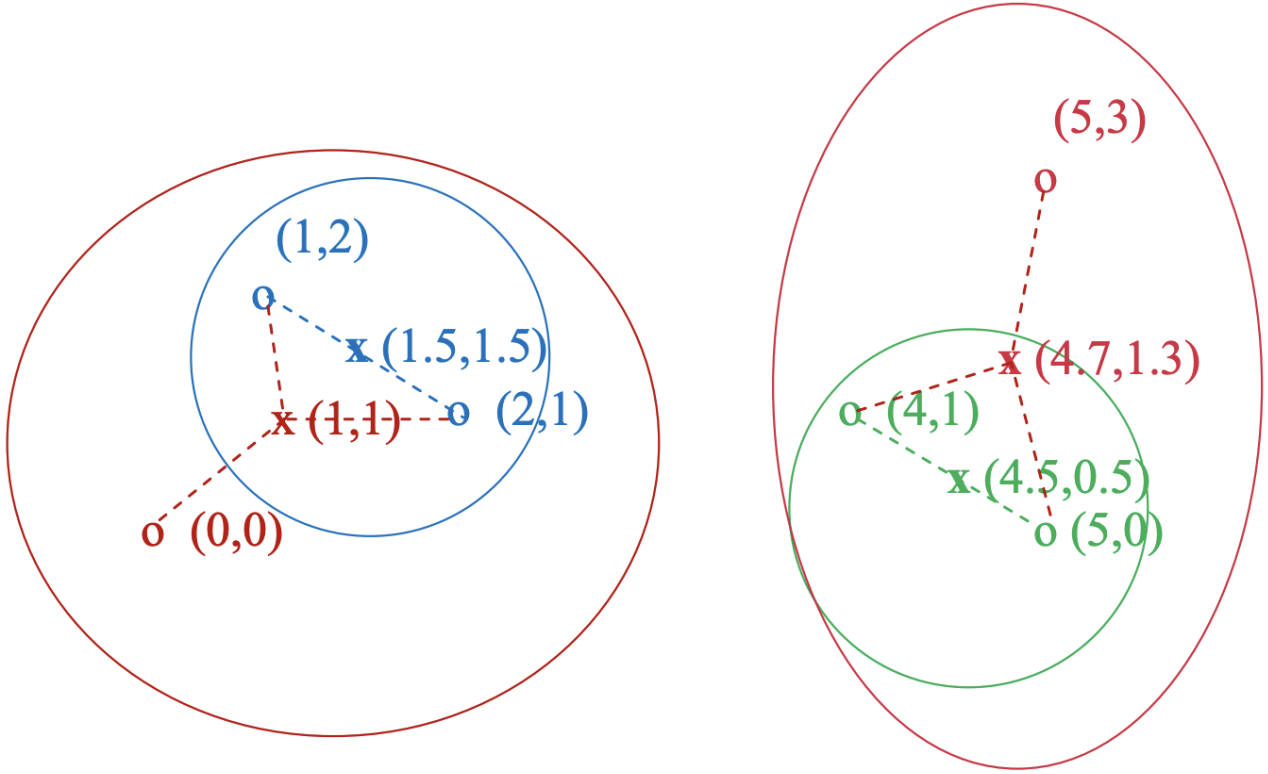
层次聚类

在欧几里得空间中：

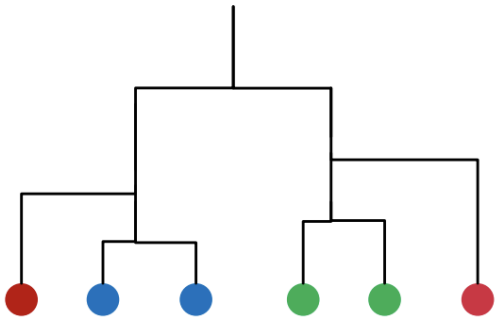
- **(1)如何表示一个由许多点组成的簇?**
  - 当我们合并集群时，我们用它的***质心***=它的(数据)点的平均值 来表示每个集群的“位置”

- **(2)如何确定聚类的接近度?**
  - 通过质心的距离来测量簇的距离
  - 合并两个距离最短的集群

层次聚类



数据：
O……数据点
X……质心



系统树图

层次聚类

在非欧几里得空间中：

- 我们唯一能谈论的“位置”就是点本身。主要有三种方法：

(1) 如何表示一个由多个点组成的簇?

1. 选择一个**簇中心点** = 离其他点“最近” 的点

2,3. 作为点的集合

(2)如何确定聚类的接近度?

1. 把簇心视为质心
2. 两簇点之间的各种距离测度
3. 两簇并集的各种内聚性度量

层次聚类

方法1：

(1) 如何表示一个簇：

- 选择一个**簇中心点** = 离其他点“最近” 的(数据)点

  “最近”的可能含义:

  - 到其他点的最大距离最小
  - 到其他点的平均距离最小
  - 到其他点的距离的平方和最小
    - 对于距离度量d，簇C的簇心c是$arg\min_{c} \sum_{x \in C} d(x,c)^2$

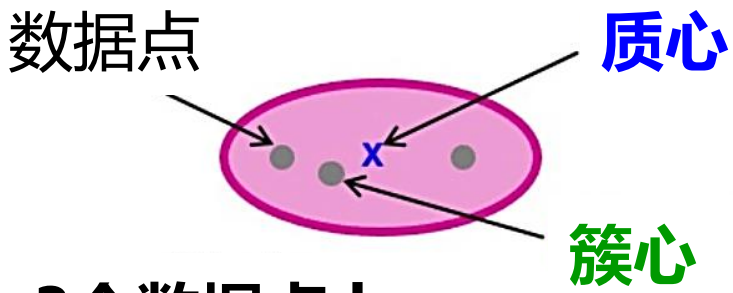
(2)如何确定聚类的接近度?

- 把簇心视为质心

质心 VS 簇心

- **质心**是簇中所有(数据)点的平均值。
  - 这意味着质心是一个“人造”点。

- **簇心**是一个与簇中所有其他点“最近”的(数据)点。



3个数据点上的聚类

层次聚类

方法2：

(1) 如何表示一个簇？ 作为点的集合

(2)如何确定聚类的接近度？

定义 *簇间* 距离：

- 任意两点之间的最小距离，每个簇一个点
- 所有点对的平均距离，每个簇一个点

层次聚类：非欧几里得空间

方法3：

(1) 如何表示一个簇？ 作为点的集合

(2)如何确定聚类的接近度？

定义内聚的概念，合并内聚性最强的簇

可能的内聚概念(越小，内聚性越强)：

- 合并的簇的**直径** = 簇中点之间的最大距离
- 簇中点之间的**平均距离**
- 合并的簇的**密度** = 簇中点的数量/直径或平均距离

何时停止?

(3) 何时停止聚类合并?

- 当找到k个簇时(假设我们知道簇的个数)
- 当停止准则满足时
  - 如果直径超过阈值，则停止
  - 如果密度低于某个阈值，则停止
  - 如果合并集群产生了一个坏的集群，则停止
    - 例如，直径长度突然跳跃
- 不断合并，直到只剩下1个簇

哪种设计选择是最好的?

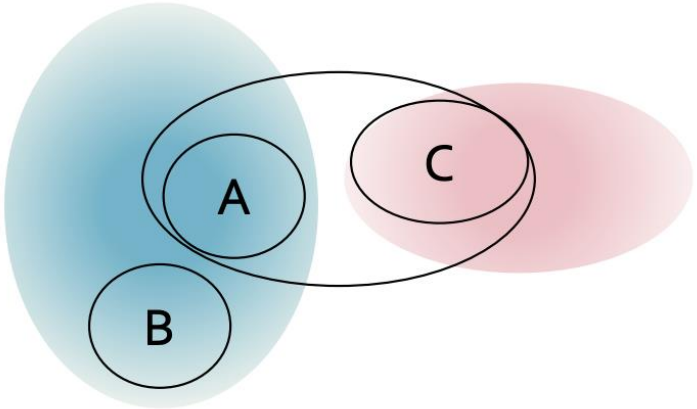
- 这取决于簇的形状。
  - 你可能事先不知道。

- **示例:** 我们将比较两种方法:

1. 合并质心之间距离最小的簇(或非欧几里得空间的簇)
2. 每个簇选一个点，合并两个点之间距离最小的簇

案例1：凸簇

- 基于质心的合并效果很好。
- 但是基于最近点的合并可能会不小心合并错误。

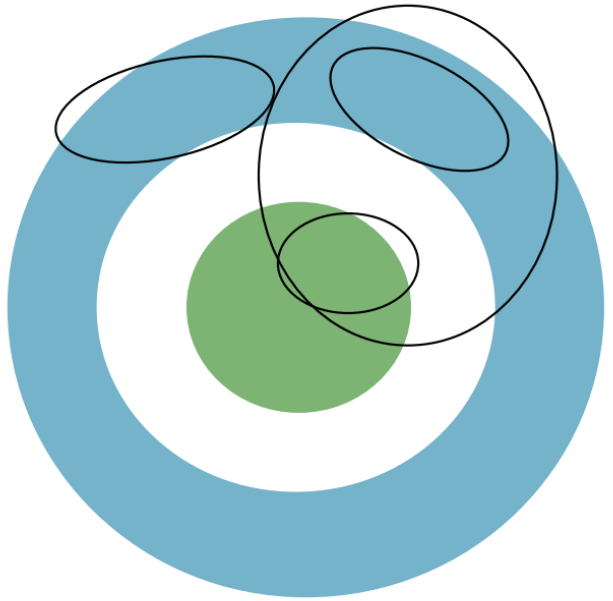


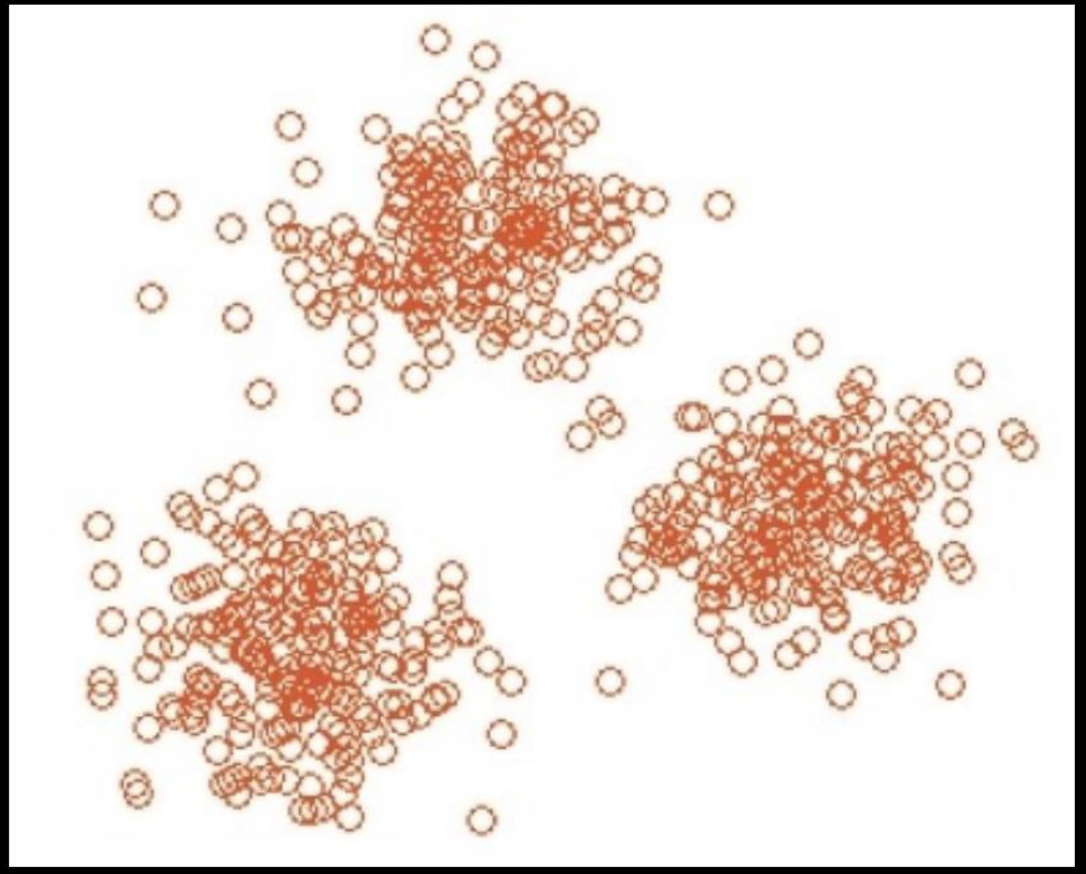
A和B的质心比A和C更近，但A和C最近的点之间距离最近。

数据密度

案例2：同心集群

- 基于最近点的合并效果很好
- 但是基于质心的合并可能会导致错误





K-Means聚类

K-Means算法

- K-Means（k-均值算法）通常用来解决如下问题：

- **问题：**给定欧式空间/距离和***k*=**簇/聚类数量，确定每个样本点应该属于***k***个簇中的哪一个，并且找到能够将每个点到其簇中心的距离平方和最小化的***k***个簇中心（质心）

- 找到精确的解是NP难的。
- 近似解法是Lloyd's算法或k-均值算法。

K-Means算法

- 通过选择k个中心的方式初始化簇

直到收敛：

- 1）对于每个点，将其分配给当前质心最近的簇
- 2）分配完所有点后，更新k个簇的质心，更新方式为到每个簇中数据点的距离平方和最小

收敛意味着点不会在簇之间移动，质心稳定

K-Means算法

1. 初始化

随机选择 k 个点作为初始中心：

$$\mu_1^{(0)}, \mu_2^{(0)}, \ldots, \mu_k^{(0)}$$

2. Assignment Step (分配步骤)

在第 t 次迭代时，将每个点 x_i 分配给最近的簇中心：

$$c_i^{(t)} = \arg\min_{j \in \{1,\ldots,k\}} \|x_i - \mu_j^{(t)}\|^2$$

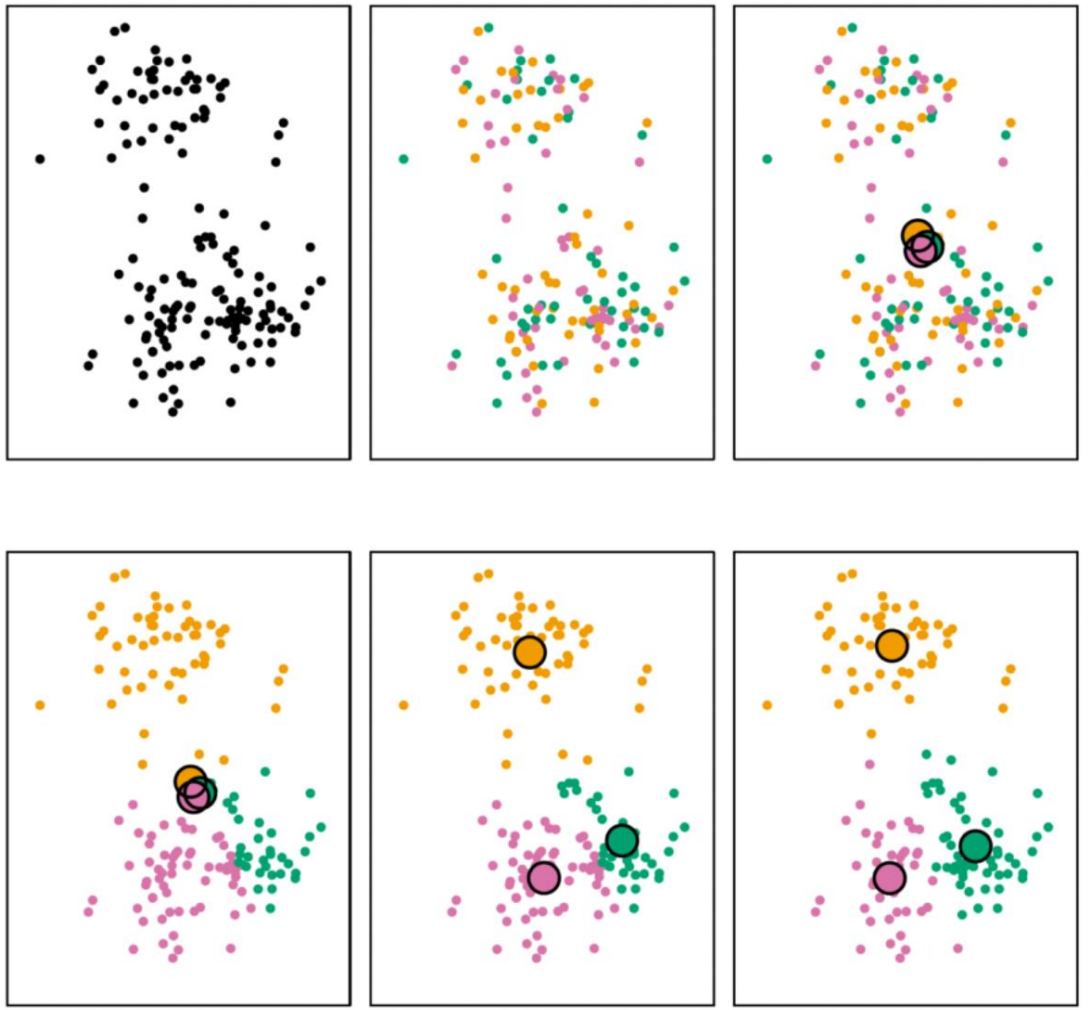
即：样本 x_i 属于簇 $C_{c_i^{(t)}}$。

3. Update Step (更新步骤)

对每个簇 j，重新计算中心为该簇所有点的均值：

$$\mu_j^{(t+1)} = \frac{1}{|C_j^{(t)}|} \sum_{x_i \in C_j^{(t)}} x_i$$

其中 $C_j^{(t)} = \{x_i \mid c_i^{(t)} = j\}$。



K-Means算法的缺点

k-means算法的收敛严重依赖于质心的初始选择。它有时可能表现得非常糟糕。

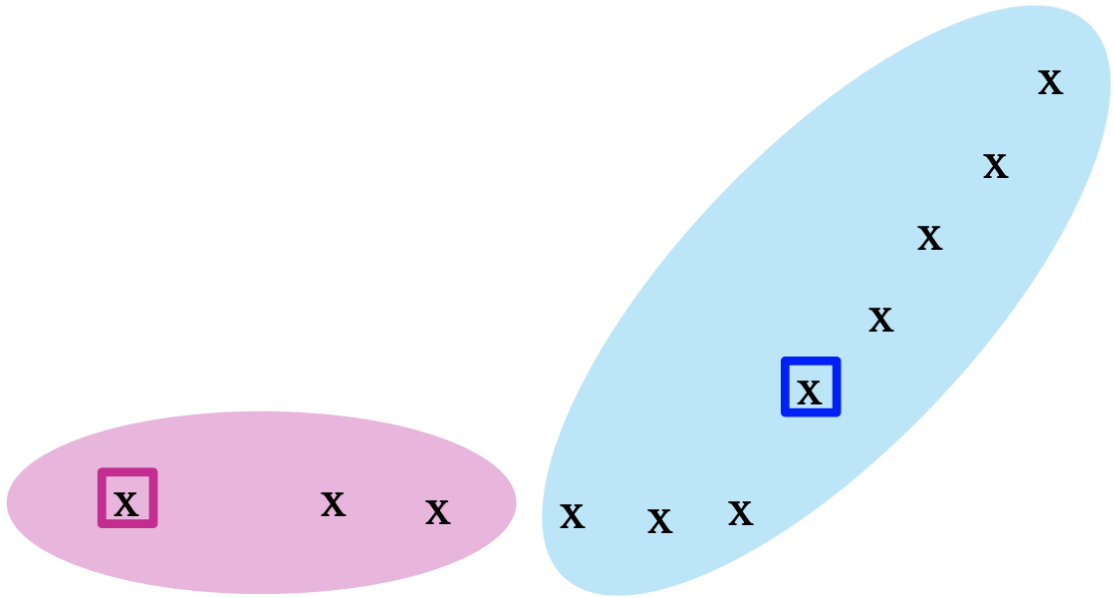
- 选取k个中心点的不同策略：
  - 随机选择k个数据点
  - K-means ++

K-means ++

- **基本思想：**选取一个小样本点集S，通过任意算法对其进行聚类，并使用质心作为种子
- 在 K-means ++ 中，样本大小 $|S|=k$ 乘以一个因子，该因子是点的总数的对数

- **如何选择样本点：**随机访问样本点，但在样本中添加点 p 的概率与 $D(p)^2$ 成正比。
  - $D(p) = p$和最近的已选点之间的距离。

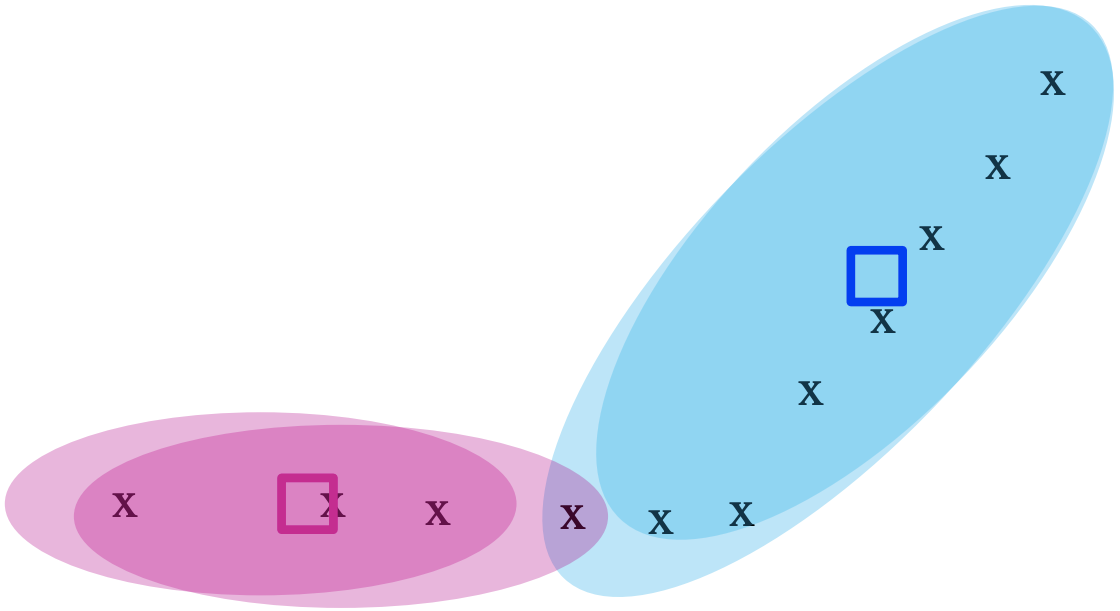
示例：点分配



x……数据点
□……质心

第一轮过后的集群

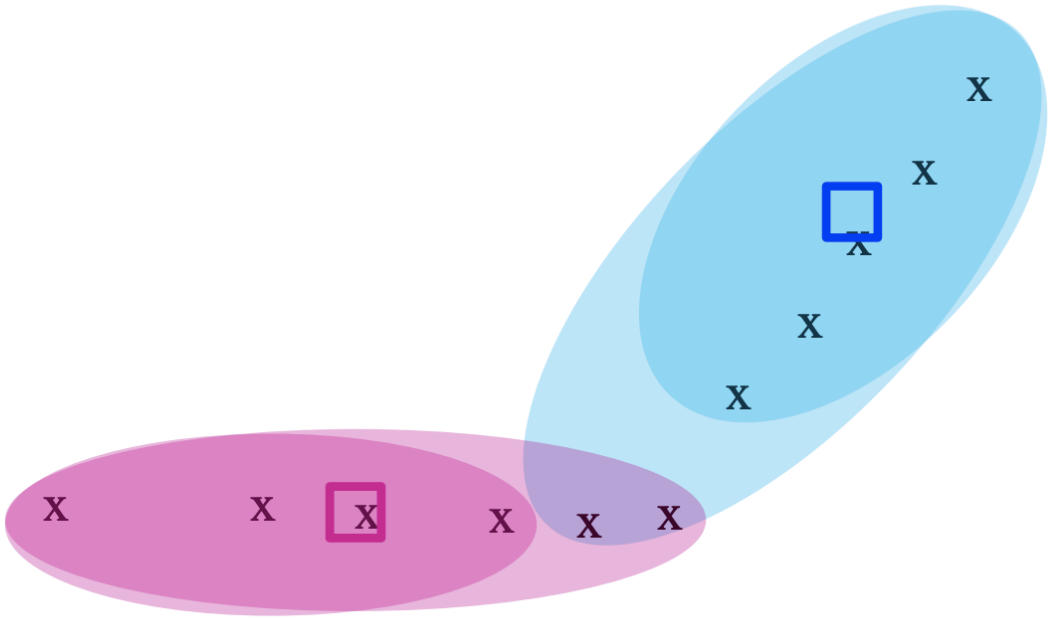
示例：点分配



x……数据点
□……质心

第二轮过后的集群

示例：点分配



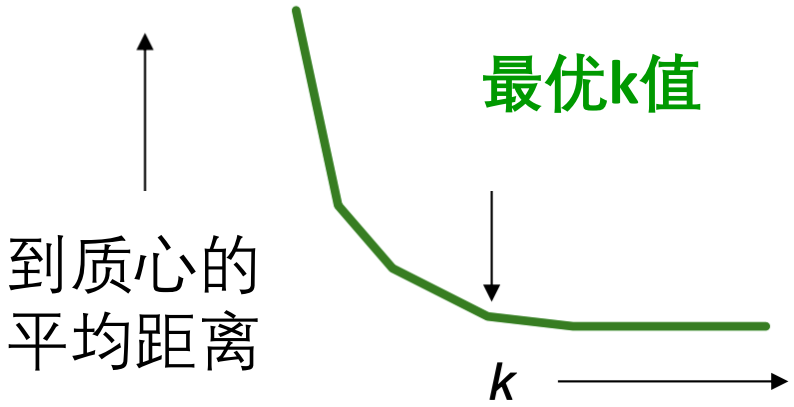
x……数据点
□……质心

最后的集群

选择合适的*k*

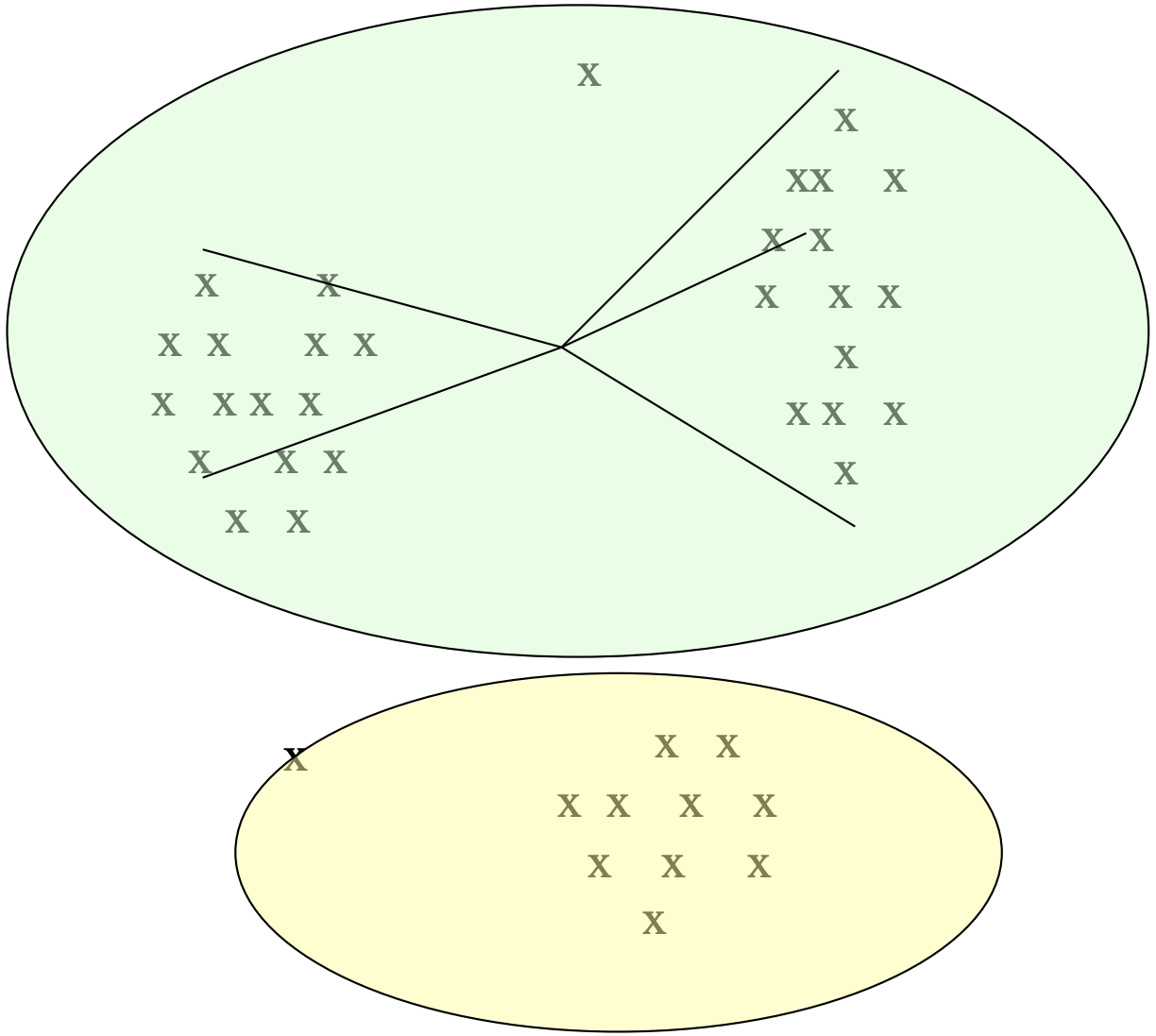
如何选择k?

- 尝试不同的 k，观察随着 k 的增加到质心的平均距离的变化
- 平均值迅速下降，直到正确的k，然后变化就很小



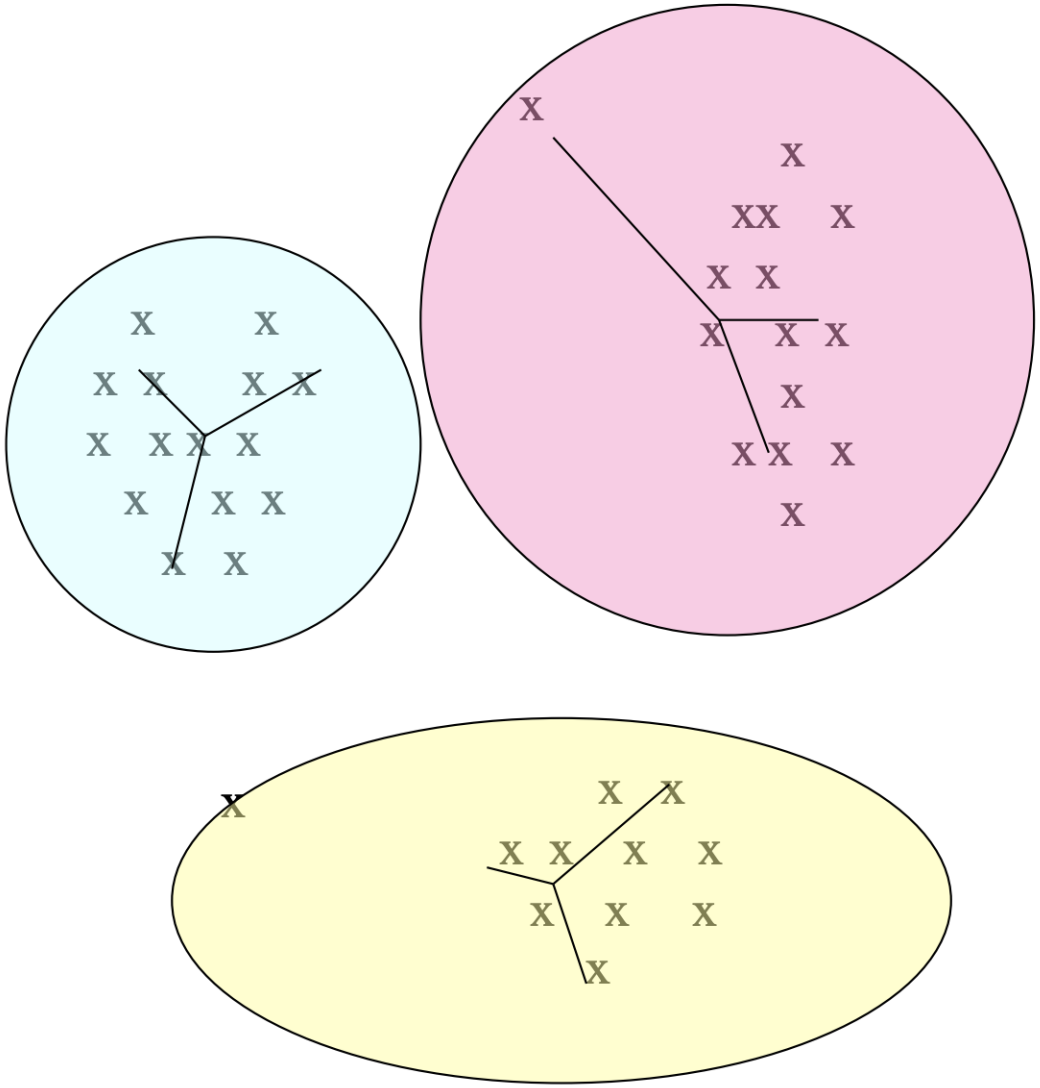
示例：选择k

太少；
许多点到质心的距离过长



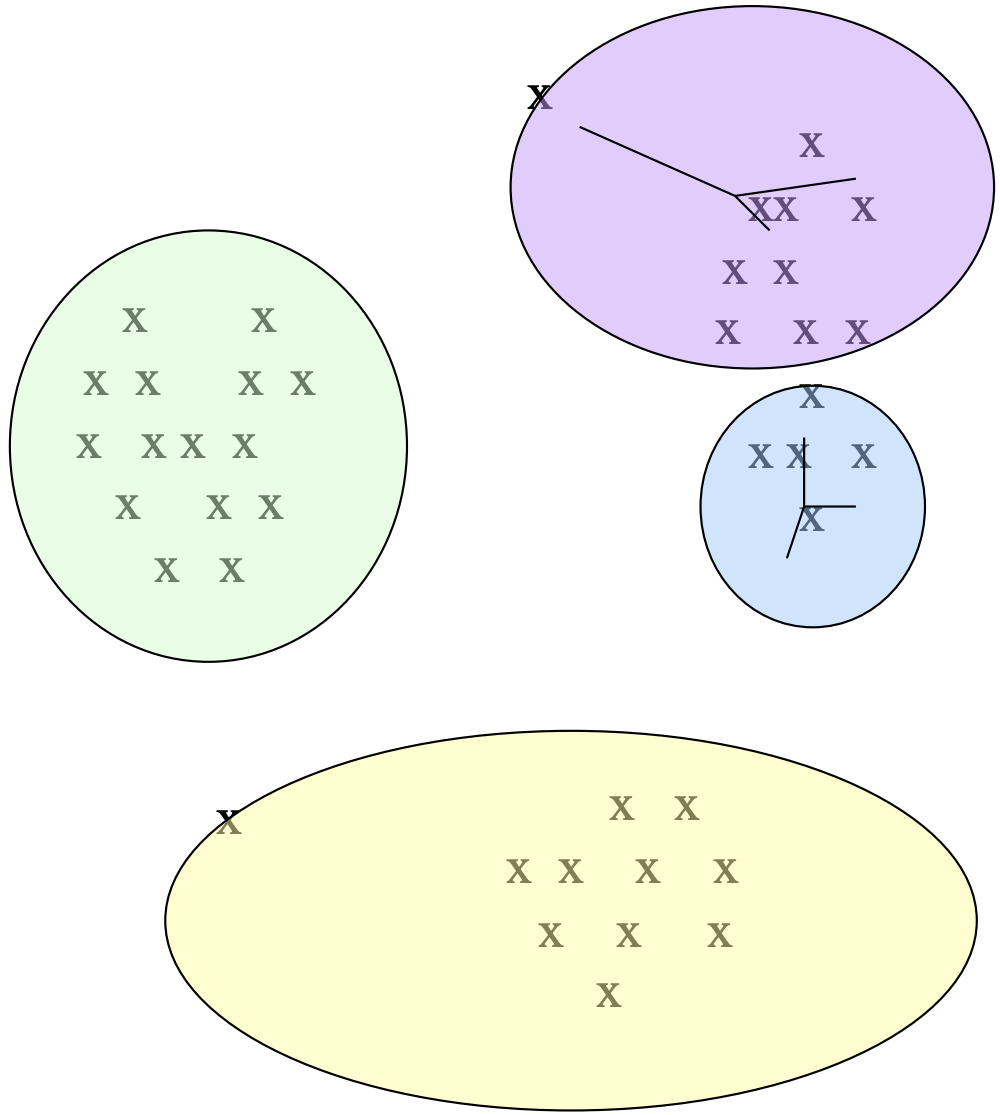
示例：选择k

刚刚好；
点到质心的
距离较短



示例：选择k

太多;
点到质心的平均距离几乎没有改善

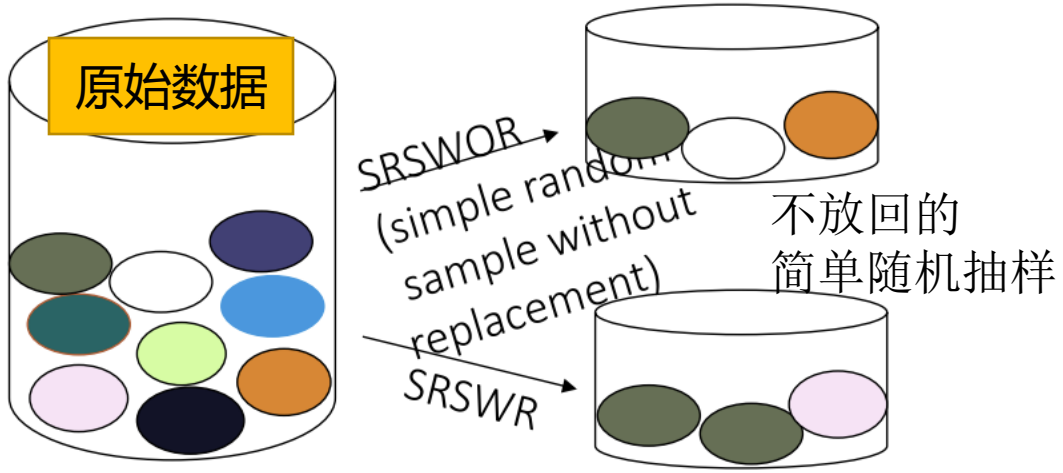


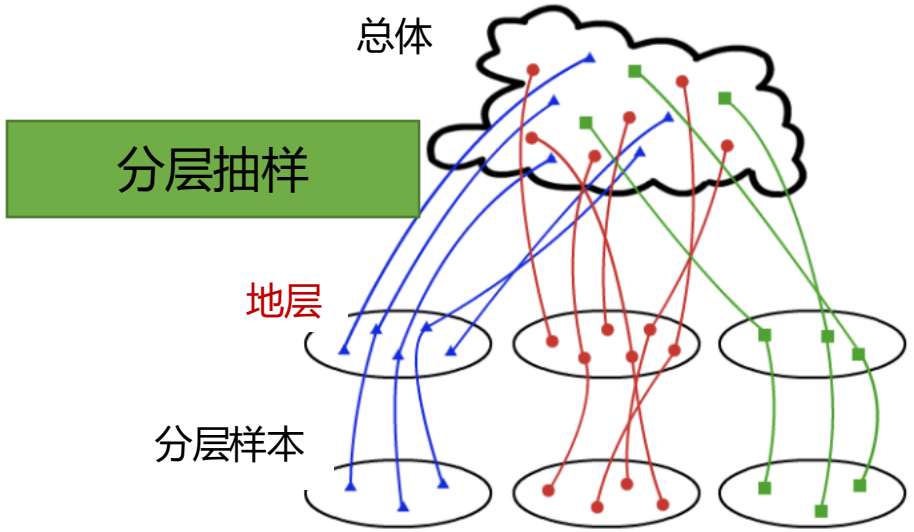
抽样

- **抽样：**获取一个小样本 s 来代表整个数据集 N
- 允许挖掘算法以次线性于数据大小的复杂度运行
- 关键原则：选择具有**代表性的**数据子集
  - 在存在偏斜的情况下，简单随机抽样可能具有非常差的性能
  - 开发自适应抽样方法，例如，分层抽样
- 注意：抽样可能不会减少数据库I/O（一次一页）

抽样类型

- **简单随机抽样：**选择任何项的概率均相等
- **不放回抽样**
  - 一旦一个对象被选中，它就从总体中删除
- **有放回抽样**
  - 选取的对象不会从总体中删除
- **分层抽样**
  - 对数据集进行划分(或聚类)，并从每个划分中抽取样本(按比例，即近似相同的数据百分比)。





数据变换

- 将给定属性的整个值集合映射到一组新的替换值的函数，以便每个旧值都可以用一个新值来标识
- 方法
  - 平滑：从数据中去除噪声
  - 属性/特征构造
    - 根据给定属性构建的新属性
  - 聚合：汇总, 数据立方构建
  - 归一化：缩放到更小的指定范围内
    - 最小-最大 归一化
    - z分数归一化
    - 十进制归一化
  - 离散化：概念层次攀升

归一化

- **最小-最大归一化：**到[new_min$_A$, new_max$_A$]

$$v' = \frac{v - min_A}{max_A - min_A}(new\_max_A - new\_min_A) + new\_min_A$$

  - 例：将收入范围12,000美元至98,000美元归一化为[0.0,1.0]
    - 那么73000美元就映射为$\frac{73,600-12,000}{98,000-12,000}(1.0-0)+0=0.716$
- **Z-值归一化** (μ: 平均值，σ: 标准差):

$$v' = \frac{v-\mu_A}{\sigma_A}$$

Z-值：原始得分和总体均值之间的距离，以标准差为单位

  - 例：设μ = 54000， σ= 16000，则$\frac{73,600-54,000}{16,000}=1.225$
  - 十进制归一化

$$v' = \frac{v}{10^j}$$

其中，j 是使$max(|v'|) < 1$的最小整数

离散化

- 属性的三种类型
  - 名义型——无序集合中的值，如颜色, 职业
  - 有序型——有序集合中的值，如军衔或学术级别
  - 数值型——实数，如整数或实数
- 离散化：将连续属性的范围划分为区间
  - 然后可以使用区间标签来替换实际数据值
  - 通过离散化降低数据规模
  - 拆分(自上而下) vs. 合并(自下而上)
  - 为进一步分析做准备，例如分类

简单离散化：分箱

- **等宽**（距离）分区
  - 将范围划分为大小相等的N个区间：均匀网格
  - 如果 A 和 B 是属性的最小值和最大值，则间隔的宽度为：$\boldsymbol{W = (B - A)/N}$
  - 是最直接的方法，但离群点可能会主导展示
  - 偏斜数据处理不好
- **等深度**（频率）分区
  - 将范围划分为N个区间，每个区间包含的样本数量大致相同
  - 良好的数据扩展性
  - 管理分类属性可能很棘手

示例：用于数据平滑的分箱方法

- 价格排序数据（美元）：4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* 划分为等频率(**等深度**)箱： **3 个箱**

示例：用于数据平滑的分箱方法

- 价格排序数据（美元）：4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* 划分为等频率(**等深度**)箱：
  - 箱 1：4, 8, 9, 15
  - 箱 2：21, 21, 24, 25
  - 箱 3：26, 28, 29, 34

* 按**箱均值**平滑：

示例：用于数据平滑的分箱方法

- 价格排序数据（美元）：4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* 划分为等频率(**等深度**)箱：
  - 箱 1：4, 8, 9, 15
  - 箱 2：21, 21, 24, 25
  - 箱 3：26, 28, 29, 34

* 按**箱均值**平滑：
  - 箱 1：9, 9, 9, 9
  - 箱 2：23, 23, 23, 23
  - 箱 3：29, 29, 29, 29

* 按**箱边界**平滑：

示例：用于数据平滑的分箱方法

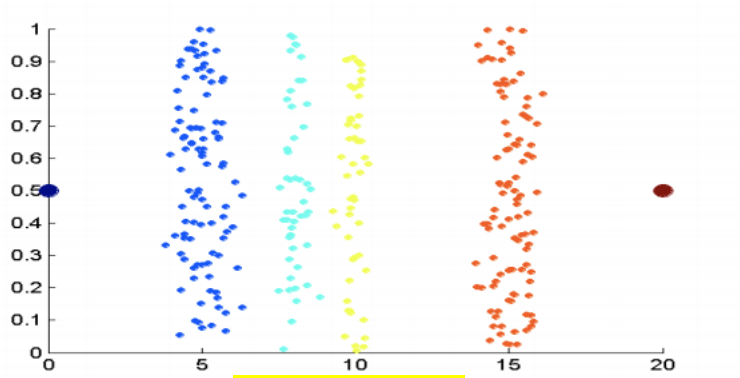
- 价格排序数据（美元）：4, 8, 9, 15, 21, 21, 24, 25, 26, 28, 29, 34

* 划分为等频率(**等深度**)箱：
  - 箱 1：4, 8, 9, 15
  - 箱 2：21, 21, 24, 25
  - 箱 3：26, 28, 29, 34

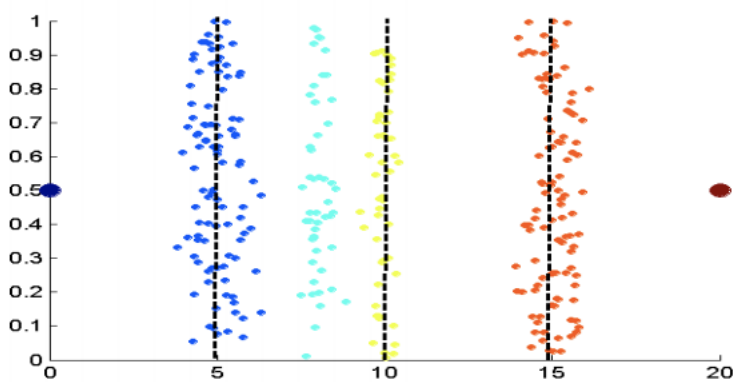
* 按**箱均值**平滑：
  - 箱 1：9, 9, 9, 9
  - 箱 2：23, 23, 23, 23
  - 箱 3：29, 29, 29, 29

* 按**箱边界**平滑：
  - 箱 1：4, 4, 4, 15
  - 箱 2：21, 21, 25, 25
  - 箱 3：26, 26, 26, 34

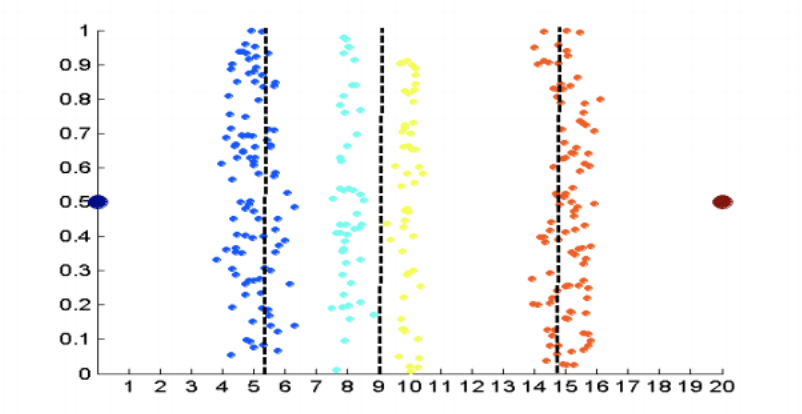
无监督离散化：分箱 vs. 聚类



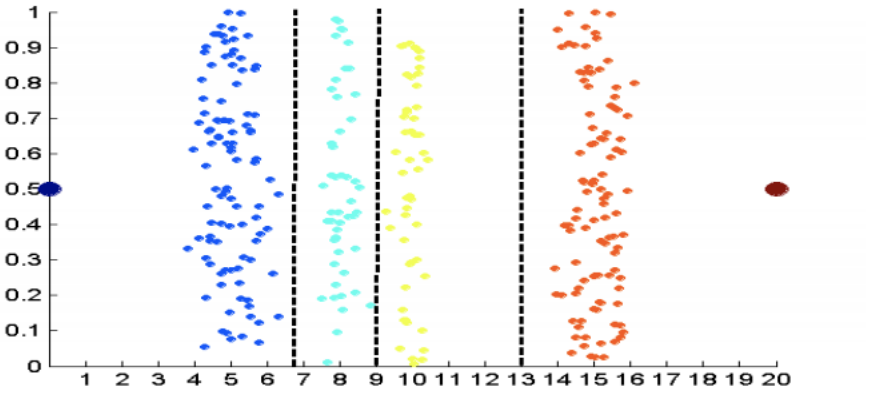
数据



等宽(距离)分箱



等深度(频率)分箱



K-均值聚类可以得到更好的结果

数据预处理

- 数据预处理：概述
- 数据清洗
- 数据集成
- 数据规约与变换
- 降维
- 总结

维度规约（降维）

- 维度诅咒
  - 当维度增加时，数据变得越来越稀疏
  - 对于聚类和离群点分析至关重要的点之间的密度和距离，变得不那么有意义
  - 子空间的可能组合数将指数级增长
- 维度规约（降维）
  - 通过获得一组主变量的方式来减少需要考虑的随机变量的数量
- 降维的优势
  - 避免维度诅咒
  - 有助于消除不相关的特征和减少噪声
  - 减少数据挖掘所需的时间和空间
  - 更容易可视化

降维技巧

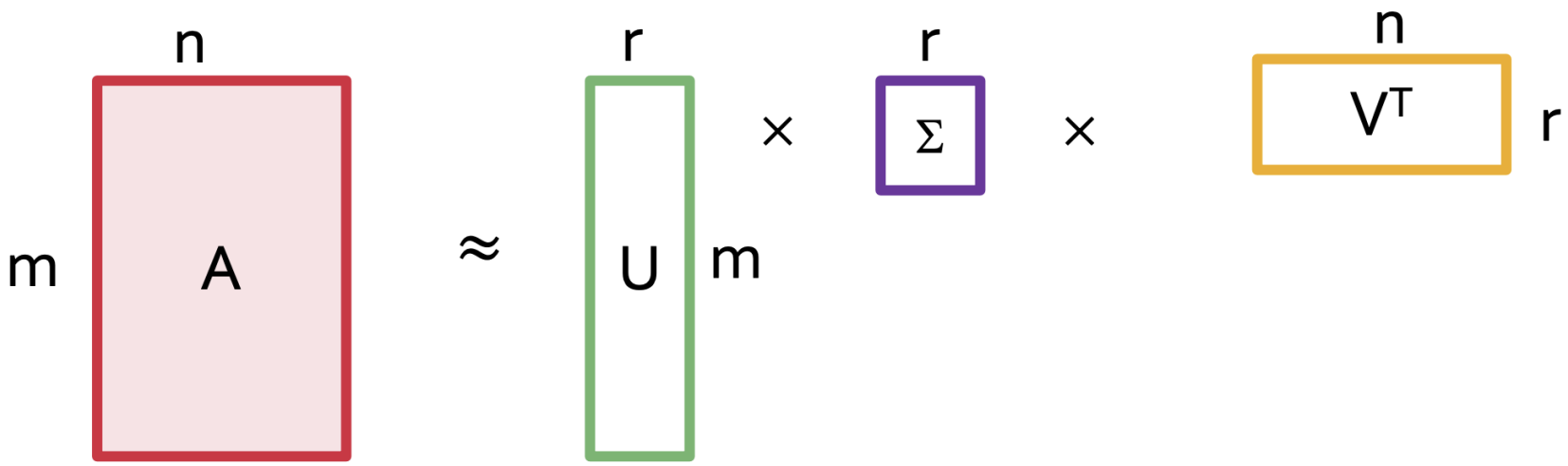
- 降维方法
  - **特征选择：**找到原始变量（或特征、属性）的子集
  - **特征提取：**将高维空间中的数据转换到低维空间中
- 一些典型的降维方法
  - 主成分分析
  - 有监督的非线性技术
    - 特征子集选择
    - 特征创建
  - 矩阵分解

降维技巧

- 降维方法
  - **特征选择：**找到原始变量（或特征、属性）的子集
  - **特征提取：**将高维空间中的数据转换到低维空间中
- 一些典型的降维方法
  - 主成分分析
  - 有监督的非线性技术
    - 特征子集选择
    - 特征创建
  - **矩阵分解**

矩阵降维

- 通常，我们的数据可以用 m × n 矩阵表示
- 这个矩阵可以近似为三个共享一个公共维度 r（很小） 的矩阵的乘积



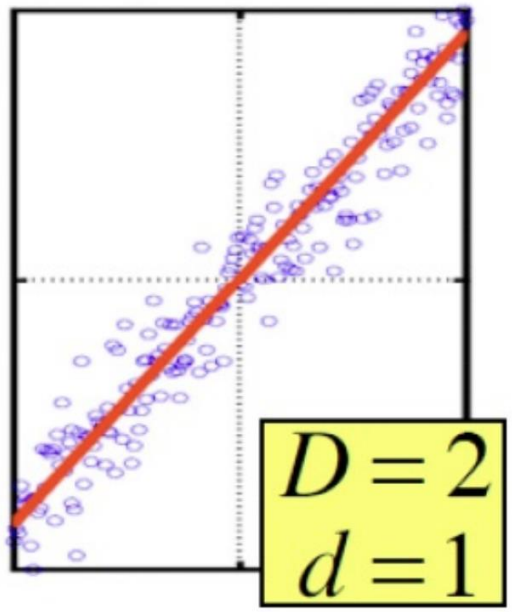
矩阵降维

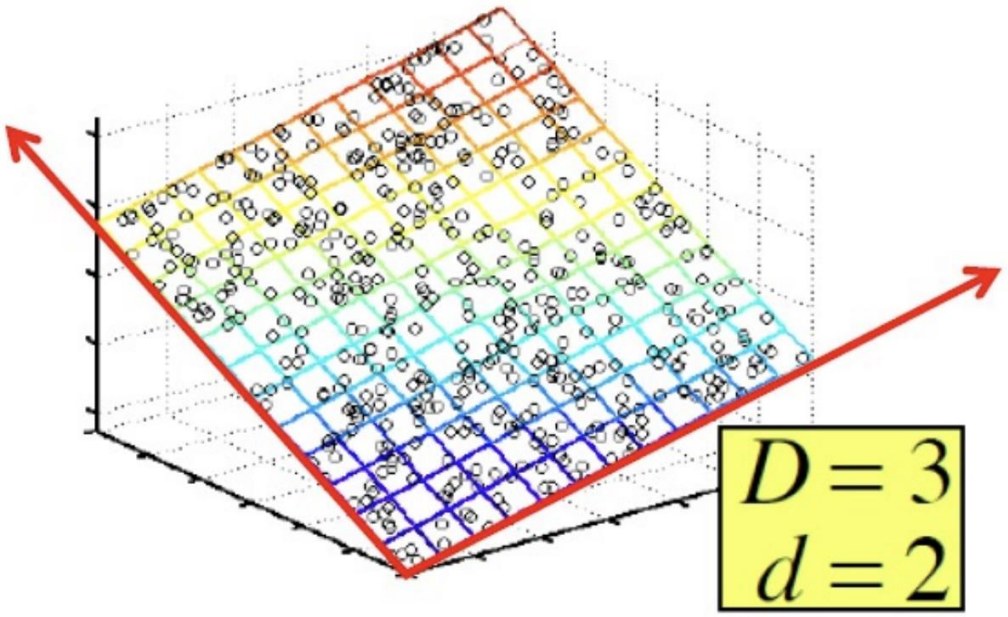
- **压缩 / 降维：**
  - 10^6 行；10^3 列；

day / customer	Wc 7/10/96	Th 7/11/96	Fr 7/12/96	Sa 7/13/96	Su 7/14/96	新表示
ABC Inc.	1	1	1	0	0	[1 0]
DEF Ltd.	2	2	2	0	0	[2 0]
GHI Inc.	1	1	1	0	0	[1 0]
KLM Co.	5	5	5	0	0	[5 0]
Smith	0	0	0	2	2	[0 2]
Johnson	0	0	0	3	3	[0 3]
Thompson	0	0	0	1	1	[0 1]

注意：上面的矩阵实际上是“二维的”。所有行都可以通过缩放为 [1 1 1 0 0] 或 [0 0 0 1 1] 的方式来重构

矩阵降维



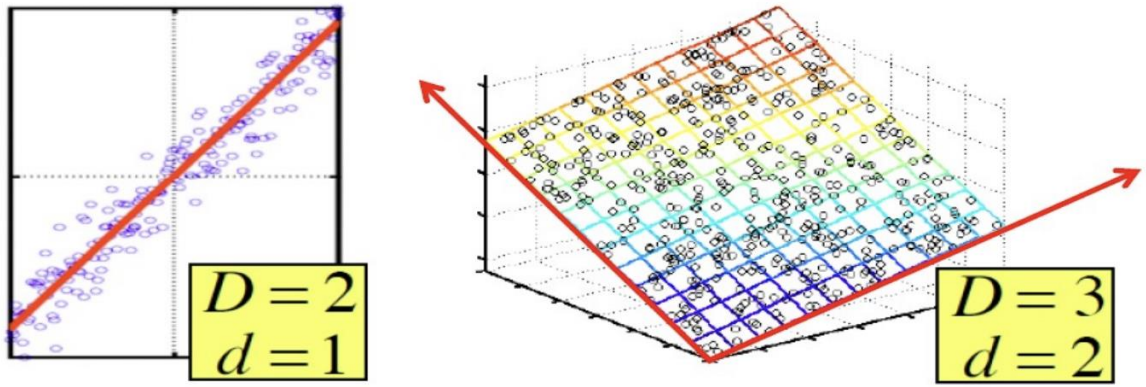


有一些隐藏的或**潜在的因素、潜在的维度**，在很大程度上可以解释为什么这些值在数据矩阵中出现

矩阵降维

这些维度的轴可以通过以下方式选择：

- 第一个维度是点表现出最大方差的方向
- 第二个维度是与第一个维度正交的方向，该维度中的点显示出第二大方差
- 等等...，直到有足够多的维度使得方差降得够低

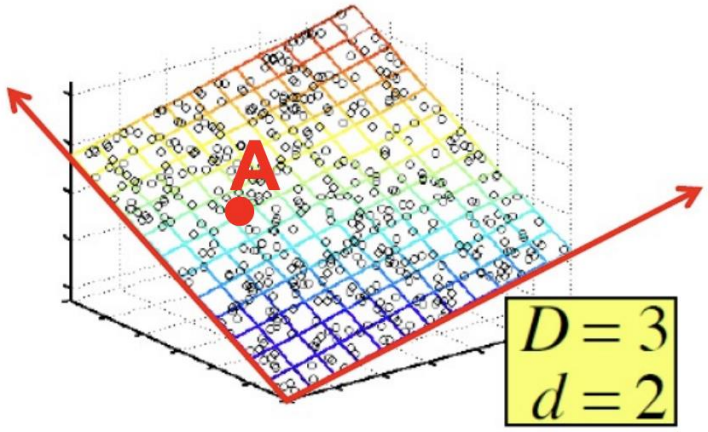


秩是维度

- **问题：**矩阵 **A** 的**秩**是多少?
- **回答：**矩阵A **线性无关**的行数
- **三维空间点云：**
  - 将点坐标视为一个矩阵：

一点一行

$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \begin{matrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{matrix}$$



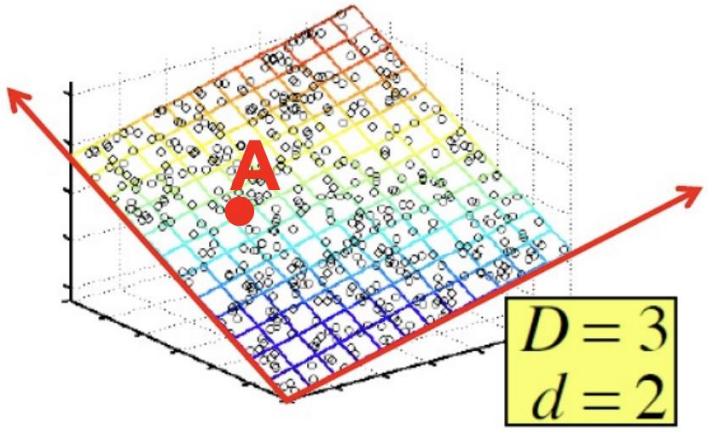
- **我们可以更高效地重写坐标！**
  - 旧的基向量：[1 0 0] [0 1 0] [0 0 1]
  - **新的基向量：**

秩是维度

- **问题：** 矩阵 **A** 的**秩**是多少?
- **回答：** 矩阵A **线性无关**的行数
- **三维空间点云：**
  - 将点坐标视为一个矩阵：

一点一行

$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \begin{matrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{matrix}$$



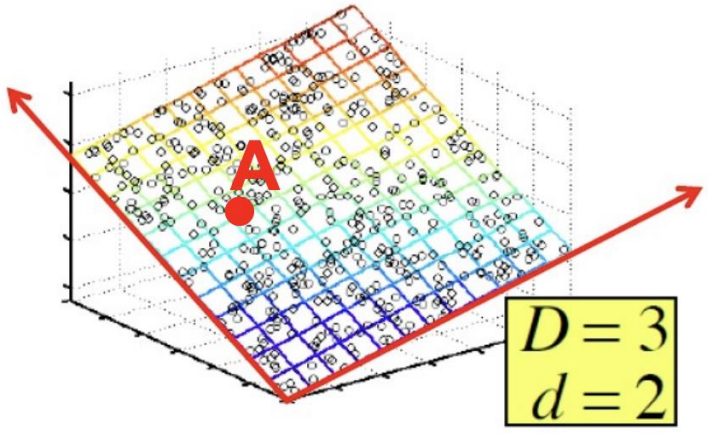
- **我们可以更高效地重写坐标！**
  - 旧的基向量：[1 0 0] [0 1 0] [0 0 1]
  - **新的基向量：[1 2 1] [-2 -3 1]**
  - 那么 **A** 就有了新的坐标：[1 0]，**B:** [0 1]，**C:** [1 -1]

秩是维度

- **问题：** 矩阵 **A** 的**秩**是多少?
- **回答：** 矩阵A **线性无关**的行数
- **三维空间点云：**
  - 将点坐标视为一个矩阵：

一点一行

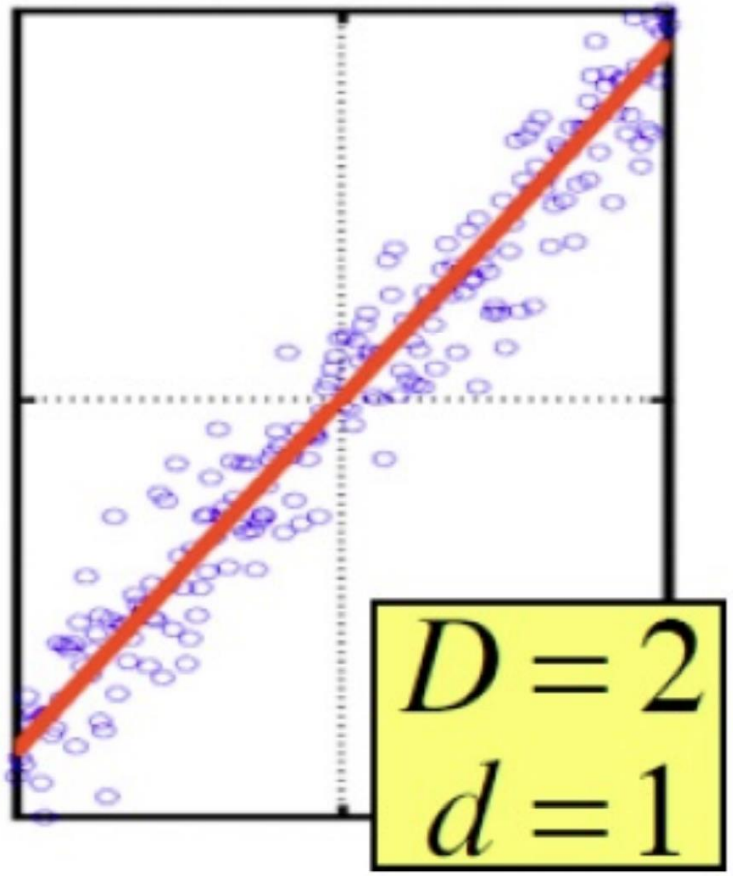
$$\begin{bmatrix} 1 & 2 & 1 \\ -2 & -3 & 1 \\ 3 & 5 & 0 \end{bmatrix} \begin{matrix} \mathbf{A} \\ \mathbf{B} \\ \mathbf{C} \end{matrix}$$



- **我们可以更高效地重写坐标！**
  - 旧的基向量：[1 0 0] [0 1 0] [0 0 1]
  - **新的基向量：[1 2 1] [-2 -3 1]**
  - 那么 **A** 就有了新的坐标：[1 0]，**B:** [0 1]，**C:** [1 -1]
    - **注意：我们减少了维度/坐标的数量！**

秩是维度

- **降维的目标是发现数据的轴!**



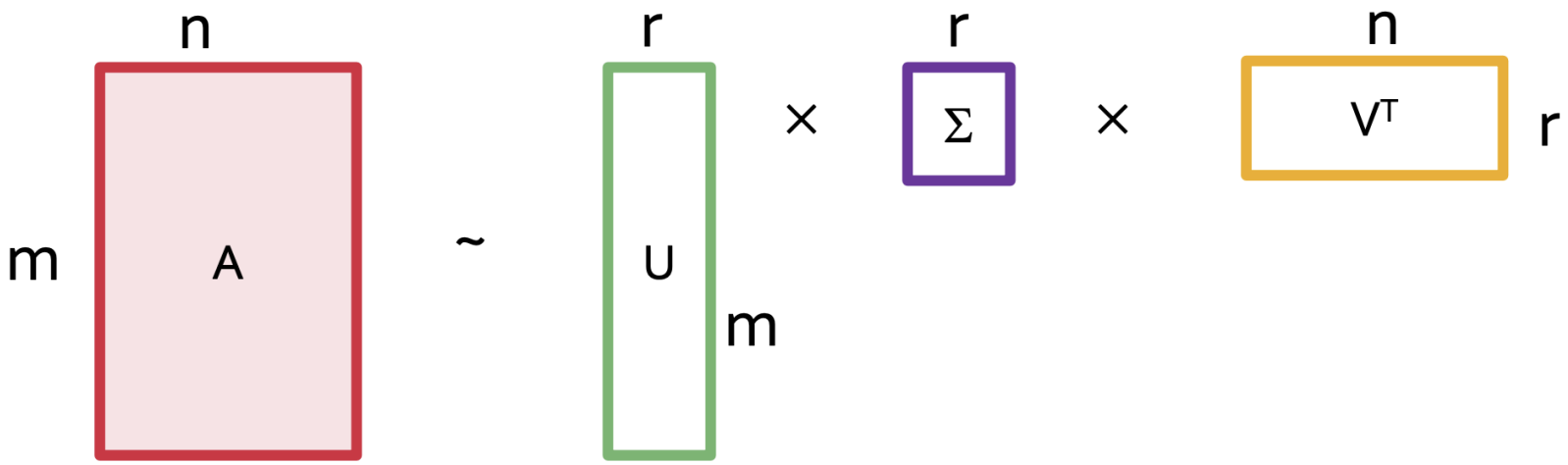
我们不是用2个坐标来刻画每个点，而是用1个坐标来刻画每个点(对应于红线上点的位置)。

这样做会产生一些误差，因为点并不完全位于直线上

SVD: 奇异值分解

降低矩阵维数

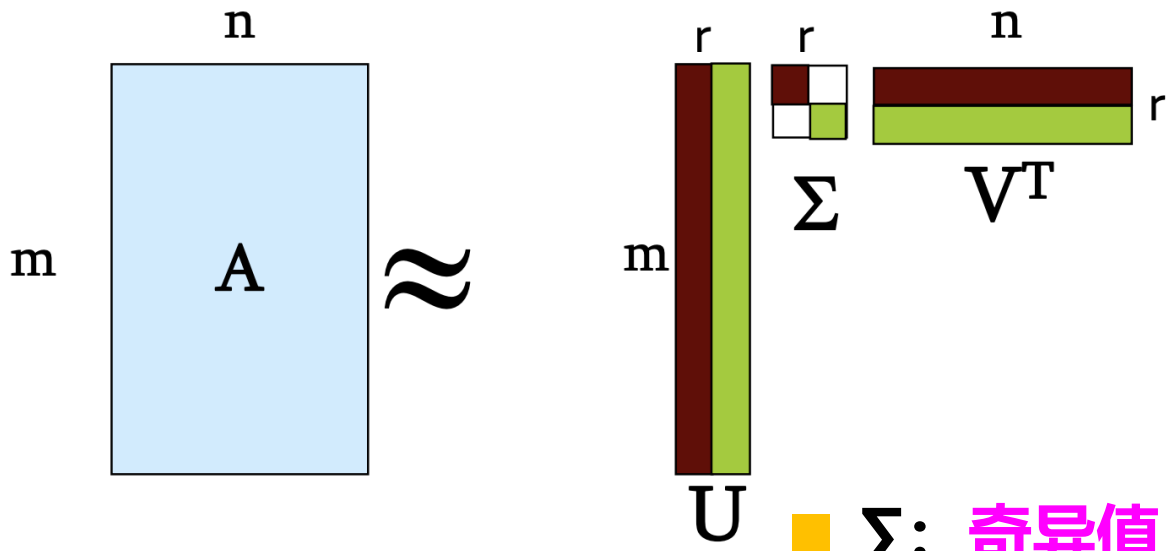
- **将任意矩阵分解为三个矩阵的乘积：**



- **这些矩阵的形式有很强的约束**
  - **分解结果是唯一的**
- 从这种分解中，你可以选择任意数量r的中间概念(潜在因子)，以最小化重构误差

SVD-定义

$$A \approx U\Sigma V^T = \sum_i \sigma_i u_i \circ v_i^T$$

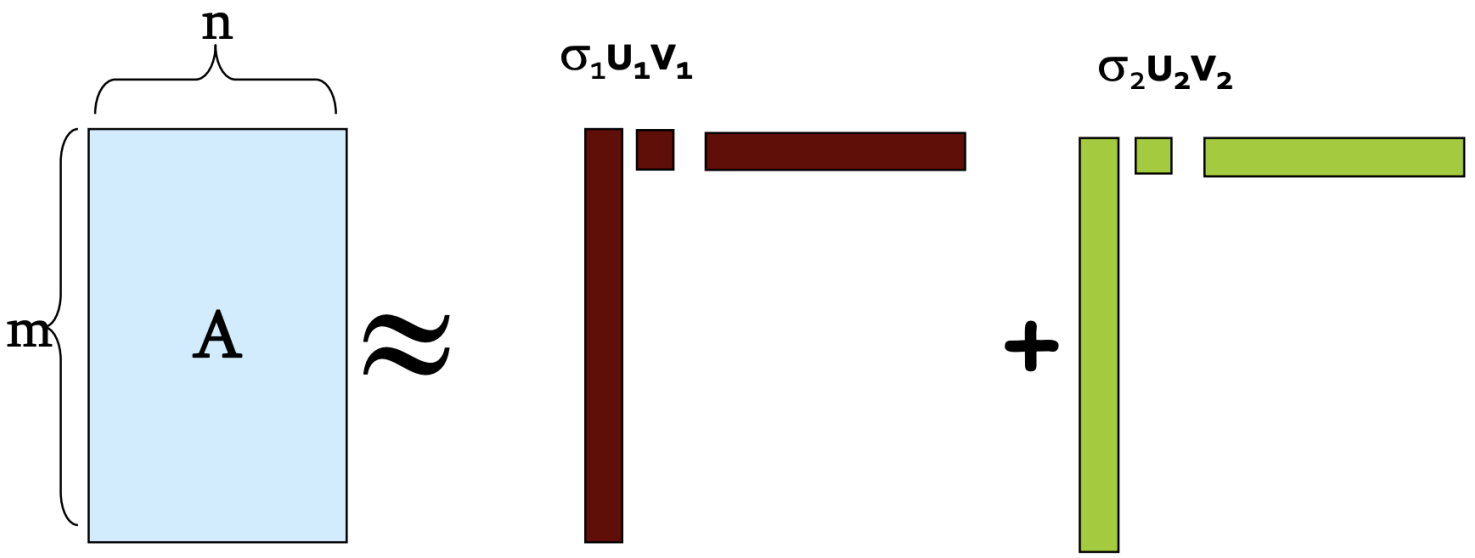


- **A：** **输入数据矩阵**
  - *m* x *n* 矩阵（如，*m* 个文档，*n* 个词）
- **U：** **左奇异向量**
  - *m* x *r* 矩阵（*m* 个文档，*r* 个概念）
- **Σ：** **奇异值**
  - *r* x *r* 对角矩阵（每个“概念”的强度）

  （*r*：矩阵A的秩）
- **V：** **右奇异向量**
  - *n* × *r* 矩阵（*n* 个词，*r* 个概念）

SVD-定义

$$\mathbf{A} \approx \mathbf{U}\mathbf{\Sigma}\mathbf{V}^T = \sum_i \sigma_i \mathbf{u}_i \circ \mathbf{v}_i^T$$



如果我们设置 $\sigma_2=0$，那么绿色列也可以不存在。

σ_i … 标量
u_i … 向量
v_i … 向量

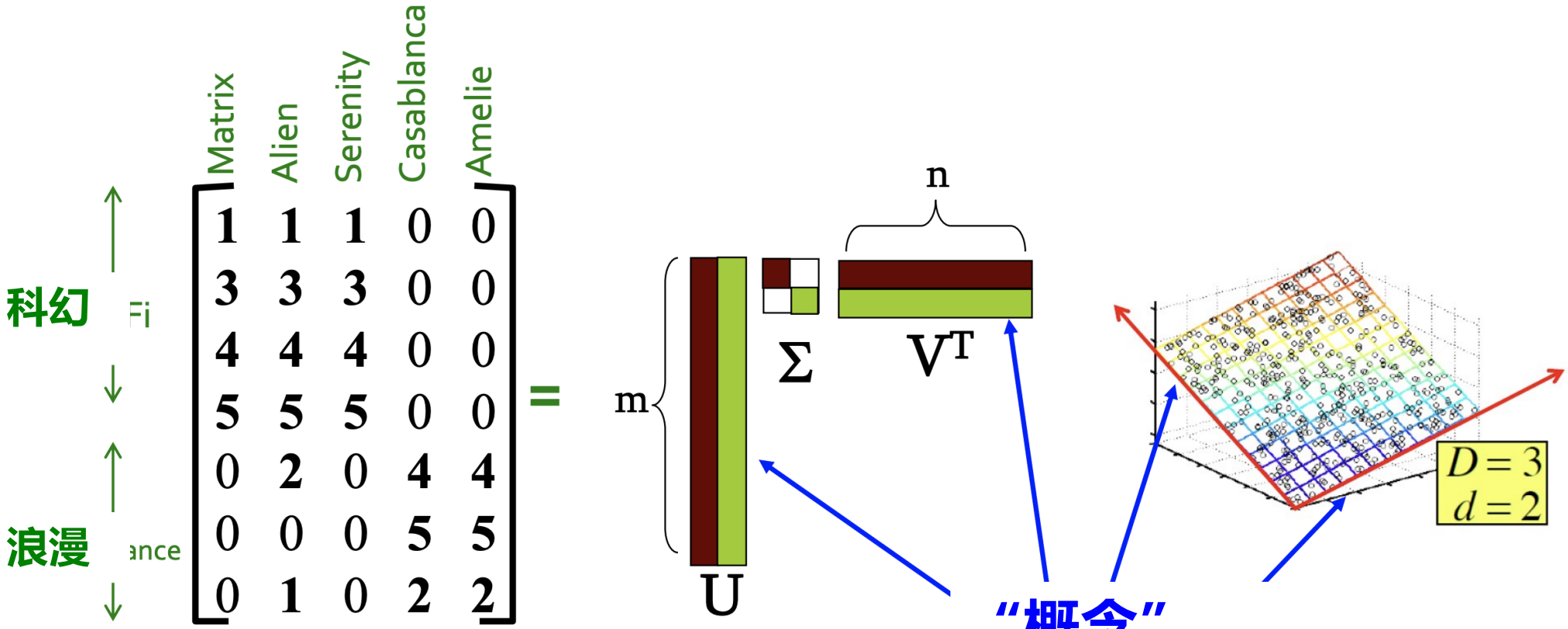
SVD-特性

总有可以将一个实矩阵 **A** 分解为 $A = U\Sigma V^T$，其中，

- ***U, Σ, V:*** 唯一
- ***U, V:*** 列正交
  - $U^T U = I; V^T V = I$ (I: 单位矩阵)
  - (列是正交的单位向量)
- ***Σ:*** 对角
  - 元素(**奇异值**)是非负的，并且按降序排序 $(\sigma_1 \geq \sigma_2 \geq \ldots \geq 0)$

示例：用户-电影

- 考虑一个矩阵。SVD的作用是什么？



评分矩阵，其中每一列对应一部电影，每一行对应一个用户。前4名用户更喜欢科幻题材，而其他用户更喜欢爱情题材。

N

示例：用户-电影

- $A = U \Sigma V^T$ - 示例：用户-电影

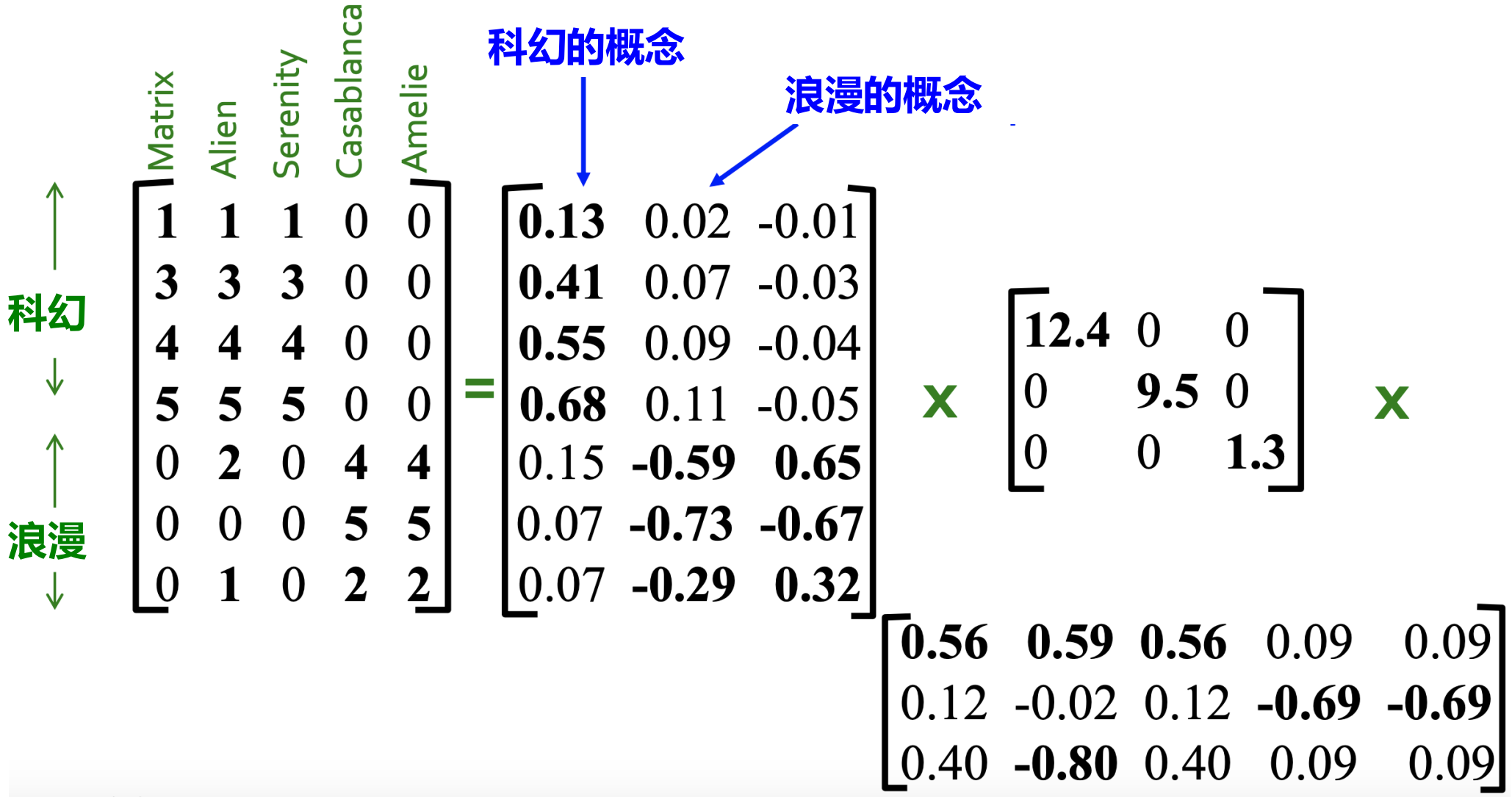
	Matrix	Alien	Serenity	Casablanca	Amelie
科幻	1	1	1	0	0
	3	3	3	0	0
	4	4	4	0	0
	5	5	5	0	0
浪漫	0	2	0	4	4
	0	0	0	5	5
	0	1	0	2	2

$$=\begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times$$

$$\begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

示例：用户-电影

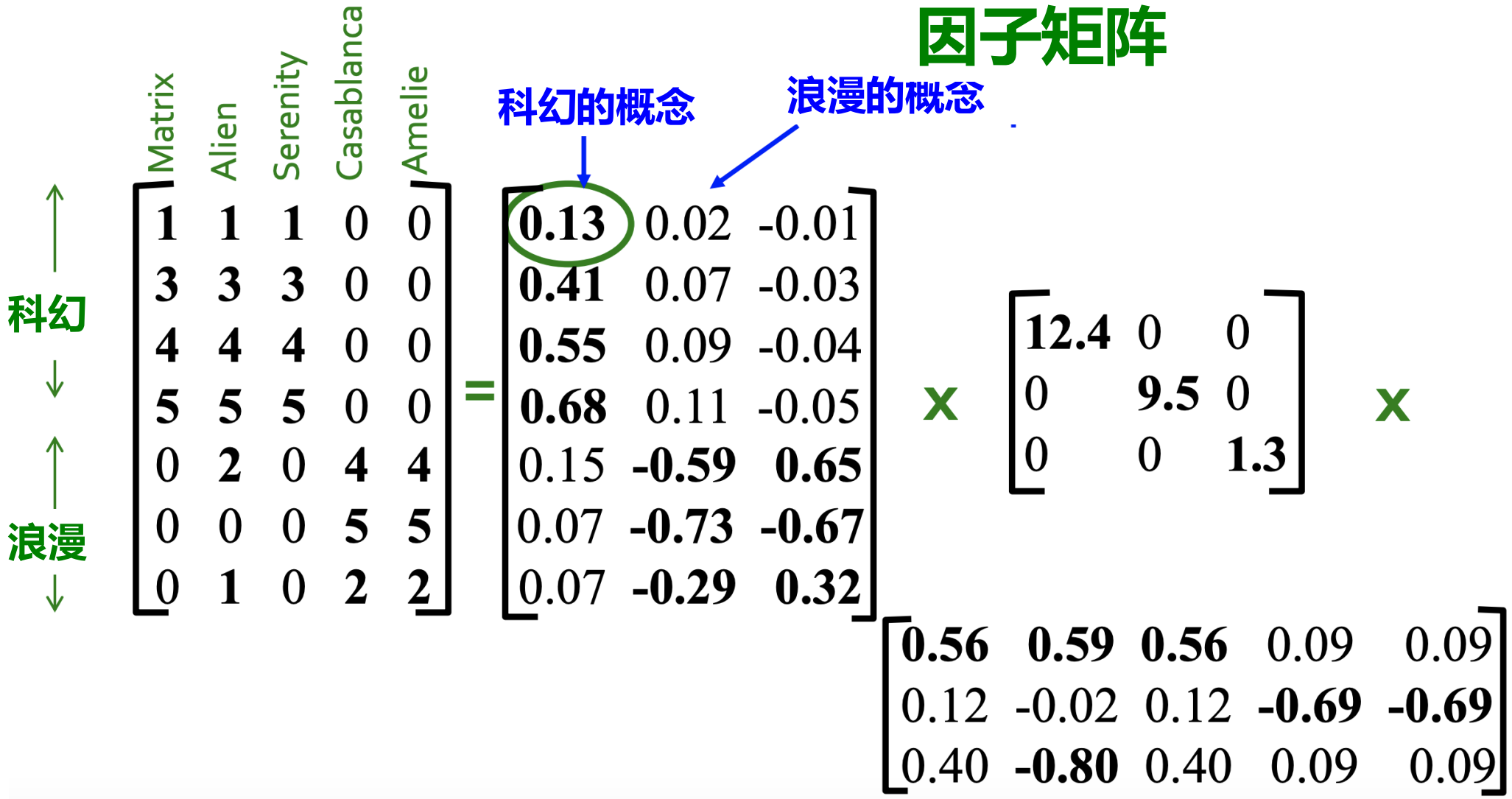
- $A = U \Sigma V^T$ - 示例：用户-电影



示例：用户-电影

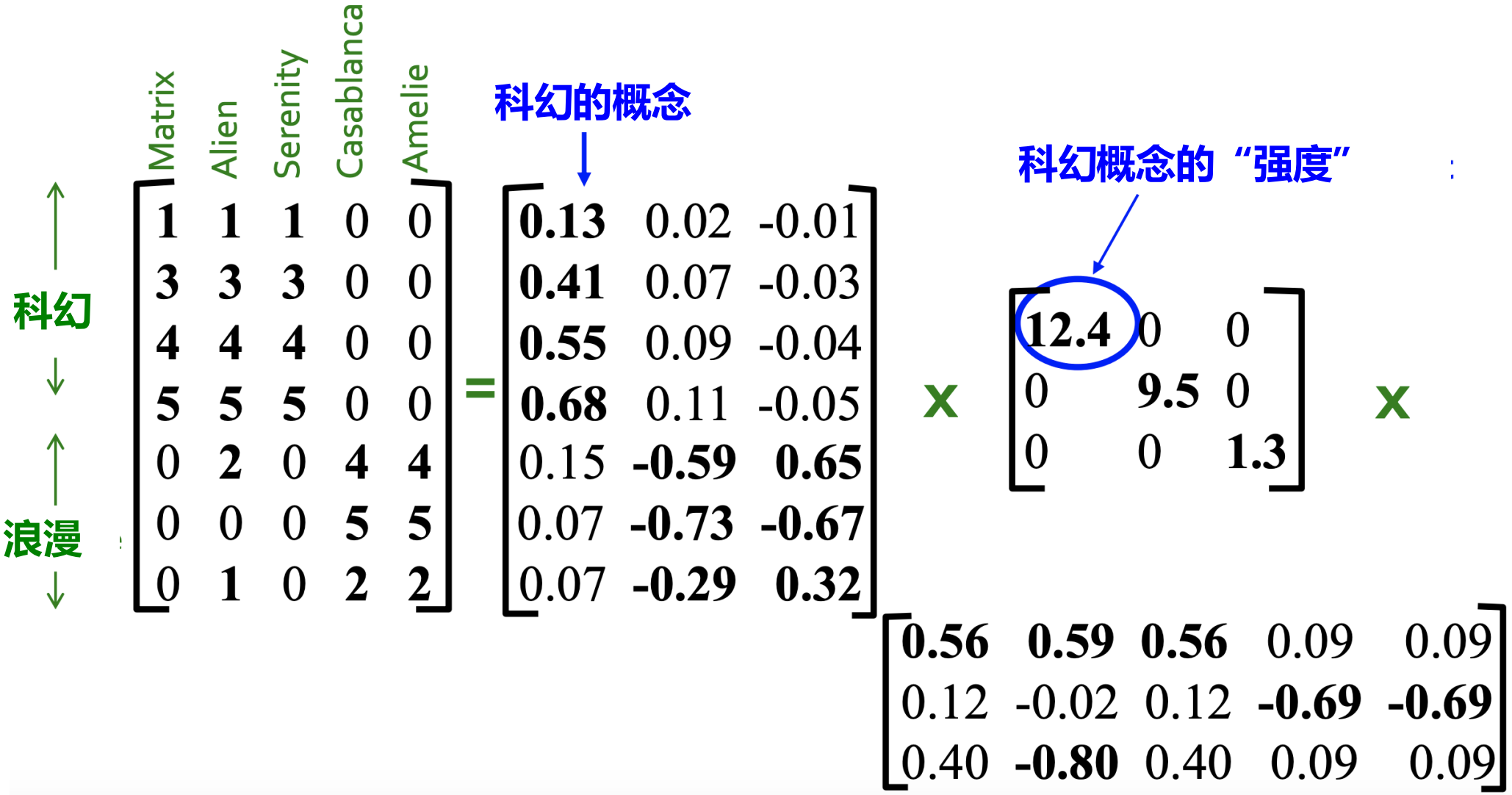
- $A = U \Sigma V^T$

U 是“用户到概念”因子矩阵



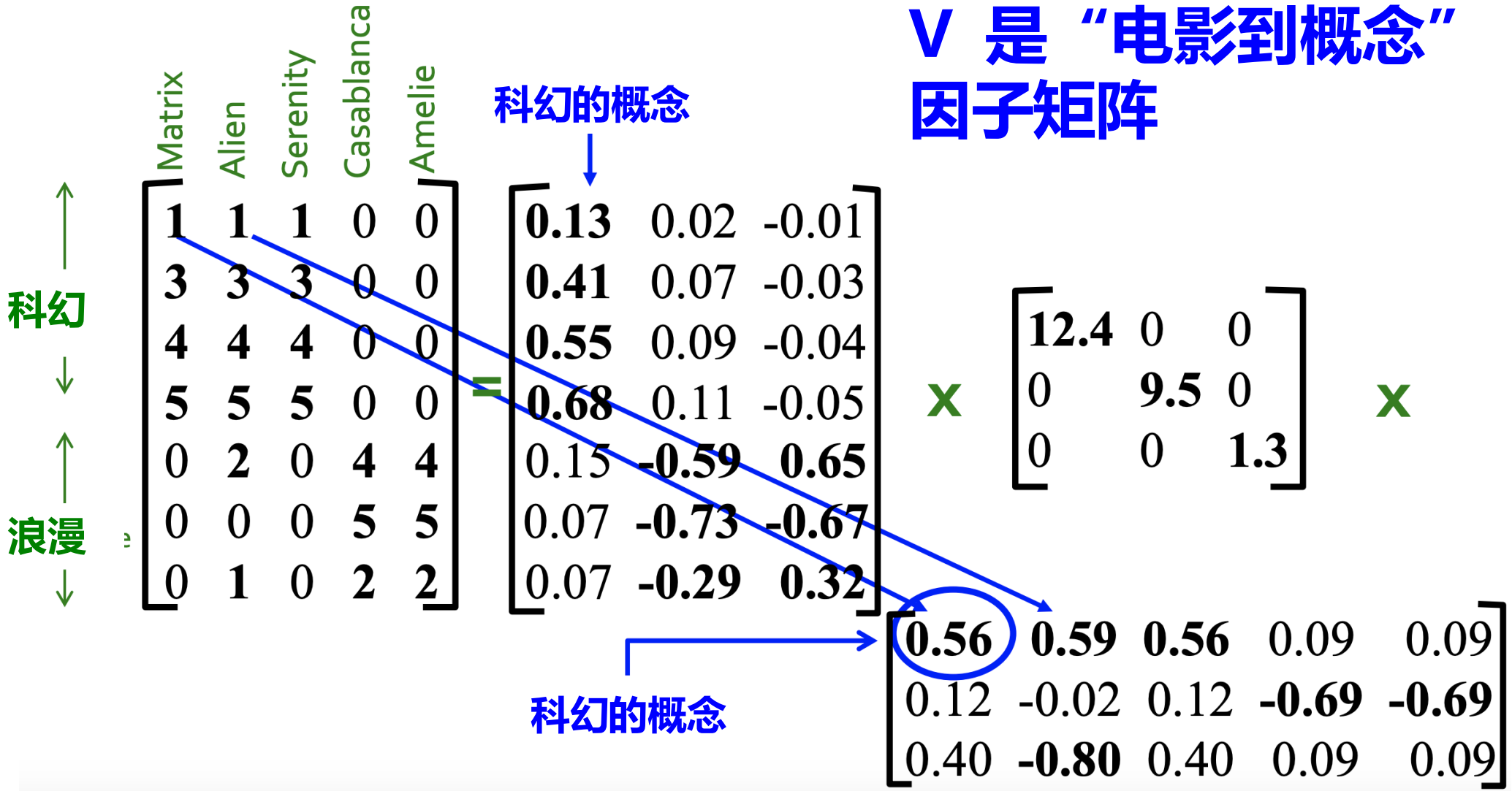
示例：用户-电影

- $A = U \Sigma V^T$ - 示例：



示例：用户-电影

- $A = U \Sigma V^T$ - 示例：



V 是“电影到概念”因子矩阵

SVD：解释

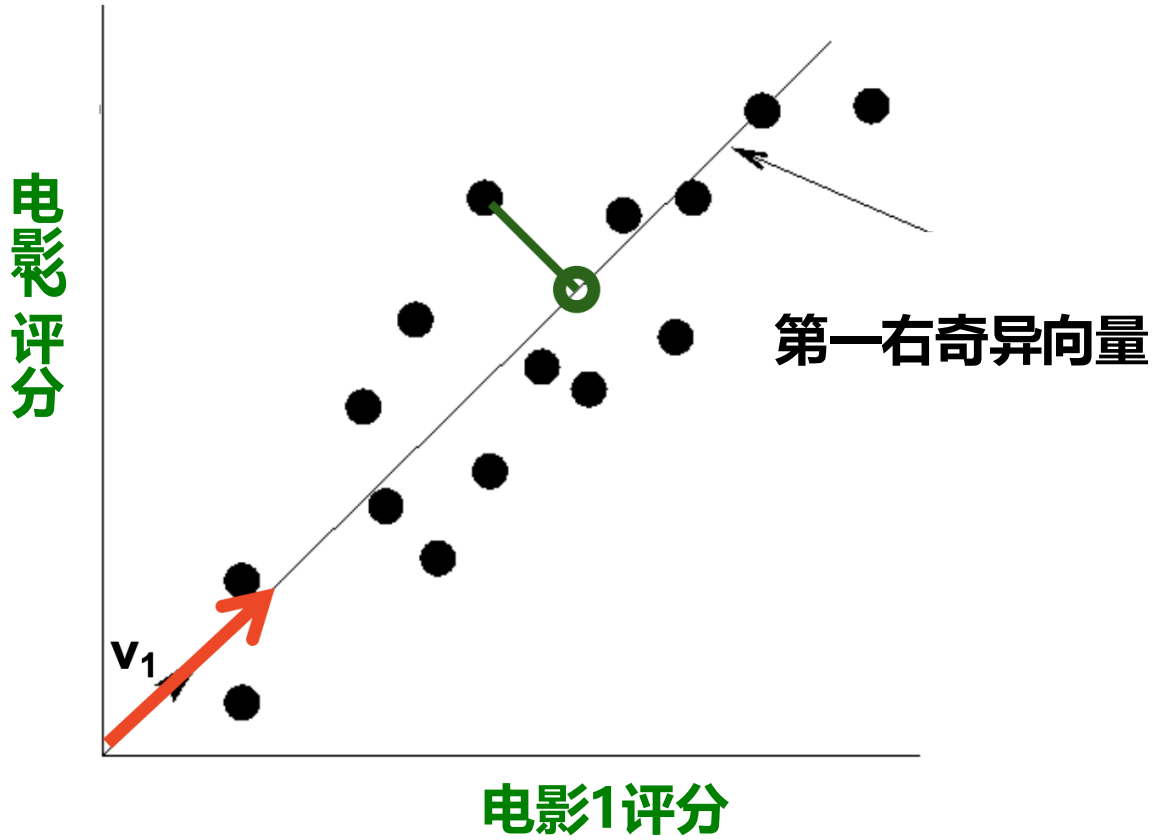
电影、**用户**和**概念**:

- **U:** 用户到概念矩阵
- **V:** 电影到概念矩阵
- **Σ:** 对角线元素:

  每个概念的“强度”

SVD降维

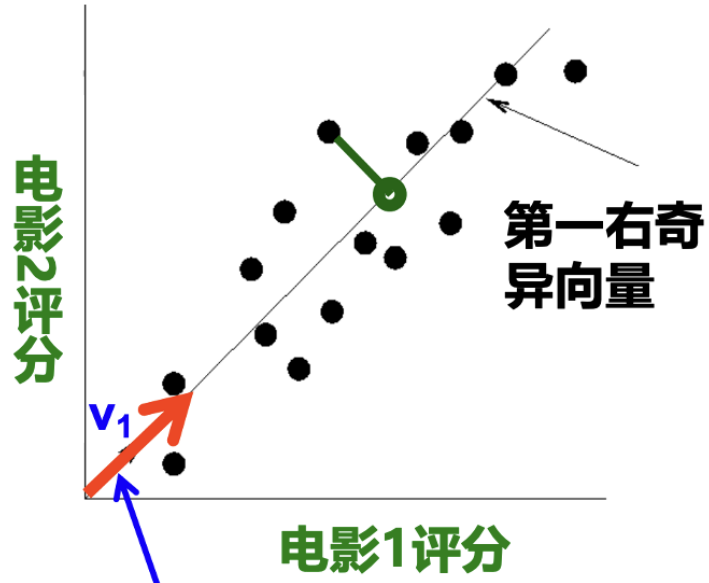
SVD：降维



- 我们不再使用两个坐标 (x, y) 来描述点的位置，而是只使用一个坐标
- 点的位置就是它沿着向量 v_1 的位置

SVD：降维

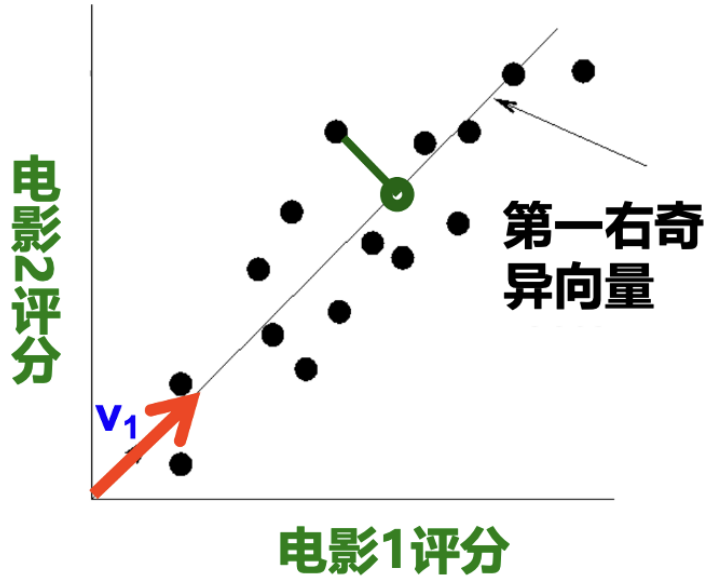
- $A = U \Sigma V^T$ - 示例：
  - U: “用户到概念”矩阵
  - V: “电影到概念”矩阵



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD：降维

- $A = U \Sigma V^T$ - 示例：



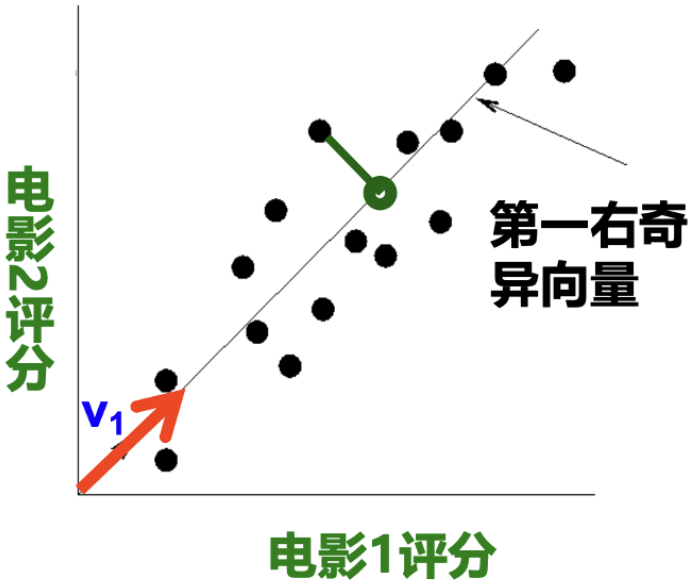
v_1 轴上的方差(散布)

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD：降维

$A = U \Sigma V^T$ - 示例：

- $U \Sigma$：给出投影轴上点的坐标



$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix}$$

用户在“科幻”轴上的投影

$U \Sigma$:

$$\begin{bmatrix} 1.61 & 0.19 & -0.01 \\ 5.08 & 0.66 & -0.03 \\ 6.82 & 0.85 & -0.05 \\ 8.43 & 1.04 & -0.06 \\ 1.86 & -5.60 & 0.84 \\ 0.86 & -6.93 & -0.87 \\ 0.86 & -2.75 & 0.41 \end{bmatrix}$$

SVD：解释

更多细节

- 问：到底如何进行降维?

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD：解释

更多细节

- 问：到底如何进行降维?
- 答：将最小奇异值设为零

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} = \begin{bmatrix} \mathbf{0.13} & 0.02 & -0.01 \\ \mathbf{0.41} & 0.07 & -0.03 \\ \mathbf{0.55} & 0.09 & -0.04 \\ \mathbf{0.68} & 0.11 & -0.05 \\ 0.15 & \mathbf{-0.59} & \mathbf{0.65} \\ 0.07 & \mathbf{-0.73} & \mathbf{-0.67} \\ 0.07 & \mathbf{-0.29} & \mathbf{0.32} \end{bmatrix} \times \begin{bmatrix} \mathbf{12.4} & 0 & 0 \\ 0 & \mathbf{9.5} & 0 \\ 0 & 0 & \mathbf{1.3} \end{bmatrix} \times \begin{bmatrix} \mathbf{0.56} & \mathbf{0.59} & \mathbf{0.56} & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & \mathbf{-0.69} & \mathbf{-0.69} \\ 0.40 & \mathbf{-0.80} & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD：解释

这是A的2阶近似，我们也可以做1阶近似。秩越大，近似越准确。

更多细节

- 问：到底如何进行降维?
- 答：将最小奇异值设为零

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.13 & 0.02 & -0.01 \\ 0.41 & 0.07 & -0.03 \\ 0.55 & 0.09 & -0.04 \\ 0.68 & 0.11 & -0.05 \\ 0.15 & -0.59 & 0.65 \\ 0.07 & -0.73 & -0.67 \\ 0.07 & -0.29 & 0.32 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 & 0 \\ 0 & 9.5 & 0 \\ 0 & 0 & 1.3 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \\ 0.40 & -0.80 & 0.40 & 0.09 & 0.09 \end{bmatrix}$$

SVD：解释

这是A的2阶近似，我们也可以做1阶近似。秩越大，近似越准确。

更多细节

- 问：到底如何进行降维?
- 答：将最小奇异值设为零

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.13 & 0.02 \\ 0.41 & 0.07 \\ 0.55 & 0.09 \\ 0.68 & 0.11 \\ 0.15 & -0.59 \\ 0.07 & -0.73 \\ 0.07 & -0.29 \end{bmatrix} \times \begin{bmatrix} 12.4 & 0 \\ 0 & 9.5 \end{bmatrix} \times \begin{bmatrix} 0.56 & 0.59 & 0.56 & 0.09 & 0.09 \\ 0.12 & -0.02 & 0.12 & -0.69 & -0.69 \end{bmatrix}$$

SVD：解释

这是A的2阶近似，我们也可以做1阶近似。秩越大，近似越准确。

更多细节

- 问：到底如何进行降维?
- 答：将最小奇异值设为零

$$\begin{bmatrix} 1 & 1 & 1 & 0 & 0 \\ 3 & 3 & 3 & 0 & 0 \\ 4 & 4 & 4 & 0 & 0 \\ 5 & 5 & 5 & 0 & 0 \\ 0 & 2 & 0 & 4 & 4 \\ 0 & 0 & 0 & 5 & 5 \\ 0 & 1 & 0 & 2 & 2 \end{bmatrix} \approx \begin{bmatrix} 0.92 & 0.95 & 0.92 & 0.01 & 0.01 \\ 2.91 & 3.01 & 2.91 & -0.01 & -0.01 \\ 3.90 & 4.04 & 3.90 & 0.01 & 0.01 \\ 4.82 & 5.00 & 4.82 & 0.03 & 0.03 \\ 0.70 & 0.53 & 0.70 & 4.11 & 4.11 \\ -0.69 & 1.34 & -0.69 & 4.78 & 4.78 \\ 0.32 & 0.23 & 0.32 & 2.01 & 2.01 \end{bmatrix}$$

重构数据矩阵B

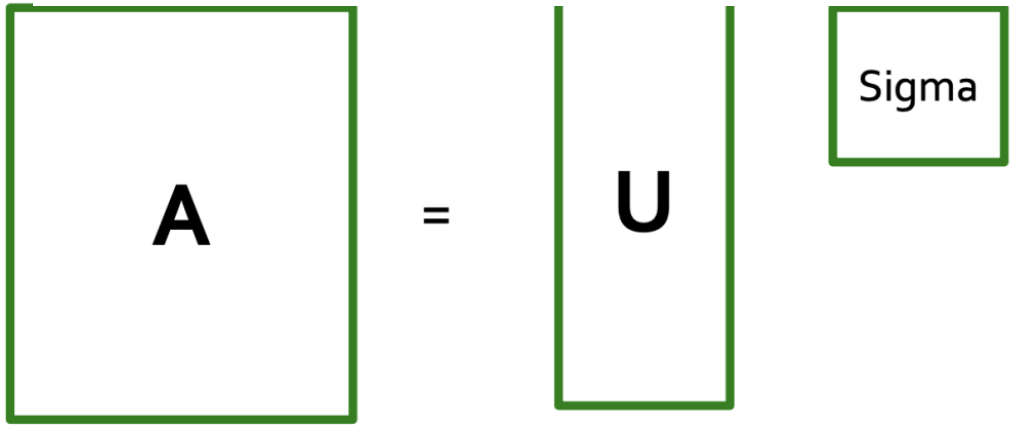
重构误差由Frobenius范数量化:

$$\|M\|_F = \sqrt{\sum_{ij} M_{ij}^2}$$

$$\|A - B\|_F = \sqrt{\sum_{ij} (A_{ij} - B_{ij})^2}$$ 很小

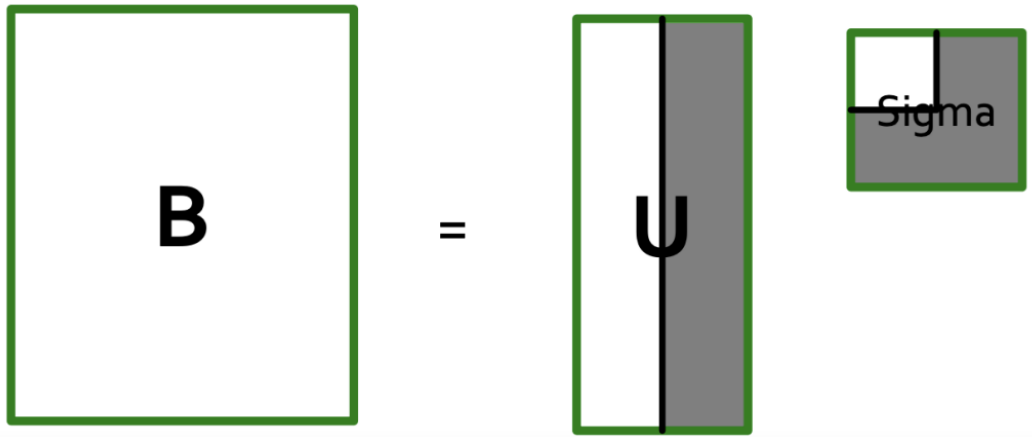
SVD-最佳低秩近似

- **事实: SVD提供了“最佳”投影轴:**
  - **‘最佳'** =最小化重构误差总和



$$\|A - B\|_F = \sqrt{\sum_{ij} (A_{ij} - B_{ij})^2}$$

B是A的最佳近似:



SVD-迄今为止的结论

- **SVD: $A = U\Sigma V^T$: 唯一**
  - **U:** 用户到概念的因素
  - **V:** 电影到概念的因素
  - **Σ:** 每个概念的强度

- **问: 那么 r（潜在因素的数量）的合适值是多少呢?**
- 设一组奇异值的能量是它们的平方和。
- 选取r，使保留的奇异值至少占总能量的90%。

- **回到我们的例子:**
  - 当奇异值为12.4, 9.5和1.3时，总能量= 245.7
  - 如果我们下降1.3，其平方只有1.7，那么我们剩下的能量是244，或者说超过总能量的99%以上

如何计算SVD

如何计算SVD?

- **首先假设** $A = U\Sigma V^T$
- $A^T = (U\Sigma V^T)^T = (V^T)^T \Sigma^T U^T = V\Sigma U^T$
  - **为什么?** (1)乘积的转置规则;(2)转置矩阵的转置和对角矩阵的转置都是恒等函数（或恒等变换）
- $A^T A =$

如何计算SVD?

- **首先假设**$A = U\Sigma V^T$
- $A^T = (U\Sigma V^T)^T = (V^T)^T \Sigma^T U^T = V\Sigma U^T$
  - **为什么?** (1)乘积的转置规则;(2)转置矩阵的转置和对角矩阵的转置都是恒等函数（或恒等变换）
- $A^T A = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T$
  - **为什么?** U 是标准正交的，因此 $U^T U$ 是单位矩阵。
  - 此外，注意Σ^2是一个对角矩阵，其第 i 个元素是Σ的第 i 个元素的平方
- $A^T AV =$

如何计算SVD?

- **首先假设**$A = U\Sigma V^T$
- $A^T = (U\Sigma V^T)^T = (V^T)^T \Sigma^T U^T = V\Sigma U^T$
  - **为什么?** (1)乘积的转置规则;(2)转置矩阵的转置和对角矩阵的转置都是恒等函数（或恒等变换）
- $A^T A = V\Sigma U^T U\Sigma V^T = V\Sigma^2 V^T$
  - **为什么?** U 是标准正交的，因此 $U^T U$ 是单位矩阵。
  - 此外，注意Σ^2是一个对角矩阵，其第 i 个元素是Σ的第 i 个元素的平方
- $A^T AV = V\Sigma^2 V^T V = V\Sigma^2$
  - **为什么?** V 也是标准正交的

如何计算SVD?

- 因为$A^T A = V\Sigma^2 V^T \rightarrow A^T AV = V\Sigma^2$
  - 所以V的第 i 列是 $A^T A$ 的一个特征向量，其特征值是Σ^2的第 i 个元素

- 因此，我们可以通过寻找 $A^T A$ 的特征对的方式来找到V和Σ
  - 一旦我们得到了Σ^2中的特征值，我们可以通过对这些特征值开平方来得到奇异值

-

如何计算SVD?

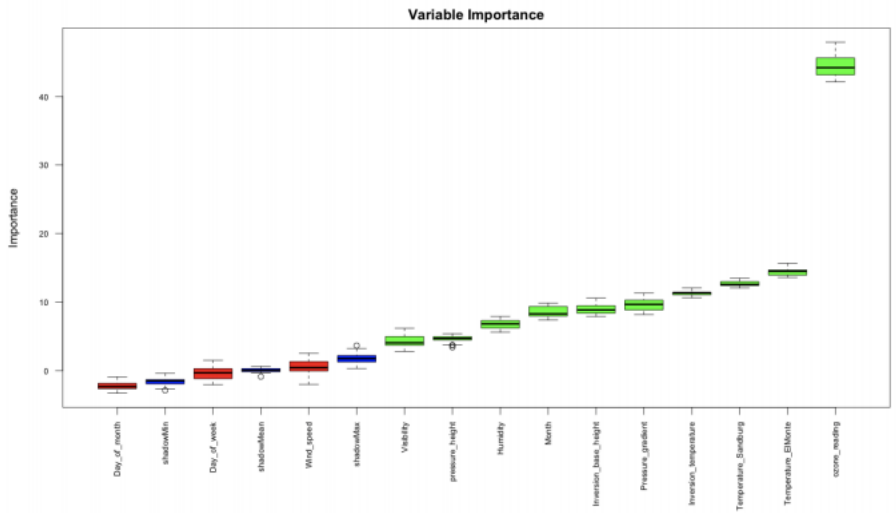
- 因为$A^TA = V\Sigma^2V^T \rightarrow A^TAV = V\Sigma^2$
  - 所以V的第 i 列是 A^TA 的一个特征向量，其特征值是Σ^2的第 i 个元素

- 因此，我们可以通过寻找 A^TA 的特征对的方式来找到V和Σ
  - 一旦我们得到了Σ^2中的特征值，我们可以通过对这些特征值开平方来得到奇异值

- 对称论证，AA^T 推出U

属性子集选择

- 另一种降低数据维度的方法
- 冗余属性
  - 复制一个或多个其他属性中包含的大部分或全部信息
    - 例如，产品的购买价格和缴纳的销售税金额
- 无关属性
  - 不包含对当前数据挖掘任务有用的信息
    - 例：学生ID通常对预测他/她的GPA无用



寻找特征对

- **如何实际计算SVD？**

- 首先，我们需要一种方法求对称矩阵的**主特征值**（最大特征值）和相应**特征向量**
  - 如果对于所有 i 和 j ，都有 $m_{ij} = m_{ji}$ ，则称 M 是*对称的*
- **方法： 施密特正交化**
  - 从任意“猜测特征向量” x_0 开始
  - 为$k = 0, 1, \ldots$ 构造 $x_{k+1} = \frac{Mx_k}{||Mx_k||}$
    - || ... ||表示Frobenius范数
  - 当临近步骤的 x_k 相差足够小时停止

幂迭代法

例子：迭代特征向量

$$M = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \quad \mathbf{x}_0 = \begin{matrix} 1 \\ 1 \end{matrix}$$

$$\frac{M\mathbf{x}_0}{||M\mathbf{x}_0||} = \qquad\qquad = \mathbf{x}_1$$

.....

例子：迭代特征向量

$$M = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \quad \mathbf{x}_0 = \begin{matrix} 1 \\ 1 \end{matrix}$$

$$\frac{M\mathbf{x}_0}{||M\mathbf{x}_0||} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} / \sqrt{34} = \begin{pmatrix} 0.51 \\ 0.86 \end{pmatrix} = \mathbf{x}_1$$

$$\frac{M\mathbf{x}_1}{||M\mathbf{x}_1||} = \qquad\qquad = \mathbf{x}_2$$

.....

例子：迭代特征向量

$$M = \begin{pmatrix} 1 & 2 \\ 2 & 3 \end{pmatrix} \quad \mathbf{x}_0 = \begin{matrix} 1 \\ 1 \end{matrix}$$

$$\frac{M\mathbf{x}_0}{||M\mathbf{x}_0||} = \begin{pmatrix} 3 \\ 5 \end{pmatrix} / \sqrt{34} = \begin{pmatrix} 0.51 \\ 0.86 \end{pmatrix} = \mathbf{x}_1$$

$$\frac{M\mathbf{x}_1}{||M\mathbf{x}_1||} = \begin{pmatrix} 2.23 \\ 3.60 \end{pmatrix} / \sqrt{17.93} = \begin{pmatrix} 0.53 \\ 0.85 \end{pmatrix} = \mathbf{x}_2$$

$$\mathbf{X}_3 = ?$$

…..

例子：迭代特征向量

- 一旦有了主特征向量 $\boldsymbol{x}$，你就可以通过$\boldsymbol{\lambda = x^T Mx}$找到它的特征值$\boldsymbol{\lambda}$。
  - **证明：** 我们知道如果 $\boldsymbol{\lambda}$ 是特征值，则 $\boldsymbol{x\lambda = Mx}$ ； 两边同时左乘 $\boldsymbol{x^T}$
  - 因为 $\boldsymbol{x^T x = 1}$，所以 $\boldsymbol{\lambda = x^T Mx}$

- **例子：** 如果取 $\boldsymbol{x^T = [0.53,\ 0.85]}$，则

$$\lambda = [0.53\ 0.85] \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} = 4.25$$

寻找更多特征向量

- 去掉矩阵 M 中可以由第一个特征对 λ 和 x 生成的部分：

$$\boldsymbol{M^*} := \boldsymbol{M} - \boldsymbol{\lambda x x^T}$$

- 递归地找到$\boldsymbol{M^*}$的主特征对，消除该特征对的影响，以此类推

- **例子：**

$$M^* = \begin{bmatrix} 1 & 2 \\ 2 & 3 \end{bmatrix} - 4.25 \begin{bmatrix} 0.53 \\ 0.85 \end{bmatrix} [0.53 \quad 0.85] = \begin{bmatrix} -0.19 & 0.09 \\ 0.09 & 0.07 \end{bmatrix}$$

属性选择中的启发式搜索

- d个属性中有2^d个可能的属性组合
- 典型的启发式属性选择方法：
  - 属性独立性假设下的最佳单属性：通过显著性检验进行选择
  - 最佳分步特征选择：
    - 首先选出最好的单一属性
    - 然后找到第二最佳属性，....
  - 逐步消除属性：
    - 反复消除最差属性
  - 最优属性选择和消除的组合
  - 最优分支定界：
    - 使用属性消除和回溯

属性创建(特征生成)

- 创建新的属性(特征)，使其比原始属性更有效地捕获数据集中的重要信息
- 3种通用方法
  - 属性提取
    - 特定领域的
  - 将数据映射到新空间(参见：数据缩减)
    - 例如，傅里叶变换, 小波变换, 流形方法(未介绍)
  - 属性结构
    - 组合特征（参见“高级分类”一章中的判别频繁模式）
    - 数据离散化

总结

- **数据质量：**准确性, 完整性, 一致性, 时效性, 可信度, 可解释性
- **数据清洗：**例如，缺失值/噪声, 异常值
- 多个源的**数据集成**：
  - 实体识别问题；冗余消除；不一致检测
- **数据缩减, 数据变换和数据离散化**
  - 数量规约；数据压缩
  - 归一化；概念层次生成
- **维度规约（降维）**