大数据分析与挖掘

课程PPT可以从以下网址下载：

http://jz81.github.io/course/id/id.html

“大数据”的5V特性

巨量Volume

- 每天产生 **1,000,000^{3}B**数据
- Flickr每天新增 **180万**张照片 (TB级)
- Facebook、Twitter每天新增**>10TB**
- **90%**的数据产生于过去**2**年中 (增速加剧)

高速化Velocity

- 强子对撞机LHC：**700MB/sec**
- 平方千米阵列射电望远镜SKA：**40GB/sec**

多样Variety

- 文本 图像 视频 机器数据

价值Value

真实性Veracity

大数据的特性——HACE理论

HACE理论

2

Autonomous
- 自治的分布式数据源
- 去中心化的控制

Complex
- 复杂的数据类型和表示
- 复杂的数据间关系

3

4

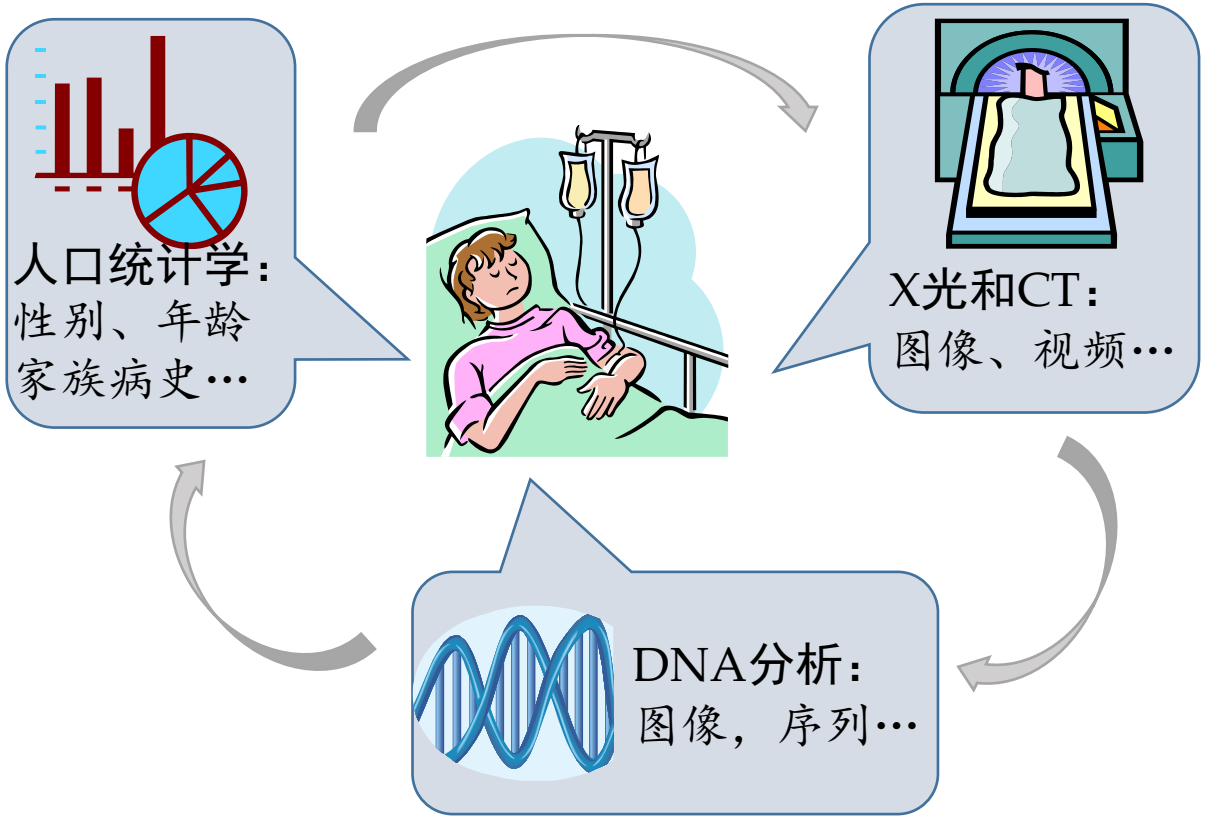
Evolving
- 数据类型的不断丰富
- 数据量的动态增长
- 数据间关系的动态演化

Heterogeneous
- 异构的数据类型
- 稀疏的维度

1

巨量的异构(H)离散维度数据

多种数据收集源 （多源异构数据融合）
使用不同的模式(schemata)收集数据



分布式、去中心化自治(A)数据源

同一应用的数据源去中心化控制

不同数据源松耦合、功能叠加、全局容错

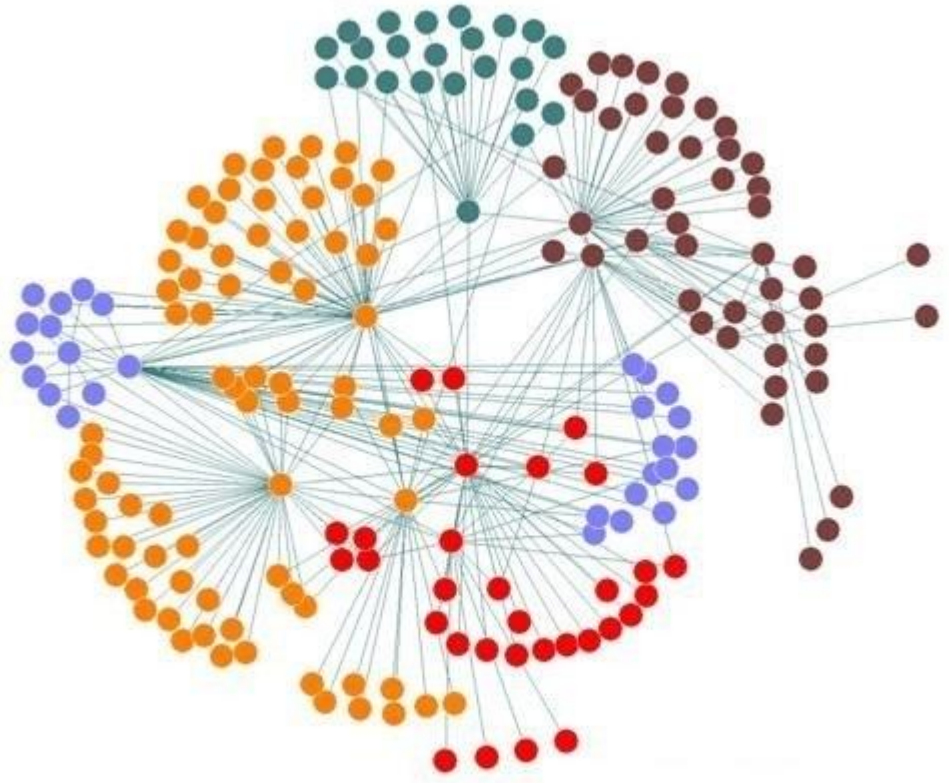
数据挖掘结果的差异性

- 地区差异
- 宗教文化差异
- 购买行为差异
- 促销策略差异
- 其它更多…

复杂(C)和演化(E)的数据间关系

“样本–特征”表示(Sample-feature)

“样本-特征-关系”表示(Sample-feature-relationship)



数据表示复杂

数据关系复杂

动态性强

演化规律复杂

盲人摸象的“大数据”挖掘

“盲人”们之间可以相互交流，互相学习对方的知识

“大数据”挖掘的挑战(1)

多信息源的本地学习与模型融合

- 挑战 ❶数据传输代价
  ❷隐私保护
  ❸不同站点的“偏见”
- 目标 分布式数据源协同挖掘
  获得全局最优化结果
- 方法 ❶本地挖掘（local mining）
  ❷全局相关（global correlation）
  ❸数据、模型、知识层

“大数据”挖掘的挑战(2)

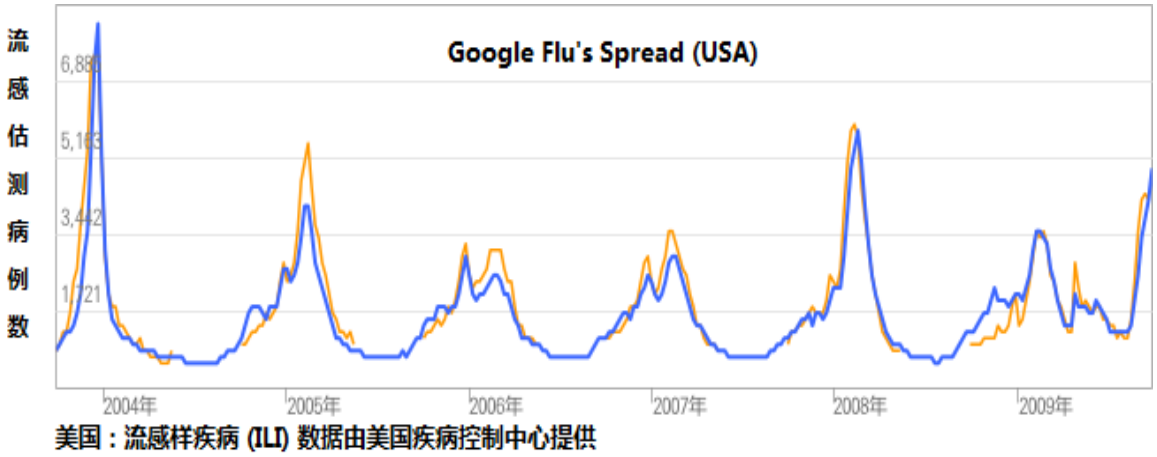
稀疏、不确定、不完整数据的挖掘

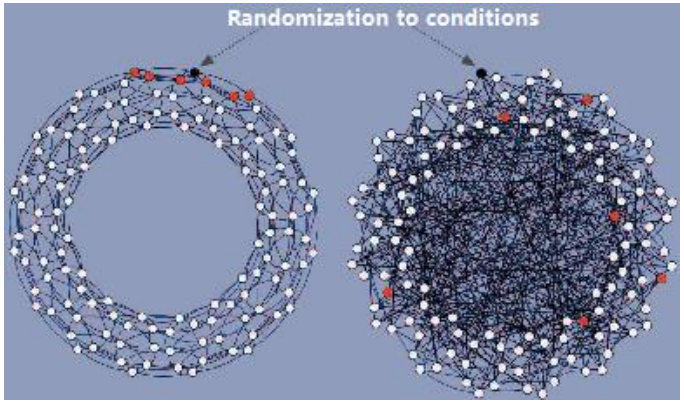
- 挑战
  - ❶特征稀疏(Sparse)： 高维
    趋势/分布不清
  - ❷不确定(Uncertain)： 随机/错误分布
  - ❸不完整(Incomplete)： 数据缺失
- 方法
  - ❶维度约减与特征选择
  - ❷error-aware的挖掘
  - ❸imputation方法（缺失值处理）

“大数据”挖掘的挑战(3)

复杂、动态数据的挖掘

- 挑战 ❶结构化与非结构化数据并存
  ❷动态复杂庞大的网络结构（如：社交数据）
  ❸特定应用相关
- 成果 ❶Google流感疫情预测
  ❷在线社交网络行为传播



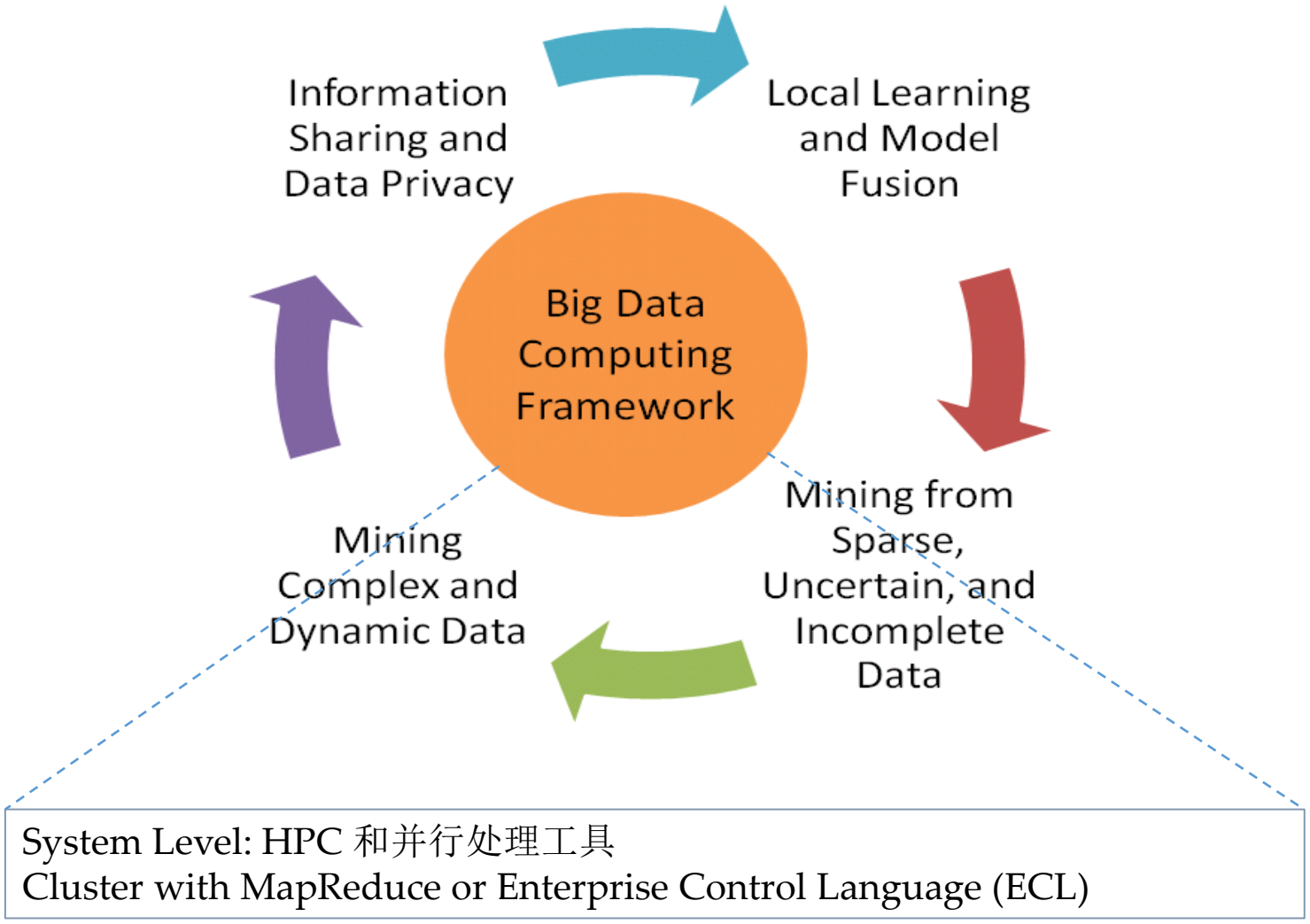


“大数据”挖掘的挑战(4)

信息共享与数据隐私保护

- 挑战 ❶保护用户隐私（信用卡、健康医疗 ... ）
  ❷实现信息共享（知识发现、共享与集成）
- 方法 ❶隐私保护
  限制敏感信息访问 （访问控制、认证机制 … ）
  匿名化敏感数据 （数据注入随机性）
  ❷隐私保护的数据挖掘
  使用特殊通信协议避免访问个体数据
  设计匿名数据的挖掘方法

“大数据”挖掘的处理框架



System Level: HPC 和并行处理工具
Cluster with MapReduce or Enterprise Control Language (ECL)

关联规则分析

关联规则挖掘的含义

- 关联规则用于表示数据库中诸多属性（项集）之间的关联程度。
- 关联规则挖掘（ **Association Rules Mining**）则是利用数据库中的大量数据通过关联算法寻找属性间的相关性。
- 应用：
  - 购物篮分析、分类设计、捆绑销售和亏本销售分析
  - **例：(超级市场)在购买商品A的客户中有90%的人会同时购买商品B，则可用关联规则表示为：**

$$A \rightarrow B$$

例子1

Frequently Bought Together

Price For All Three: $166.83

Add all three to Cart | Add all three to Wish List

Show availability and shipping details

- **This item:** Data Mining: Practical Machine Learning Tools and Techniques, Second Edition (Morgan Kaufmann Series in Data Management Systems) by Eibe Frank
- The Elements of Statistical Learning: Data Mining, Inference, and Prediction, Second Edition (Springer Series in Statistics) by Robert Tibshirani
- Pattern Recognition and Machine Learning (Information Science and Statistics) by Christopher M. Bishop

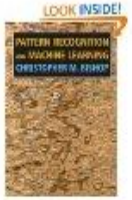
Customers Who Bought This Item Also Bought

Handbook of Statistical Analysis and Data Mining... by John Elder IV

★★★★½ (9) $71.96

Introduction to Data Mining by Pang-Ning Tan

★★★★½ (15) $80.80



Pattern Recognition and Machine Learning... by Christopher M. Bishop

★★★★☆ (47) $58.03

Machine Learning (Mcgraw-Hill International Edit) by Tom M. Mitchell

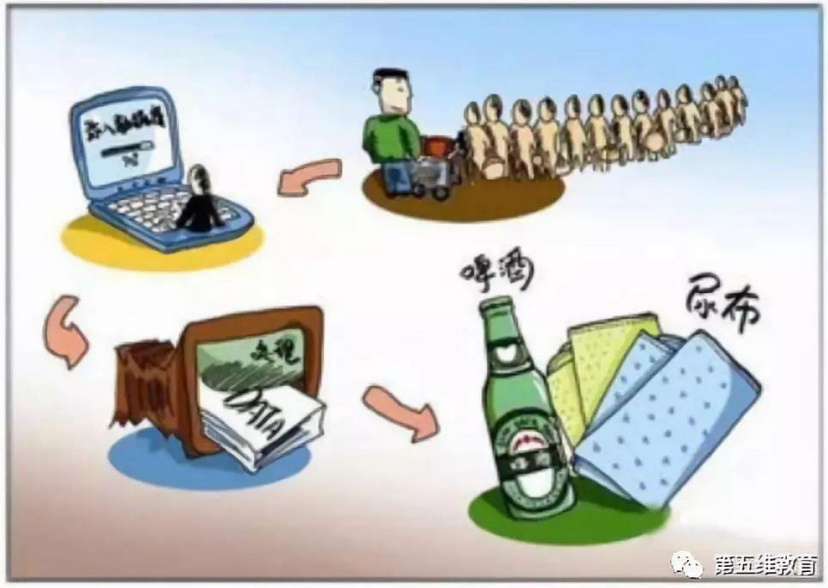
★★★★½ (38) $73.72

Introduction to Machine Learning (Adaptive Comp... by Ethem Alpaydin

★★★★½ (8) $43.79

例子2——尿布与啤酒

- 采用关联模型比较典型的案例是“尿布与啤酒”的故事。在美国，一些年轻的父亲下班后经常要到超市去买婴儿尿布，沃尔玛超市也因此发现了一个规律，在购买婴儿尿布的年轻父亲们中，有30%～40%的人同时要买一些啤酒。超市随后调整了货架的摆放，把尿布和啤酒放在一起，明显增加了销售额。同样的，我们还可以根据关联规则在商品销售方面做各种促销活动。



Association Rule Mining

- Given a set of transactions, find rules that will predict the occurrence of an item based on the occurrences of other items in the transaction

Market-Basket transactions

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Association Rules

{Diaper} $\rightarrow$ {Beer},
{Milk, Bread} $\rightarrow$ {Eggs,Coke},
{Beer, Bread} $\rightarrow$ {Milk},

Implication means co-occurrence, not causality!

Definition: Frequent Itemset（频繁项目集）

- **Itemset**
  - A collection of one or more items
    - Example: {Milk, Bread, Diaper}
  - k-itemset
    - An itemset that contains k items
- **Support count (σ)**
  - Frequency of occurrence of an itemset
  - E.g. σ({Milk, Bread,Diaper}) = 2
- **Support**
  - Fraction of transactions that contain an itemset
  - E.g. s({Milk, Bread, Diaper}) = 2/5
- **Frequent Itemset**
  - An itemset whose support is greater than or equal to a *minsup* threshold

TID	*Items*
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Definition: Association Rule

Association Rule

- An implication expression of the form $X \rightarrow Y$, where X and Y are itemsets
- Example:
  {Milk, Diaper} → {Beer}

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Rule Evaluation Metrics

- Support (s)
  - Fraction of transactions that contain both X and Y

  Support(X →Y)=P(XUY)

  $$Support(X \rightarrow Y) = \frac{\#\,(X \cup Y)}{n}$$
- Confidence (c)
  - Measures how often items in Y appear in transactions that contain X

  Confidence(X →Y)=P(Y|X)

$$Confidence(X \rightarrow Y) = \frac{Support(X \cup Y)}{Support(X)}$$

Example:

$$\{\text{Milk}, \text{Diaper}\} \Rightarrow \{\text{Beer}\}$$

$$s = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{|\text{T}|} = \frac{2}{5} = 0.4$$

$$c = \frac{\sigma(\text{Milk}, \text{Diaper}, \text{Beer})}{\sigma(\text{Milk}, \text{Diaper})} = \frac{2}{3} = 0.67$$

贝叶斯定理P(Beer|Milk, Diaper)

Association Rule Mining Task

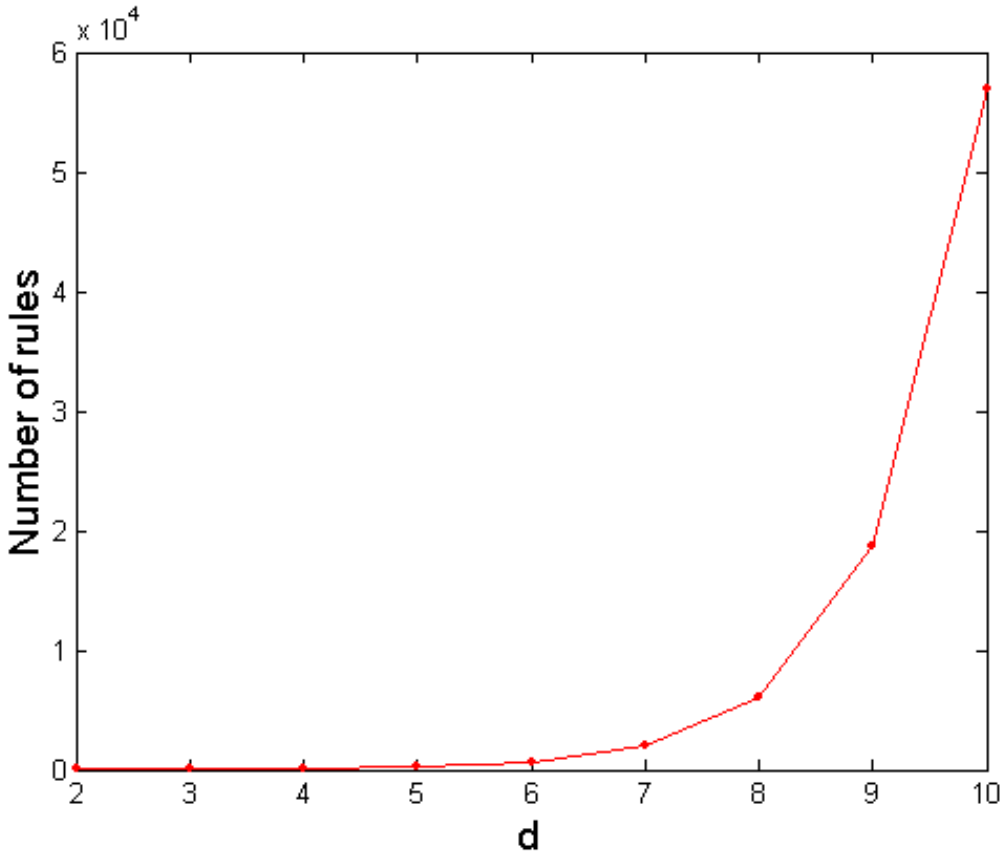
- Given a set of transactions T, the goal of association rule mining is to find all rules having
  - support ≥ *minsup* threshold
  - confidence ≥ *minconf* threshold

- Brute-force approach:
  - List all possible association rules
  - Compute the support and confidence for each rule
  - Prune rules that fail the *minsup* and *minconf* thresholds

  $\Rightarrow$ Computationally prohibitive!

Computational Complexity

- Given *d* unique items:
  - Total number of itemsets = 2^d
  - Total number of possible association rules:



$$R = \sum_{k=1}^{d-1}\left[\binom{d}{k} \times \sum_{j=1}^{d-k}\binom{d-k}{j}\right]$$

$$= 3^d - 2^{d+1} + 1$$

If $d=6$, $R = 602$ rules

Mining Association Rules

TID	Items
1	Bread, Milk
2	Bread, Diaper, Beer, Eggs
3	Milk, Diaper, Beer, Coke
4	Bread, Milk, Diaper, Beer
5	Bread, Milk, Diaper, Coke

Example of Rules:

{Milk,Diaper} → {Beer} (s=0.4, c=0.67)
{Milk,Beer} → {Diaper} (s=0.4, c=1.0)
{Diaper,Beer} → {Milk} (s=0.4, c=0.67)
{Beer} → {Milk,Diaper} (s=0.4, c=0.67)
{Diaper} → {Milk,Beer} (s=0.4, c=0.5)
{Milk} → {Diaper,Beer} (s=0.4, c=0.5)

Observations:

- All the above rules are binary partitions of the same itemset:
  {Milk, Diaper, Beer}
- Rules originating from the same itemset have identical support but can have different confidence.
- Thus, we may decouple the support and confidence requirements.

Mining Association Rules

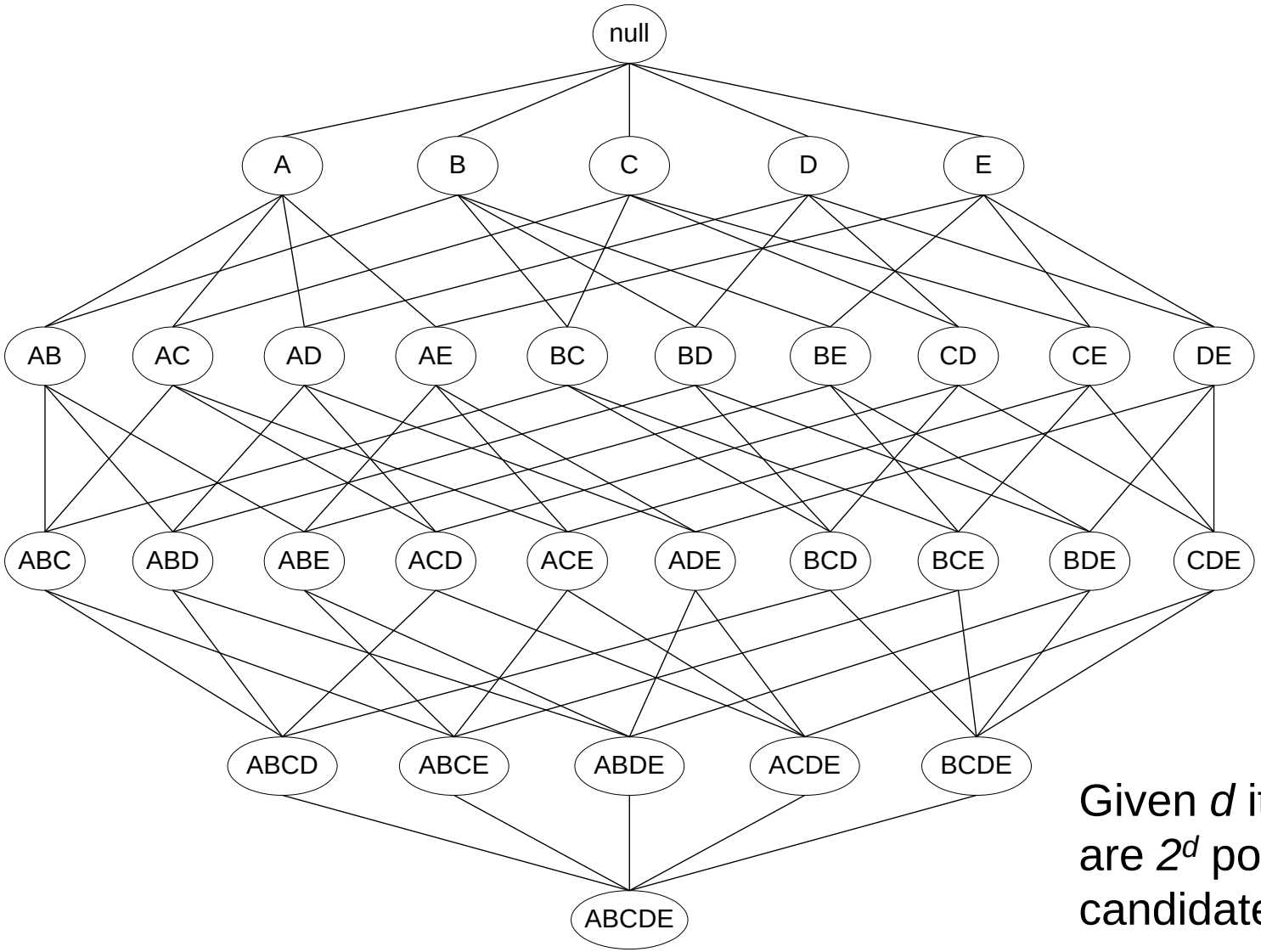
Two-step approach:

1. Frequent Itemset Generation
   - Generate all itemsets whose *support* $\geq$ *minsup*

2. Rule Generation
   - Generate high confidence rules from each frequent itemset, where each rule is a binary partitioning of a frequent itemset

Frequent itemset generation is still computationally expensive

Frequent Itemset Generation

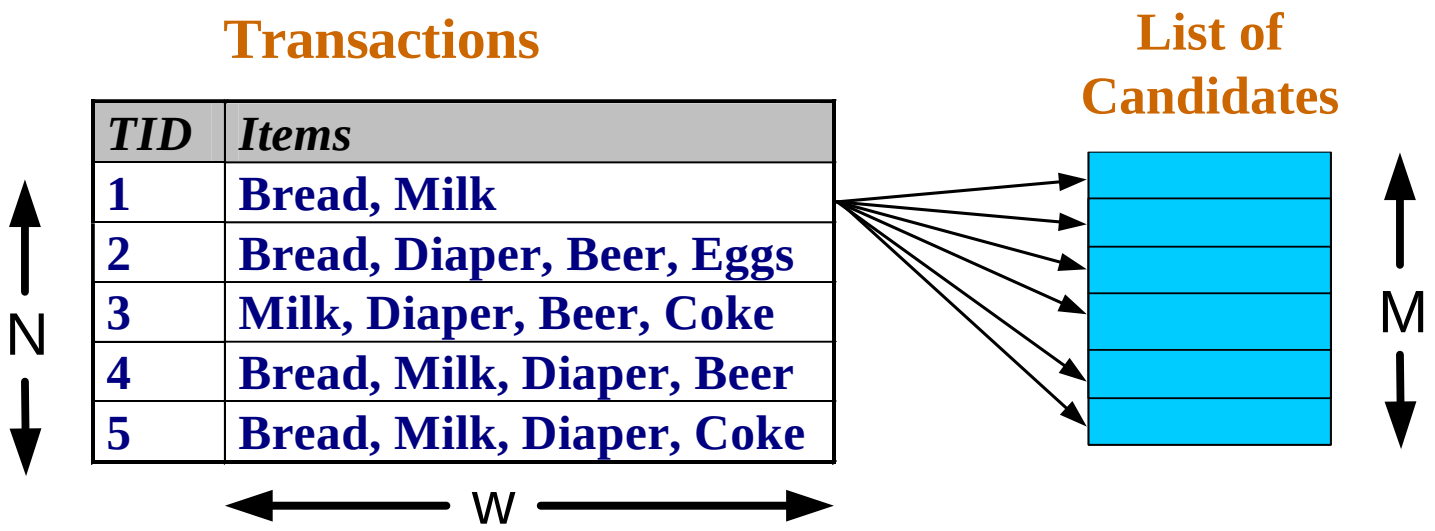


Given *d* items, there are 2^d possible candidate itemsets

Frequent Itemset Generation

Brute-force approach:

- Each itemset in the lattice is a candidate frequent itemset
- Count the support of each candidate by scanning the database



- Match each transaction against every candidate
- Complexity ~ O(NMw) => Expensive since M = 2^d !!!

Reducing Number of Candidates

Apriori principle:

- If an itemset is frequent, then all of its subsets must also be frequent

Apriori principle holds due to the following property of the support measure:

$$\forall X, Y : (X \subseteq Y) \Rightarrow s(X) \geq s(Y)$$

- Support of an itemset never exceeds the support of its subsets
- This is known as the anti-monotone property of support

Illustrating Apriori Principle

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Minimum Support = 3

Illustrating Apriori Principle

TID	*Items*
1	Bread, Milk
2	Beer, Bread, Diaper, Eggs
3	Beer, Coke, Diaper, Milk
4	Beer, Bread, Diaper, Milk
5	Bread, Coke, Diaper, Milk

Items (1-itemsets)

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Minimum Support = 3

Illustrating Apriori Principle

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset
{Bread,Milk}
{Bread, Beer }
{Bread,Diaper}
{Beer, Milk}
{Diaper, Milk}
{Beer,Diaper}

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

Illustrating Apriori Principle

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Beer, Bread}	2
{Bread,Diaper}	3
{Beer,Milk}	2
{Diaper,Milk}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

Illustrating Apriori Principle

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

Triplets (3-itemsets)

Itemset
{ Beer, Diaper, Milk}
{ Beer,Bread,Diaper}
{Bread,Diaper,Milk}
{ Beer, Bread, Milk}

Illustrating Apriori Principle

Item	Count
Bread	4
Coke	2
Milk	4
Beer	3
Diaper	4
Eggs	1

Items (1-itemsets)

Itemset	Count
{Bread,Milk}	3
{Bread,Beer}	2
{Bread,Diaper}	3
{Milk,Beer}	2
{Milk,Diaper}	3
{Beer,Diaper}	3

Pairs (2-itemsets)

(No need to generate candidates involving Coke or Eggs)

Minimum Support = 3

Triplets (3-itemsets)

Itemset	Count
{ Beer, Diaper, Milk}	2
{ Beer,Bread, Diaper}	2
{Bread, Diaper, Milk}	2
{Beer, Bread, Milk}	1

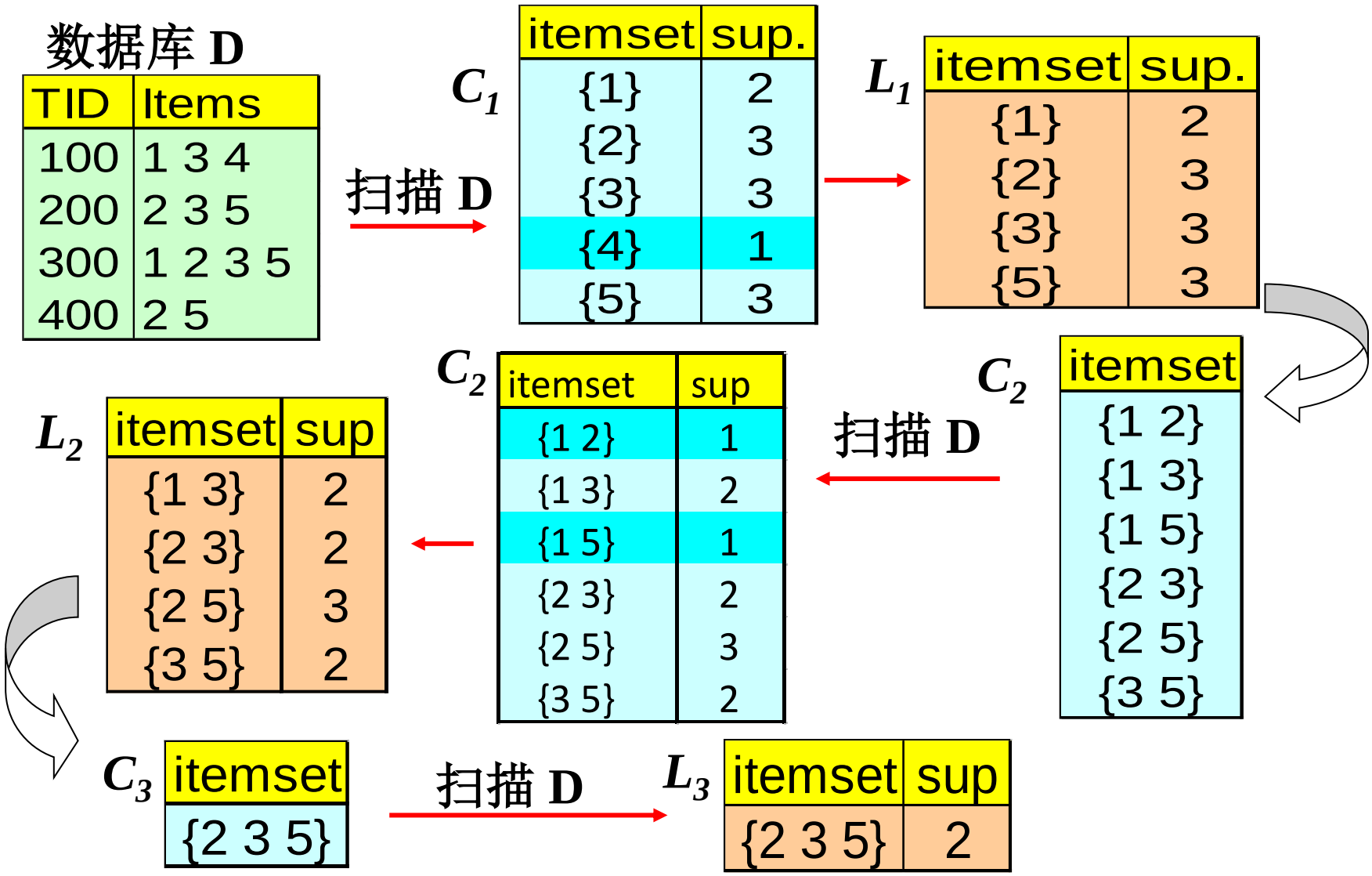
Apriori Algorithm

- F_k: frequent k-itemsets
- L_k: candidate k-itemsets

Algorithm

- Let k=1
- Generate F_1 = {frequent 1-itemsets}
- Repeat until F_k is empty
  - **Candidate Generation**: Generate L_{k+1} from F_k
  - **Candidate Pruning**: Prune candidate itemsets in L_{k+1} containing subsets of length k that are infrequent
  - **Support Counting**: Count the support of each candidate in L_{k+1} by scanning the DB
  - **Candidate Elimination**: Eliminate candidates in L_{k+1} that are infrequent, leaving only those that are frequent => F_{k+1}

另外一个例子（最小支持度2）



产生关联规则

• L_1 前面 … 一个频 … L_2 … }， 非空 L_3 … {3}，{5} … {3，5 …

itemset	sup.
{1}	2
{2}	3
{3}	3
{5}	3

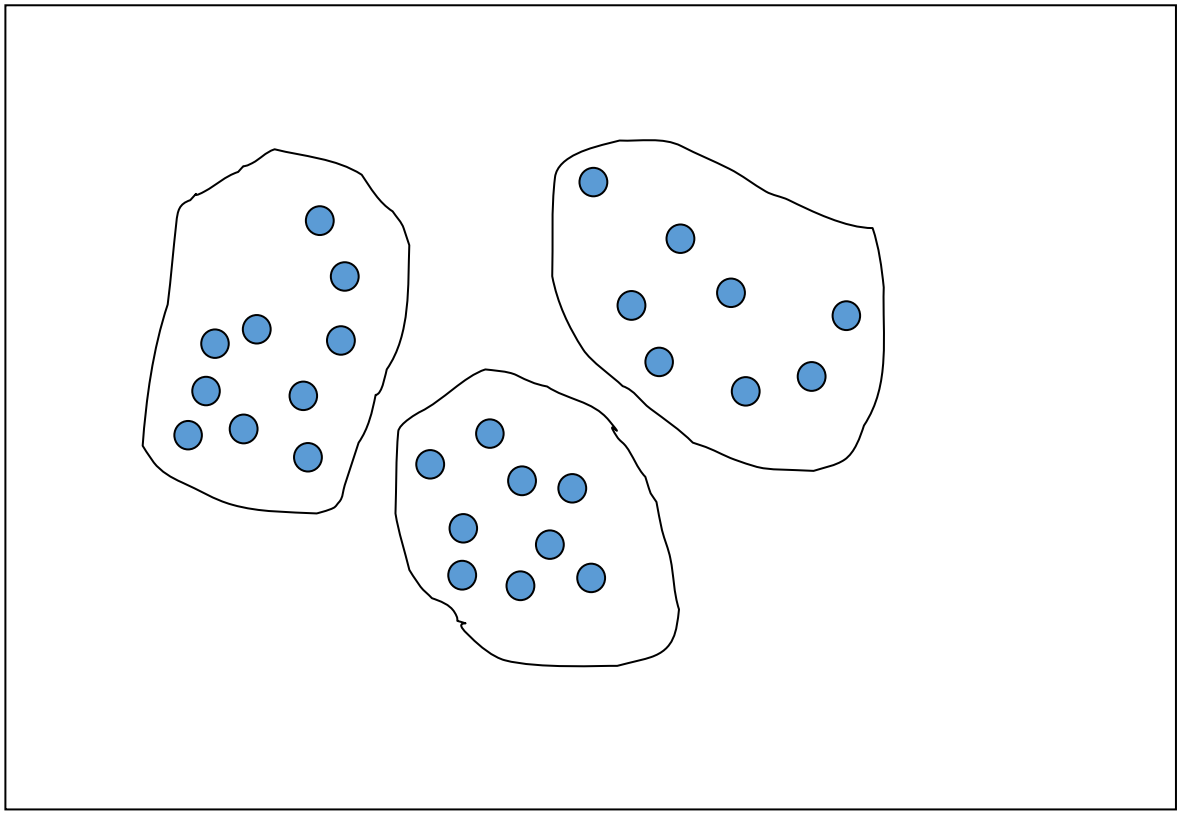
itemset	sup
{1 3}	2
{2 3}	2
{2 5}	3
{3 5}	2

itemset	sup
{2 3 5}	2

规则：	置信度：
$2 \Rightarrow 3 \wedge 5$	2/3=66%（{2，3，5}频度/{2}频度）
$3 \Rightarrow 2 \wedge 5$	2/3=66%（{2，3，5}频度/{3}频度）
$5 \Rightarrow 2 \wedge 3$	2/3=66%（{2，3，5}频度/{5}频度）
$2 \wedge 3 \Rightarrow 5$	2/2=100%（{2，3，5}频度/{2，3}频度）
$2 \wedge 5 \Rightarrow 3$	2/3=66%（{2，3，5}频度/{2，5}频度）
$3 \wedge 5 \Rightarrow 2$	2/2=100%（{2，3，5}频度/{3，5}频度）

支持度：2/4=50%

聚类分析



概述

- 聚类分析含义
  - 将物理或抽象对象的集合分组成为由类似的对象组成的多个类的过程称为聚类，由聚类所组成的簇是一组对象的集合，这些对象与同一簇中的对象彼此相似，与其它簇中的对象相异。
- 与分类不同，它要划分的类是未知的。

什么是好的聚类分析？

- 一个好的聚类分析方法会产生高质量的聚类
  - 高类内相似度
  - 低类间相似度
- 作为统计学的一个分支，聚类分析的研究主要是基于距离的聚类；一个高质量的聚类分析结果，将取决于所使用的聚类方法
  - 聚类方法的所使用的相似性度量和方法的实施
  - 方法发现隐藏模式的能力

类间距离

- 距离函数都是关于两个样本的距离刻画，然而在聚类应用中，最基本的方法是计算类间的距离。
- 设有两个类C_a和C_b，它们分别有m和h个元素，它们的中心分别为γ_a和γ_b。设元素$x \in C_a$，$y \in C_b$，这两个元素间的距离通常通过类间距离来刻画，记为$D(C_a, C_b)$。
- 类间距离的度量主要有：
  - 最短距离法：定义两个类中最靠近的两个元素间的距离为类间距离。
  - 最长距离法：定义两个类中最远的两个元素间的距离为类间距离。
  - 中心法：定义两类的两个中心间的距离为类间距离。
  - 类平均法：它计算两个类中任意两个元素间的距离，并且综合他们成为类间距离。

$$D_G(C_a, C_b) = \frac{1}{mh} \sum_{x \in C_a} \sum_{y \in C_b} d(x, y)$$

距离度量

（1） 欧几里德距离

$$d(i,j)=\sqrt{|x_{i1}-x_{j1}|^2+|x_{i2}-x_{j2}|^2+..+|x_{ip}-x_{jp}|^2}$$

（2） 曼哈顿距离

$$d(i,j)=|x_{i1}-x_{j1}|+|x_{i2}-x_{j2}|+..+|x_{ip}-x_{jp}|$$

（3） 明斯基距离

$$d(i,j)=\sqrt[q]{|x_{i1}-x_{j1}|^q+|x_{i2}-x_{j2}|^q+..+|x_{ip}-x_{jp}|^q}$$

中心法

- 中心法涉及到类的中心的概念。假如C_i是一个聚类，x是C_i内的一个数据点，那么类中心定义如下：

$$\overline{x_i} = \frac{1}{n_i} \sum_{x \in C_i} x$$

其中n_i是第i个聚类中的点数。因此，两个类C_a和C_b的类间距离为：

$$D_C(C_a, C_b) = d(r_a, r_b)$$

其中γ_a和γ_b是类C_a和C_b的中心点，d是某种形式的距离公式。

划分方法

- **划分方法:** 给定一个有*n*个对象的数据集，划分聚类技术将构造数据*k*个划分，每一个划分就代表一个簇，$k \leq n$。也就是说，它将数据划分为*k*个簇，而且这*k*个划分满足下列条件：
  - 每一个簇至少包含一个对象。
  - 每一个对象属于且仅属于一个簇。
- 对于给定的*k*，算法首先给出一个初始的划分方法，以后通过反复迭代的方法改变划分，使得每一次改进之后的划分方案都较前一次更好。
- **给定一个k，要构造出k个簇，并满足采用的划分准则：**
  - ***k-平均*:由簇的中心来代表簇；**
  - ***k-中心点*: 每个簇由簇中的某个数据对象来代表。**

聚类设计的评价函数

- 一种直接方法就是观察聚类的类内差异（Within cluster variation）和类间差异(Between cluster variation）。
  - 类内差异：衡量聚类的紧凑性，类内差异可以用特定的距离函数来定义，例如，

$$w(C) = \sum_{i=1}^{k} w(C_i) = \sum_{i=1}^{k} \sum_{x \in C_i} d(x, \overline{x_i})^2$$

  - 类间差异：衡量不同聚类之间的距离，类间差异定义为聚类中心间的距离，例如，

$$b(C) = \sum_{1 \le j < i \le k} d(\overline{x_j}, \overline{x_i})^2$$

- 聚类的总体质量可被定义为w(c)和b(c)的一个单调组合，比如w(c) / b(c) 。

k-means算法

- *k*-means算法，也被称为*k*-平均或*k*-均值，是一种得到最广泛使用的聚类算法。相似度的计算根据一个簇中对象的平均值来进行。

输入：簇的数目*k*和包含*n*个对象的数据库。

输出：*k*个簇，使平方误差准则最小。

(1)assign initial value for means; /*任意选择*k*个对象作为初始的簇中心；*/

(2) REPEAT

(3) FOR *j*=1 to *n* DO assign each x_j to the closest clusters;

(4) FOR *i*=1 to *k* DO /*更新簇平均值*/

$$\overline{x_i} = |C_i| \sum_{x \in C_i} x$$

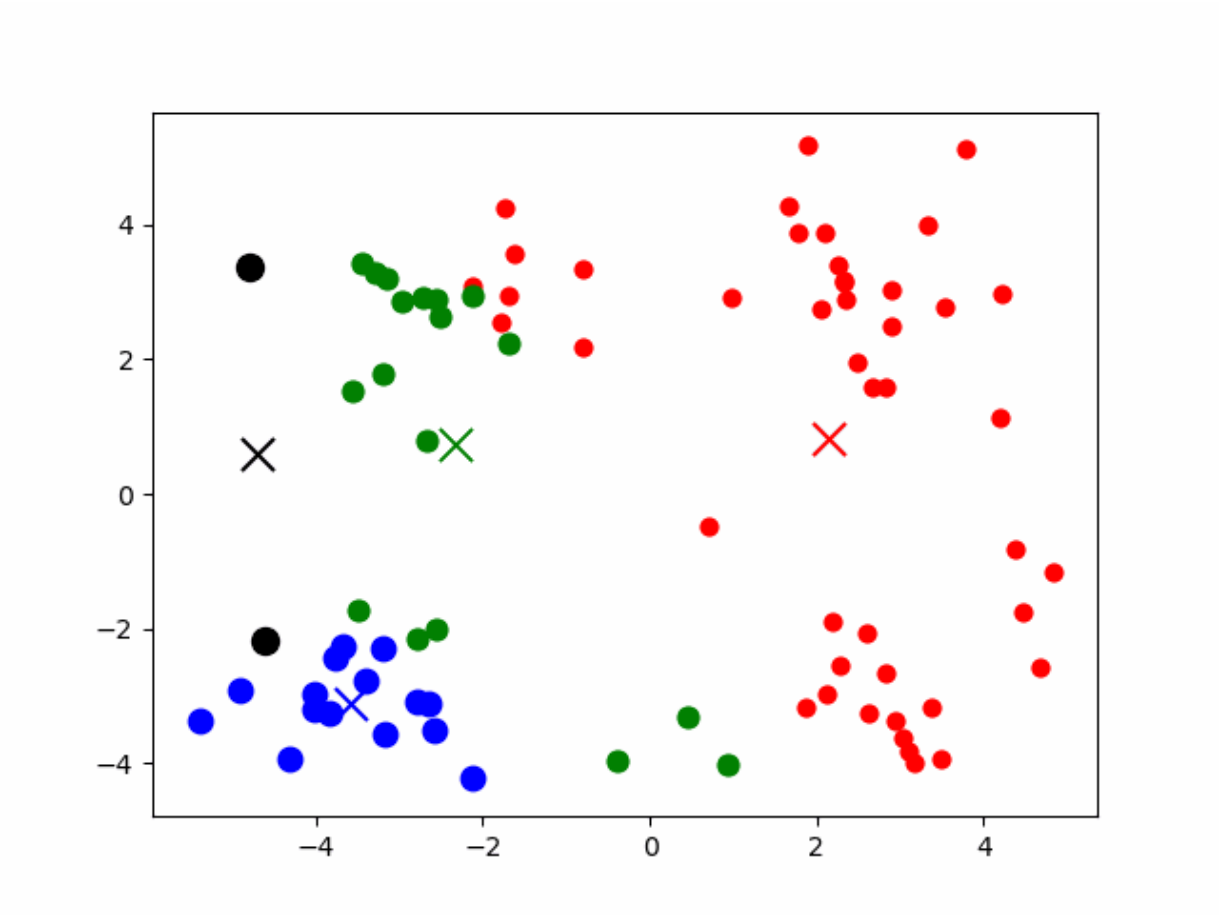
(5) Compute $E = \sum_{i=1}^{k} \sum_{x \in C_i} \left| x - \overline{x_i} \right|^2$ /*计算准则函数*E*/

(6) UNTIL *E*不再明显地发生变化。

k-means算法

- 算法首先随机地选择*k*个对象，每个对象初始地代表了一个簇的平均值或中心。对剩余的每个对象根据其与各个簇中心的距离，将它赋给最近的簇。然后重新计算每个簇的平均值。这个过程不断重复，直到准则函数收敛。
- 准则函数试图使生成的结果簇尽可能地紧凑和独立。

示例1



示例2

样本数据 序号	属性 1	属性 2
1	1	1
2	2	1
3	1	2
4	2	2
5	4	3
6	5	3
7	4	4
8	5	4

根据所给的数据通过对其实施*k*-means (设*n*=8，*k*=2),，其主要执行执行步骤：

第一次迭代：假定随机选择的两个对象，如序号1和序号3当作初始点，分别找到离两点最近的对象，并产生两个簇{1，2}和{3，4，5，6，7，8}。

对于产生的簇分别计算平均值，得到平均值点。

对于{1，2}，平均值点为（1.5，1）（这里的平均值是简单的相加出2）；

对于{3，4，5，6，7，8}，平均值点为（3.5，3）。

第二次迭代：通过平均值调整对象的所在的簇，重新聚类，即将所有点按离平均值点（1.5，1）、（3.5，1）最近的原则重新分配。得到两个新的簇：{1，2，3，4}和{5，6，7，8}。重新计算簇平均值点，得到新的平均值点为（1.5，1.5）和（4.5，3.5）。

第三次迭代：将所有点按离平均值点（1.5，1.5）和（4.5，3.5）最近的原则重新分配，调整对象，簇仍然为{1，2，3，4}和{5，6，7，8}，发现没有出现重新分配，而且准则函数收敛，程序结束。

迭代次数	平均值（簇1）	平均值（簇2）	产生的新簇	新平均值（簇1）	新平均值（簇2）
1	（1，1）	（1，2）	{1，2}，{3，4，5，6，7，8}	（1.5，1）	（3.5，3）
2	（1.5，1）	（3.5，3）	{1，2，3，4}，{5，6，7，8}	（1.5，1.5）	（4.5，3.5）
3	（1.5，1.5）	（4.5，3.5）	{1，2，3，4}，{5，6，7，8}	（1.5，1.5）	（4.5，3.5）

k-means算法的性能分析

- 主要优点：
  - 是解决聚类问题的一种经典算法，简单、快速。
  - 对处理大数据集，该算法是相对可伸缩和高效率的。
  - 当结果簇是密集的，它的效果较好。
- 主要缺点
  - 在簇的平均值被定义的情况下才能使用，可能不适用于某些应用。
  - 必须事先给出*k*（要生成的簇的数目），而且对初值敏感，对于不同的初始值，可能会导致不同结果。
  - 不适合于发现非凸面形状的簇或者大小差别很大的簇。而且，它对于“躁声”和孤立点数据是敏感的。

k-中心点（K-medoids）

- K-medoids聚类算法的基本策略是：首先通过任意为每个聚类找到一个代表对象，确定n个数据对象的k个聚类。其他对象则根据它们与这些聚类代表的距离分别将它们归属到各相应聚类中。而如果替换一个聚类代表能够改善所获聚类质量的话，那么就可以用一个新对象替换老聚类对象。

k-中心点（K-medoids）

- K-medoids聚类算法具体步骤：
- 输入：聚类个数k，以及包含n个数据对象的数据库。
- 输出：满足基于各聚类中心对象的方差最小标准的 个聚类。
- 1）从 个数据对象任意选择 个对象作为初始聚类代表。
- 2）循环3）到5）直到每个聚类不再发生变化为止；
- 3）依据每个聚类的中心代表对象，以及各对象与这些中心对象间距离，并根据最小距离重新对相应对象进行划分。
- 4）任意选择一个非中心对象o_{random}，计算其与中心对象 交换的整个成本S
- 5）若为 负值则交换,以构成新聚类的 个中心对象。

计算复杂度相当大

层次方法

- **凝聚的层次聚集：**自底向上的策略首先将每个对象作为一个簇，然后合并这些原子簇为越来越大的簇，直到所有的对象都在某个簇中，或者终结条件满足。
- **分裂的层次聚类：**自顶向下，首先将所有对象置于一个簇中，然后逐渐细分为越来越小的簇，直到每个对象自成一个簇或者达到了某个终结条件。（达到希望的簇数或两个簇之间的距离超过了某个阀值）

密度方法

- **DBSCAN(Density-Based Spatial Clustering of Applications with Noise)，基于密度的聚类**
- 把目标的密度作为考虑聚类的因素。
- 基于密度聚类算法的代表。它将具有足够高密度的区域划分成簇，并可以在带有“噪声”的空间数据库中发现任意形状的聚类。它定义簇为密度相连的点的最大集合。

密度方法

- 给定对象半径e内的区域称为该对象的e邻域。如果一个对象的e邻域至少包含最小数目$MinPts$个对象，则称该对象为核心对象。
- DBSCAN通过检查数据库中每个点的e邻域来寻找聚类。如果一个点的e邻域包含多于$MinPts$个点

↓

创建一个以P作为核心对象的新簇

↓

DBSCAN反复地寻找从这些核心对象直接密度可达的对象并加入该簇，直到没有新的点可以被添加

隐马可夫模型

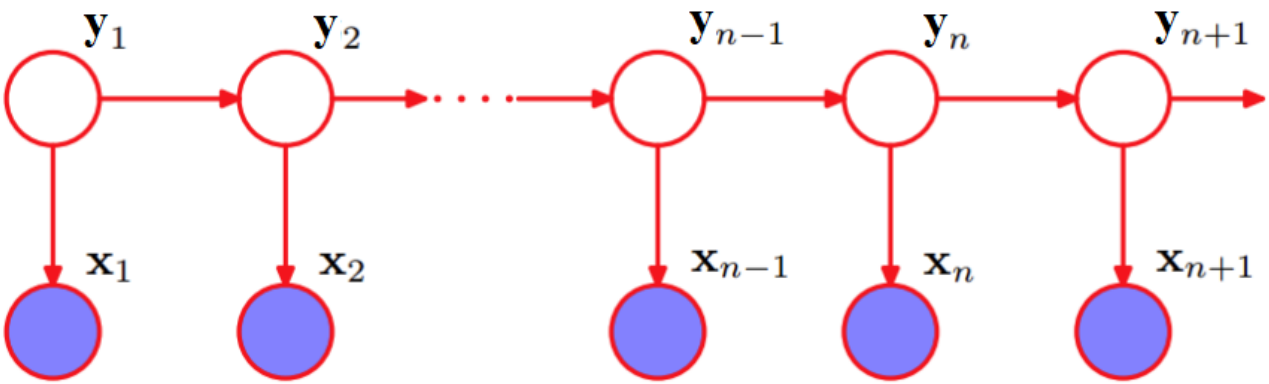
推断（Inference）

- 在概率模型中，利用已知变量推测未知变量的分布称为“推断”。
- 假定关心的变量集合为Y，可观测变量的集合为O，其他变量的集合为R，生成模型考虑联合分布P(Y,R,O)，判别模型考虑条件分布P(Y,R|O).给定一组观测变量的值，推断就是要由P(Y,R,O)或者P(Y,R|O)，得到条件概率P(Y|O)。
- 概率图模型是一类用图来表达变量相关关系的概率模型。用一个结点表示一个或一组随机变量，结点之间的边表示变量间的概率相关关系。
- 有向无环图表示变量间的依赖关系——贝叶斯网络
- 无向图表示变量间的相关关系——马可夫网（Markov network）

隐马可夫模型（Hidden Markov Model，HMM）

- 结构最简单的动态贝叶斯网，一种著名的有向图模型，主要用于时序数据建模，在语音识别、自然语言处理等领域有广泛应用。
- 比如语音识别，给你一段音频数据，需要识别出该音频数据对应的文字。这里音频数据就是观测变量，文字就是隐藏变量。我们知道，对单个文字而言，虽然在不同语境下有轻微变音，但大致发音是有统计规律的。另一方面，当我们说出一句话时，文字与文字之间也是有一些转移规律的。比如，当我们说出“比”这个字时，下一个大概率的字一般是“如”“较”等。虽然文字千千万，但文字与文字之间的转移却是有章可循的。有了文字的发音特征，以及文字与文字之间的转移规律，那么从一段音频中推测出对应的文字也就可以一试了。

模型表示



- 状态变量$\{y_1, y_2, \ldots, y_n\}$， y_i表示第i个时刻系统状态，通常该状态是隐藏的，不可被观察的，即隐变量。
- 观测变量$\{x_1, x_2, \ldots, x_n\}$， x_i表示第i个时刻的观测值。
- 状态变量y_i的取值范围称为状态空间，通常具有N个离散的状态取值$\{s_1, s_2, \ldots, s_N\}$。（HMM中隐变量必须是离散的）
- 观测变量可以离散也可以连续。为了方便，假设观测变量x_i的取值范围是$\{o_1, o_2, \ldots, o_M\}$

模型假设

- 在任意一时刻t，观测变量的取值仅仅依赖于状态变量，即x_t由y_t决定。
- t时刻的状态y_t仅依赖于前一个状态y_{t-1}。
- 所有变量的联合概率分布为

$$P(x_1, y_1, ..., x_n, y_n) = p(y_1)p(x_1 \mid y_1)\prod_{i=2}^{n} p(y_1 \mid y_{i-1})P(x_i \mid y_i)$$

- 欲确定一个隐马可夫模型还需要以下三组参数
  - 状态转移概率：各状态间转换的概率 $A = [a_{ij}]_{N \times N}$

    $a_{ij} = P(y_{t+1} = s_j \mid y_t = s_i), \quad 1 \le i, j \le N$
  - 输出观测概率：根据当前状态获得各观测值的概率 $B = [b_{ij}]_{N \times M}$

    $b_{ij} = P(x_t = o_j \mid y_t = s_i), \quad 1 \le i \le N, 1 \le j \le M$
  - 初始状态概率：模型在初始时刻各状态出现的概率 $\boldsymbol{\pi} = (\pi_1, \pi_2, ..., \pi_N)$

    $\pi_i = P(y_1 = s_i), \quad 1 \le i \le N$

模型过程

- 给定隐马可夫模型 $\lambda=[A,B,\pi]$ 可以按照如下过程产生观测序列
  - （1）设置t=1，并根据初始状态概率π选择初始状态y_1
  - （2）根据状态y_t和输出观测概率B选择观测变量取值x_t
  - （3）根据状态y_t和状态转移矩阵A转移模型状态，即确定yt+1
  - （4）若t<n，设置t=t+1，并转移到第（2）步，否则停止。

三种应用

- 给定模型 $\lambda=[A,B,\pi]$，如何有效计算其产生的观测序列

$\mathbf{x}=\{x_1, x_2, \ldots, x_n\}$的概率$P(\mathbf{x}|\lambda)$。即如何评估模型与观测序列的匹配程度。

- 给定模型 $\lambda=[A,B,\pi]$ 和观测序列$\mathbf{x}=\{x_1, x_2, \ldots, x_n\}$，如何找到与此观测序列最匹配的状态序列$\mathbf{y}=\{y_1, y_2, \ldots, y_n\}$。即如何根据观测序列推断出隐藏的模型状态？【Viterbi算法】
- 给定观测序列$\mathbf{x}=\{x_1, x_2, \ldots, x_n\}$，如何调整模型参数$\lambda=[A,B,\pi]$是的该序列出现的概率$P(\mathbf{x}|\lambda)$ 最大？即如何训练模型使其能最好地描述观测数据。【最大似然估计+EM算法】