# 数据挖掘概念与技术笔记整理

liz*

April 1, 2009

## Contents

1 基本概念整理 2
1.1 基本定义 . . . 2
1.2 MASS . . . 2

2 关联规则 3
2.1 基本定义 . . . 3
2.2 关联规则的分类 . . . 3
2.3 经典Apriori算法 . . . 3
2.4 不产生候选, 挖掘频繁项集 . . . 4
2.5 挖掘闭频繁项集 . . . 5
2.6 挖掘各种类型的关联规则 . . . 5
2.7 由关联挖掘到相关分析 . . . 5

3 分类和预测 7
3.1 基本定义 . . . 7
3.2 决策树归纳学习 . . . 7
3.3 属性选择度量 . . . 8
3.4 贝叶斯分类 . . . 9
3.5 基于规则的分类 . . . 10
3.6 用后向传播分类 . . . 11
3.7 支持向量机Support Vector Machine . . . 11
3.8 关联分类 . . . 11
3.9 惰性学习法, 或从近邻学习 . . . 11
3.10 其他分类方法 . . . 11
3.11 预测 . . . 12

4 聚类 13
4.1 基本定义 . . . 13

*Email: shengyan1985@gmail.com

4.2 计算相异度 . . . 13
4.3 几种距离计算方法 . . . 13
4.4 主要聚类方法 . . . 14
4.5 聚类高维数据 . . . 15
4.6 基于频繁模式的聚类 . . . 16
4.7 基于约束的聚类分析 . . . 16
4.8 离群点分析 . . . 16

5 多关系/序列/数据流挖掘 17
5.1 多关系数据挖掘 . . . 17
5.2 数据流挖掘 . . . 17
5.3 序列挖掘 . . . 17

6 Markov模型 18

7 潜在语义索引 18

8 杂七杂八的东西 18
8.1 文本挖掘 . . . 18
8.2 PageRank . . . 18
8.3 Vector Space Model . . . 18
8.4 文本索引技术 . . . 19
8.5 文本检索技术 . . . 19
8.6 分布式Web信息检索技术研究 . . . 19

# 1 基本概念整理

## 1.1 基本定义

1. 数据仓库 Data Warehouse: 一种数据存储仓库的系统结构, 一种多个异构数据源在单个站点以统一的模式组织的储存库, 以支持管理决策.

   - 这是一个比较抽象的概念, 刚开始读n遍估计也不知道到底什么是数据仓库. 我的理解是, 集成了多种数据源, 可以是文本, 结构化的数据库, 任何东西, 按照一定策略组织起来, 其中包含对这些数据的描述, 包含时间信息等. 之后, 可以在这巨大的数据上进行挖掘.

2. 数据挖掘 Data Mining: 从大量数据中提取或"挖掘"知识.

   - 另外一个说法就是, 知识发现（Knowledge Discovery in Database）, 可以看成是广义的数据挖掘, 因为KDD还包括数据预处理, 数据变换, 评估, 知识表示等, 数据挖掘只是其中的一个核心部分. 而数据挖掘对应的数据源, 不仅可以是精心组织的数据仓库中的数据, 也可以是动态的数据, 如数据流, 随时间改变而改变的数据等等, 可以说是任何数据.

3. 数据挖掘主要包括哪些功能:

   - 概念/类的描述: 特征化和区分.
     - 这里主要是如何对数据描述以更好的表达数据包含的信息, 如对一个文本, 如何提取特征词, 如何组织其间结构以便既能精简文本原数据量又能准确的表达出文本内容.
   - 挖掘频繁模式, 关联规则和相关规则. 包括频繁项集, 频繁序列模式, 频繁结构模式.
     - 像关联规则, 现有许多网站都有这方面的相关应用, 比较成熟. 其中, 有两个度量在关联规则中非常重要, 分别是: 支持度和置信度. 下述.
   - 分类与预测Classfication&Predication
     - 分类和预测都是一种有监督的学习, 就是说利用已有的知识, 如已经知道每个对象属于的类标号, 进行学习得到一个分类模型, 之后再利用这个分类器对未知的对象进行分类, 即给定这个未知对象的类标号. 这里的类标号本质上也是一个属性, 称之为决策属性, 而对应的, 其他属性称为分类属性, 也就是说针对已知对象（训练集）上, 寻找多个分类属性和决策属性之间的关系.
   - 聚类Cluster
     - 聚类是一种无监督的学习, 属性不区分分类属性还是决策属性, 因为事先我们不知道每个对象是属于哪个类, 需要自动发掘他们之间的类别, 类别与类别之间的边界等. 所谓物以类聚, 人与群分, 现实生活中, 仔细想想, 也无处不在聚类啦...
   - 还有其他的, 如离群点分析Outlier Mining和演变分析Evolution Analysis.

   - 可以说, 数据挖掘涉及的范围是超级广的, 层面也是非常多的. 所以非常经常的看到好多大学实验室研究的内容都是属于数据挖掘这个大大范围的.

## 1.2 MASS

待补充

# 2 关联规则

## 2.1 基本定义

1. 支持度Support: 满足规则$X \Rightarrow Y$的事物在数据库中所占的百分比. 主要针对事务数据库. 记:

$$Support(X \Rightarrow Y) = P(X \bigcup Y)$$

2. 置信度Confidence: 评估发现的规则的确定性程度. 记:

$$Support(X \Rightarrow Y) = P(X|Y)$$

   - 规则的支持度和置信度是规则兴趣度的两种度量, 分别反映所发现的规则的有用性和确定性. 一般设置最小支持度阈值minsup, 最小置信度阈值minconf. 同时满足minsup和minconf的规则成为强规则.

3. 频繁模式Frequent Pattern: 频繁的出现在数据集中的模式, 包括项集, 子序列, 子结构.
   其中的频繁项集是指频繁的同时出现在交易数据集中的东西的集合. 频繁序列是频繁的同时出现在数据集中的序列模式. 还有子结构, 子图, 子树, 子格等等这些是结构模式.

4. 项集: 项的集合, k项集表示有k个元素.
   项集频率: 支持度计数, 包括该项集的事务数.
   频繁k项集记为 $L_k$ .

$$Confidence(A \Rightarrow B) = P(B|A) = \frac{Support(A \bigcup B)}{Support(A)}$$

   所以, 挖掘关联规则的问题可归结为挖掘频繁项集. 一个项集是频繁的, 则它的每个子集也是频繁的.

5. 闭频繁项集: 不存在真超项集Y, 使Y与X在S中有相同的支持度计数, 则称X在S中是封闭的.
   极大频繁项集: 不存在超项集Y, 使 $X \subseteq Y$ 并且Y在S中是频繁的.

## 2.2 关联规则的分类

规则分类根据不同的层次也有很多中, 如根据抽象层次可分为多层关联规则和单层关联规则, 根据数据维可分为单维和多维关联规则, 等等了.

## 2.3 经典Apriori算法

最基本的思想: 频繁项集的所有非空子集也必定是频繁的. 此为分单调性(anti-monotone).

---

算法: Apriori. 使用逐层迭代方法基于候选产生找出频繁项集.
输入: D - 事务数据库; min_sup - 最小支持度计数阈值.
输出: L - D中的频繁项集.

---

方法:

1) $L_1$ = find_frequent_1 - itemsets(D);
2) for (k = 2; $L_{k-1} \neq \emptyset$; k++){
3) $C_k$ = apriori_gen($L_{k-1}$);
4) for each 事务$t \in D$ { // 扫描D用于计数
5) $C_t$ = subset($C_k$, t); //得到t的子集, 它们是候选
6) for each 候选$c \in C_t$
7) c.count++;
8) }
9) $L_k = \{c \in C_k | c.count \geq min\_sup\}$
10) }
11) return $L = \cup_k L_k$;

procedure apriori_gen($L_{k-1}$: frequent(k-1)-itemsets)
1) for each 项集$l_1 \in L_{k-1}$
2) for each项集$l_2 \in L_{k-1}$
3) if $(l_1[1] = l_2[1]) \wedge (l_1[2] = l_2[2]) \wedge ... \wedge (l_1[k-2] = l_2[k-2]) \wedge (l_1[k-1] < l_2[k-1])$ then {
4) $c = l_1 \bowtie l_2$; // 连接步: 产生候选
5) if has_infrequent_subset(c, $L_{k-1}$) then
6) delete c; //剪枝步: 删除非频繁的候选
7) else add c to $C_k$;
8) }
9) return $C_k$;

precedure has_infrequent_subset (c:candidate k-itemset; $L_{k-1}$:frequent (k-1)-itemsets) // 使用先验知识
1) for each (k-1)-subset s of c
2) if $s \notin L_{k-1}$ then
3) return TRUE;
4) return FALSE;

- 这个算法依次从频繁1项集至k项集进行选取, 利用的是频繁项集的子项集肯定也是频繁的, 反之, 不频繁的项集其父项集也必定不频繁, 这体现在剪枝步骤.

---

由频繁项集产生关联规则,

$$confidence(A \Rightarrow B) = P(B|A) = \frac{Support\_count(A \bigcup B)}{Support\_count(A)}$$

$$\begin{cases} \text{对于每个频繁项集l, 产生l的所有非空子集} \\ \text{对于l的每个非空子集s, 如果 } \frac{Support\_count(l)}{Support\_count(s)} \geq min\_conf \text{, 则输出 } s \Rightarrow (l-s) \end{cases}$$

提高Apriori算法效率

- 基于散列的技术(散列项集到对应的桶中)
- 事务压缩(压缩未来迭代扫描的事务数), 不包含任何频繁k项集的事务不可能包含任何频繁(k+1)项集
- 划分(为寻找候选项集划分数据): 2次扫描db
- 抽样(对给定数据的子集挖掘)
- 动态项集计数(在扫描的不同点添加候选项集)

## 2.4 不产生候选, 挖掘频繁项集

- 之前的Apriori算法是基于候选项集的, 对候选项集进行删减, 在实际数据库中, 这个候选集是很大的, 时空消耗都非常大. 所以提出了以下几个不是基于候选策略的频繁项集挖掘算法

1. 频繁模式增长(frequent-pattern growth), 简称FP增长
   基于分治策略, 首先, 将提供频繁想的数据库压缩到一棵频繁模式树(FP树), 但仍然保留项集关联信息; 然后将压缩后的数据库划分成一组条件数据库(一种特殊类型的投影数据库), 每个关联一个频繁或"模式段", 并分别挖掘每个条件数据库.
   大概过程如下:
   (a) 第一次扫描到处频繁1项集和支持度计数(频率)L

(b) 构造FP树, 类似于page 157

(c) 方便树的遍历, 创建项头表, 使每项通过一个节点链指向它在树中的位置, 那么, 数据库频繁模式的挖掘问题转化为挖掘FP树问题.

(d) FP树挖掘过程, 长度为1的频繁模式(初始后缀模式, 构造它的条件模式基(一个"子数据库", 由FP树中与后缀模式一起出现的前缀路径集组成, 然后, 构造他的(条件)FP树, 并递归地对该树进行挖掘.

(e) 模式增长通过后缀模式与条件FP树产生的频繁模式连接实现.$I_5$ 的条件模式基 $< I_2, I_1 : 1 >$ , $< I_2, I_1, I_3 : 1 >$ 所以, 条件FP树之包含 $< I_2 : 2, I_1 : 2 > \longrightarrow$该单个路径产生频繁模式的所有组合有: $\{I_2, I_5 : 2\}, \{I_1, I_5 : 2\}, \{I_2, I_1, I_5 : 2\}$

2. 基于内存的FP树非常不现实, 因为构造的FP树会非常大, 全部加载入内存显然不切实际. 所以一种可选方案为: 首先将数据库划分成投影数据库的集合, 然后在每个投影数据库上构造FP树并挖掘, 可递归的用于投影数据库.

3. 使用垂直数据格式挖掘频繁项集. 像Apriori和FP增长是一种水平数据格式,{TID: itemset},,即可以说是一种横向的. 对应的, 垂直数据格式, 基于项-TID集格式, {item: TID_set}表示.
这种如Zaki开发的ECLAT算法. 其主要思想: 通过每对频繁单个项的TID集的交, 可对该数据集进行挖掘, 依次进行2项 $\cdots$ n项挖掘.
缺点: TID集可能很长, 大量空间, 求长集合的交也需大量计算时间. 一种解决方法: 差集(diffset)技术.

## 2.5 挖掘闭频繁项集

直接在挖掘过程中搜索闭频繁项集, 这需要在挖掘过程中, 一旦识别闭项集就尽快对搜索空间进行剪枝, 主要步骤: 1) 项合并; 2) 子项集剪枝; 3) 项跳过.
另一种优化为: 有效的检查新导出的频繁项集, 看出是否是闭的, 因为挖掘过程本身不确保所产生的每个频繁项集都是闭的. 要做相关检查, 一般为1) 超集检查, 2) 子集检查...

## 2.6 挖掘各种类型的关联规则

- 这里有很多中, 如数据仓库中的多层关联规则(不同抽象层次上), 多维关联规则(多个维), 量化关联规则(值之间隐含排序的数值属性), 涉及到很多方面, 不再详细列出.

## 2.7 由关联挖掘到相关分析

- 低支持度阈值或挖掘长模式时, 好多规则都不是用户感兴趣的.

- 如何识别强关联规则的有趣与否?
$A \Rightarrow B$置信度有一定的欺骗性. 只是给定A, B的条件概率的估计, 并不度量A和B之间相关和蕴含的实际强度(或缺乏强度), 寻求支持度-置信度框架的替代对框架有趣的数据联系可能是有用的. 使用相关度量来扩充关联规则的支持度-置信度框架.

$$A \Rightarrow B[support, confidence, correlation]$$

有很多相关度量方法

1. 提升度lift:
A的出现独立于B的出现, $P(A \bigcup B) = P(A)P(B)$, 否则, A,B依赖(dependent)和相关(correlated)的.

$$lift(A, B) = \frac{P(A \bigcup B)}{P(A)P(B)} < 1: \text{A和B负相关,}$$

$$lift(A, B) = \frac{P(A \bigcup B)}{P(A)P(B)} > 1: \text{A和B正相关, 一个的出现蕴含另一个出现}$$

$$lift(A, B) = \frac{P(A \bigcup B)}{P(A)P(B)} = 1: \text{A和B是独立的, 无相关性}$$

$\frac{P(A|B)}{P(B)} = \frac{conf(A \Rightarrow B)}{sup(B)}$称为关联(或相关)规则$A \to B$提升度

评估一个的出现"提升"另一个的程度.

2. $\chi^2$ 度量:

$$\chi^2 = \sum_{i=1}^{c}\sum_{j=1}^{r}\frac{(O_{ij} - e_{ij})^2}{e_{ij}}$$

3. 全置信度all_confidence:

$$all\_conf(X) = \frac{sup(X)}{max\_item\_sup(X)} = \frac{sup(X)}{max\{sup(i_j)|\forall i_j \in X\}}$$

4. 余弦度量:

$$cosine(A,B) = \frac{P(A\bigcup B)}{\sqrt{P(A)\times P(B)}} = \frac{sup(A\bigcup B)}{\sqrt{sup(A)\times sup(B)}}$$

5. 兴趣度量:
零不变性, 一种好的启发式方式是让用户说明他们的直觉或期望作为限制搜索空间的约束条件, ⟵基于约束的挖掘(constraint-based mining), 包括知识类型的约束, 数据约束, 维/层约束, 兴趣度约束, 规则约束.

# 3 分类和预测

## 3.1 基本定义

1. 分类Classification: 是指预测分类(离散, 无序的)标号. 有很多种方法, 决策树分类器, 贝叶斯分类器, 贝叶斯信念网络, 基于规则的分类器, 支持向量机, 后向传播, 基于关联规则挖掘的分类, k最近邻分类器, 基于案例的推理. 涉及到相关理论: 遗传算法, 粗糙集, 模糊逻辑技术.
   - 现已提出了非常多分类的方法. 改进也有很多很多, 但实际应用中, 贝叶斯分类器应用较多, 因为基于朴素贝叶斯公式思想比较简单, 实现起来也相对容易, 也能达到较好的时空复杂度. 但朴素贝叶斯理论是基于属性是相互独立这条假设上的, 但很多情况下, 属性间是不相互独立的, 而是有相关关系的. 所以...
2. 数值预测Numeric Prediction: 建立连续值函数模型. 一般方法有: 线性回归, 非线性回归, 一般都是基于数学上的回归分析.
   - 分类和预测是有区别的, 预测所构造的模型(称为预测器Predictor)预测的是一个函数或连续值. 分类一般是一些离散值.
3. 分类一般步骤:
   (a) 学习步: 建立描述预先定义的数据类或概念集的分类器, 此为分类器的训练阶段.
   (b) 分类: 此为分类器的测试阶段. 主要用于检测数据用于评估分类规则的准确率, 一般衡量标准, 如precision和recall, 还有各种各样的指标用于衡量分类器的性能.
   (c) 衡量: 衡量分类器或预测器的性能, 考虑速度, rebust, 可伸缩性, 可解释性等.

## 3.2 决策树归纳学习

- 使用属性选择度量, 即要选择最好的划分成不同类的属性. 这里考虑噪声离群点, 树的剪枝(先剪枝, 后剪枝). 一些经典分类器,如ID3, C4.5, CART都采用贪心(非回溯的)方法, 决策树分类方法由顶向下递归的分治方法, 从训练元组集和它们的相关联的类标号开始构造决策树. 随着树的构建, 训练集递归的划分成较小的子集.

---

算法: Generate_decision_tree. 由数据划分D的训练元组产生决策树.
输入: D - 数据划分D是训练元组和对应类标号的集合; attribute_list - 候选属性的集合; Attribute_selection_method - 一个确定"最好"地划分元组为个体类的分裂准则的过程, 这个准则由分裂属性和分裂点或分裂子集组成.
输出: 一颗决策树.

---

方法:
1) 创建一个节点N
2) if D中的元组都是同一类C then
3) 返回N作为叶节点, 以类C标记
4) if attribute_list为空 then
5) 返回N作为叶节点, 标记为D中的多数类 // 多数表决
6) 使用attribute_selection_method(D, attribute_list), 找出"最好"的splitting_criterion;
7) 用splitting_criterion标记节点N;
8) if splitting_criterion 是离散值的并且允许多路划分 then // 不限于二叉树
9) attribute_list ← attribute_list - splitting_attribute; // 删除划分属性
10) for splitting_criterion的每个输出j // 划分元组并对每个划分产生子树, 包含递归过程
11) 设$D_j$是D中满足输出j的数据元组的集合; // 一个划分
12) if $D_j$为空 then
13) 加一个树叶到节点N, 标记为D中的多数类;
14) else 加一个由Generate_decision_tree($D$, attribute_list)返回的节点到节点N;
15) end for
16) 返回N;

- D为数据划分, 初始, 它是训练元组和相应类标号的完全集. 参数attribute_list是描述元组的属性列表. Attribute_selection_method指定选择属性的启发式过程, 所选择的属性按类"最好"地区分元组, 划分各个部分.

该过程使用一种属性选择度量(下述).算法调用使用attribute_selection_method确定分裂准则, 分裂准则指定分裂属性, 并且指出分裂点或分裂子集, 理想中分裂准则确定每个分支上的输出划分尽可能地纯, 就是尽可能使所有元组属于同一类.

---

## 3.3 属性选择度量

1. 信息增益: 基于香农公式, 利用信息熵. 选择具有最高信息增益的属性作为节点N的分裂属性. 该属性使结果划分中的元组分类所需的信息量最小, 并反映这些划分中的最小随机性或"不纯性". 这种方法使对给定元组分类所需的期望测试数目最小, 并确保找到一棵简单的(但不必是最简单的)树.

$$Info(D) = -\sum_{i=1}^{m} p_i \log_2(p_i)$$

$p_i$是D中任意元组属于类$C_i$的概率, 用$|C_{i,D}|/|D|$估计, 使用以2为底的对数函数, 因为信息是用二进制编码. Info(D)是识别D中元组的类标号所需要的平均信息量, 每个类的元组所占的百分比, Info(D)称为D的熵(entropy).
按照属性A划分D中的元组成v个子集$\{D_1, D_2, \cdots, D_v\}$, Dj在A上具有值$a_j$,

$$Info_A(D) = \sum_{j=1}^{v} \frac{|D_j|}{|D|} \times Info(D_j)$$

项$\frac{|D_j|}{|D|}$充当第j个划分的权重. $Info_A(D)$是基于A划分对D的元组所需要的期望信息. 期望信息越小, 划分的纯度越高.
信息增益定义为原来的信息需求(即仅基于类比例)与新的需求(即对A划分之后得到的)之间的差. 即是,

$$Gain(A) = Info(D) - Info_A(D)$$

$Gain(A)$: 通过A的划分我们得到了多少. 它是知道A的值而导致的信息需求的期望减少, 选择具有最高信息增益$Gain(A)$的属性A作为节点N的分裂属性. 这等价于按能做"最佳分类"的属性A划分, 使得完成元组分类还需要的信息最小(即最小化$Info_A(D)$).
信息增益度量偏向具有许多输出的测试, 即, 倾向于选择具有大量值的属性. 比如说某个keyid, 主关键字, 唯一性, 这种类型的属性信息增益肯定是最大的, 但这种划分对分类没用.
note: 信息增益使用在ID3中.

2. 增益率gain ratio: 它使用分裂信息(split information)值将信息增益规范化.

$$SplitInfo_A(D) = -\sum_{j=1}^{v} \frac{|D_j|}{|D|} \times \log_2\left(\frac{|D_j|}{|D|}\right)$$

该值代表通过将训练数据集D划分对应于属性A测试的v个输出的v个划分产生的信息. 注意, 对于每个输出, 关于D中元组总数考虑具有该输出的元组数. 它不同于信息增益, 信息增益关于分类度量基于相同划分所需要的信息. 增益率定义为

$$GainRatio(A) = \frac{Gain(A)}{SplitInfo(A)}$$

选择具有最大增益率的属性作为分裂属性. 然后, 随着分裂信息趋向于0, 该比例变得不稳定. 为了避免这种情况, 增加一个约束, 选取测试的信息增益必须较大, 至少与所考察的所有测试的平均增益一样大.
note: 信息增益率用于C4.5中.

3. Gini指标: CART中使用Gini指标. 它度量数据划分成训练元组集D的不纯度, 定义为

$$Gini(D) = 1 - \sum_{i=1}^{m} p_i^2$$

$p_i$是D中元组属于$C_i$类的概率, 并用$|C_{i,D}|/|D|$估计. 对于m个类计算和.Gini指标考虑每个属性的二元划分, 对于属性A, 有v个值, 存在$2^v-2$中形成数据集D的两个划分的可能方法. 当考虑二元分裂时, 计算每个结果划分的不纯度的加权和.
如果A的二元分裂将D划分成$D_1$和$D_2$, 则给定该划分D的Gini指标为

$$Gini_A(D)=\frac{|D_1|}{|D|}Gini(D_1)+\frac{|D_2|}{|D|}Gini(D_2)$$

对于每个属性, 考虑每种可能的二元划分. 对于离散值属性, 选择该属性产生最小Gini指标的子集作为它的分裂子集.
对离散或连续值属性A的二元分裂将导致的不纯度降低位

$$\Delta Gini(A)=Gini(D)-Gini_A(D)$$

最大化不纯度降低(或等价地, 具有最小Gini指标)的属性作为分裂属性. 该属性和它的分裂子集(对于离散值的分裂属性) 或分裂点(对于连续值的分裂属性) 一起形成分裂准则.

4. 其他
   (a) 最小描述长度(Minimum Description Length, MDL): 具有最小偏向多值属性的偏倚.
   (b) 多元划分: 基于属性的组合而不是单个属性.

## 3.4 贝叶斯分类

- 基于贝叶斯定理, 类条件独立性的假设, 所以为"朴素的".

1. 贝叶斯定理: X用n个属性集的测量描述. 令H为某种假设, 如数据元组X属于某特定类C. 对于分类问题, 希望确定$P(H|X)$–给定"证据"或观测数据元组X, 假设H成立的概率. 换言之, 给定X的属性描述, 找出元组X属于类C的概率.
   $P(H|X)$是后验概率(posterior probability), 或在条件X下, H的后验概率. $P(H)$是先验概率(prior probability), 或H的先验概率. 后验概率$P(H|X)$比先验概率$P(H)$基于更多的信息. $P(H)$独立于X. 类似地, $P(X|H)$是条件H下, X的后验概率. $P(X)$是X的先验概率.
   贝叶斯定理提供了一种由$P(X),P(H),P(X|H)$计算$P(H|X)$的方法:

$$P(H|X)=\frac{P(X|H)P(H)}{P(X)}$$

   原型公式是基于条件概率公式的,

$$P(AB)=P(A)P(B|A)=P(B)P(A|B)$$

2. 朴素贝叶斯分类$Na\ddot{i}ve\ \ Bayesian\ \ classifier$

---

   1) 设D是训练元组和相关联的类标号的集合, 每个元组用一个n维属性向量$X=\{x_1,x_2,\cdots,x_n\}$表示, 描述由n个属性$A_1,A_2,\cdots,A_n$对元组的n个测量.
   2) 假定有m个类$C_1,C_2,\cdots,C_m$. 给定元组X, 分类法将预测X属于具有最高后验概率(条件X下)的类. 即, 朴素贝叶斯分类法预测X属于类$C_i\Leftrightarrow P(C_i|X)>P(C_j|X)\quad 1\le j\le m,j\ne i$, 最大化$P(C_i|X)$. $P(C_i|X)$最大的类$C_i$成为最大后验假设.

$$P(C_i|X)=\frac{P(X|C_i)P(C_i)}{P(X)}$$

   3) $P(X)$对于所有类是常数, 只需要$P(X|C_i)P(C_i)$最大即可. 如果类的先验概率未知, 则通常假定这些类是等概率的, 即$P(C_1)=P(C_2)=\cdots=P(C_m)$, 并据此对$P(X|C_i)$最大化. 否则, 是最大化$P(X|C_i)P(C_i)$. $P(C_i)=|C_{i,D}|/|D|$估计, 其中, $|C_{i,D}|$是D中$C_i$类的训练元组数.
   4) 给定具有许多属性的数据集, 计算$P(X|C_i)$的开销可能非常大, 但由于类条件独立的朴素假设, 即, 给定元组的类标号, 假定属性值有条件地相互独立, 在属性间, 不存在依赖关系.

$$P(X|C_i)=\prod_{k=1}^{n}P(x_k|C_i)=P(X_1|C_i)\times P(X_2|C_i)\times\cdots\times P(X_n|C_i)$$

   $x_k$表示元组X属性$A_k$的值.

(a) 如果$A_k$是分类属性, 则

$$P(x_k|C_i) = \frac{x_k\text{的}C_i\text{类的元组数}}{D\text{中}C_i\text{类的元组数, 即为}|C_{i,D}|}$$

(b) 如果$A_k$是连续属性, 假设连续值属性服从均值为$\mu$, 标准差为$\sigma$的高斯分布

$$g(x,\mu,\sigma) = \frac{1}{\sqrt{2\pi}\sigma}e^{-\frac{(x-\mu)^2}{2\sigma^2}}$$

因此,

$$P(x_k|C_i) = g(x_k, \mu_{C_i}, \sigma_{C_i})$$

- 这里, $\mu$为均值, 就是平均值, $\sigma$ 为标准差, 数据的波动范围.

5) 分类法预测元组X的类标号为

$$C_i \Leftrightarrow P(X|C_i)P(C_i) > P(X|C_j)P(C_j) \text{ 对}1 \le j \le m, j \ne i$$

换言之, 预测的类标号使$P(X|C_i)P(C_i)$最大的类$C_i$.

---

- 当遇到零概率时, 假定D很大, 使得需要的每个计数加1, 造成的估计概率的变化可忽略不计, 但可方便的避免概率值为0的情况, 这种方法叫做拉普拉斯校准, 即, 对q个计数加上1, 还须概率分母上加q. 这些"校准"的概率估计与对应的"未校准"的差别是很接近的, 可忽略不计.

3. 贝叶斯信念网络 由于变量之间的依赖性在实际中是完全有可能存在的, 贝叶斯信念网络说明联合条件概率分布, 允许在变量的子集间定义类条件独立性, 提供一种因果关系的图形模型, 可以对其进行学习.
信念网络的表示方法:
1) 有向无环图, 其中的每个节点代表一个随机变量(实际属性, 隐藏变量), 弧表示一个概率依赖.
2) 条件概率表CPT, $P(Y|Parents(Y))$
Note: 给定其双亲, 每个变量有条件的独立于图中它的非后代.
联合概率:

$$P(x_1, x_2, \cdots, x_n) = \prod_{i=1}^{n} P(x_i|Parents(Y_i))$$

更多的贝叶斯信念网络在page205.

## 3.5 基于规则的分类

1. 使用IF-THEN规则分类: if 规则前件(前提) then 规则结论. 给定类标记的数据集D中的一个元组X, 设$n_{covers}$为规则R覆盖(即满足规则前件)的元组数, $n_{correct}$为规则R正确分类的元组数.

$$coverage(R) = \frac{n_{covers}}{|D|} \quad accuracy(R) = \frac{n_{correct}}{n_{covers}}$$

分别表示规则R的覆盖率和准确率, 前者意味着规则覆盖的元组(即其属性值对规则的前件为真)的百分比, 后者意味着规则覆盖的元组和其中可以被规则正确分类的元组的百分比.
这里面还涉及到冲突的问题, 一般有以下两种解决冲突的策略: 1) 规模序(规则前件的规模度量) 2) 规则度(重要性)

2. 从决策树提取规则 ...

3. 使用顺序覆盖算法的规则归纳 直接从训练数据提取IF-THEN规则, 关联分类算法(association classification algorithm). 流行的算法有AQ, CN2和RIPPER.

## 3.6 用后向传播分类

- 一种流行的神经网络算法
- 神经网络是一组连接的输入/输出单元, 其中每个连接都与一个权重相关联, 在学习阶段, 通过调整这些权重, 能够预测输入元组的正确类标号. 单元之间的连接, 神经网络学习又称连接者学习(connectionist learning), 这里记录个多层前馈(multilayer feed-forward)

多层前馈: 一个输入层, 一个或多个隐藏层, 一个输出层. 一般步骤:

1. 初始化权重, 权重和关联的偏倚(bias)
2. 向前传播输入,

$$I_j = \sum_i w_{ij} O_i + \theta_j$$

$w_{ij}$: 上一层单元i到j的权重. $O_i$: 上一层单元i的输出. $\theta_j$是单元j的偏倚, 充当阈值改变单元的活性. 隐藏层和输出曾的每个单元取其净输入后, 将激励函数(如logistic或sigmod函数)作用于它.

## 3.7 支持向量机Support Vector Machine

使用一种非线映射, 将原训练数据映射到较高的维上, 在新的维上, 它搜索线性最佳分离超平面(即将一类的元组与其他类分离的决策边界), 使用一个适当的对足够高维的非线性映射, 两类的数据总可以被超平面分开. SVM使用支持向量("基本"训练元组)和边缘(由支持向量定义)发现该超平面.
如何寻找最佳超平面, 即是决策平面. SVM通过搜索最大边缘超平面(maximum marginal hyperplane)来处理该问题. 分离超平面, 可记作: $\omega \cdot X + b = 0$ 权重向量$\omega = \{\omega_1, \omega_2, \cdots, \omega_n\}$, $b$为偏倚.

## 3.8 关联分类

- 这里所说的关联分类是基于关联规则分析的分类, 因为类别属性也可以看作是一个特殊的属性, 那么规则中由多个属性推导出一个属性, 可以看成是分类. 比如说一组属性取某些值可以推出什么样的另外一个属性的值, 这个值就可以看成是不同的分类. 所以, 基于规则的分类是可行的.

基本思想: 将搜索频繁模式(属性-值对的合取)与类标号之间的强关联. 关联分类算法: CBA, CMAR, CPAR. page 224

## 3.9 惰性学习法, 或从近邻学习

之前的分类都是急切学习法, 这里的k最近邻分类发和基于案例的推理分类法是基于类比学习. 涉及到相似性度量, 衡量距离问题.

---

k近邻算法思想:

---

1) 遍历整个训练空间, 计算出该文档和训练文档的相似度, 找到K个最相邻的文档
2) 该测试文档与各训练集文档的相似度, 最通常的是余弦相似度
3) 选择K个最相邻的训练文档, 就是最相似的训练文档, K一般为我们设置的值
4) 从这K个最相邻的文档中, 选择类别最多的作为该文档的类别

---

## 3.10 其他分类方法

- 还有很多很多分类方法, 各自基于的理论也不同, 如有遗传算法(群体, 拟合度, 交叉和变异), 粗糙集方法(发现不准确数据或噪声数据内的结构联系, 属性子集选择, 相关分析, 最小属性子集), 模糊集方法(模糊逻辑, 隶属程度), 这些相关理论以后整理下.

## 3.11 预测

连续值, 数值预测对于给定的输入预测连续(或有序)值. 一般采用回归(regression)方法. 回归分析可以用来对一个或多个独立预测变量(已知的)和一个(连续值的)倚赖或响应变量之间的联系缄默. 在数据挖掘环境下, 预测变量是描述元组的感兴趣的属性(即形成属性向量)

1. 线性回归(直线回归分析)

   $$y = b + \omega x$$

   y为响应变量, x为预测变量, b, $\omega$是回归系数, 分别指定直线的Y轴截距和斜率, 也可看作权重, 所以又记为

   $$y = \omega_0 + \omega_1 x$$

   系数可用最小二乘法求解. 假设, 训练数据集 $D(x_1, y_1), (x_2, y_2), \cdots, (x_{|D|}, y_{|D|})$则

   $$\omega_1 = \frac{\sum_{i=1}^{|D|}(x_i - \bar{x})(y_i - \bar{y})}{\sum_{i=1}^{|D|}(x_i - \bar{x})^2} \omega_0 = \bar{y} - \omega_1 \bar{x}$$

2. 多元线性回归 涉及到多个预测变量, 允许响应变量 y用描述元组X的n个预测变量或属性$A_1, A_2, \cdots, A_n$的线性函数建模, $X = (x_1, x_2, \cdots, x_n)$, $D(x_1, y_1), (x_2, y_2), \cdots, (x_{|D|}, y_{|D|})$, $y = \omega_0 + \omega_1 x_1 + \omega_2 x_2 + \cdots + \omega_n x_n$

3. 非线性回归 通过对变量进行变换, 可将非线性模型转换成线性的, 然后用最小二乘方法求解. 如果预测属性多, 仍然需要考虑属性子集的选择和离群点的删除.

4. 其他基于回归的方法 广义线性模型, 用于分类响应变量模型, 2种常见回归: 逻辑斯谛logicist回归, 泊松回归. 对数线性模型, 近似离散的多维概率分布....垃圾...

# 4 聚类

## 4.1 基本定义

1. 聚类Clustering: 是将数据对象分成类或簇的过程, 使同一个簇中的对象之间具有很高的相似度, 而不同簇中的对象高度相异. 相异度根据描述对象的属性值评估, 通常使用距离度量.
2. Cluster: 数据对象的集合. 同一个簇中对象相似, 与其他簇中的对象相异.
   - 基于距离的聚类分析主要有, 基于k均值, k中心点.
   - 给予内存的聚类算法一般选择两种数据结构:
     1) 数据矩阵: 这就是普通对象-属性矩阵
     2) 相异度矩阵: 对象-对象结构, 存储所有成对的n个对象的邻近度.

## 4.2 计算相异度

衡量相异度之前, 需要进行标准化. 这里又区分连续的和离散的两种情况.

1. 区间标度度量: 粗略线性标度的连续度量, 选用的度量单位将影响聚类分析的结果, 也就需要一种标准化策略(外加权重问题)
2. 数据标准化: 怎样将一个变量的数据标准化?将原来的度量转换为无单位变量, 两种方式:
   1) 均值绝对偏差(mean absolute deviation)

$$S_f = \frac{1}{n}(|x_{1f} - m_f| + \cdots + |x_{nf} - m_f|), m_f为f的均值$$

   2) 标准度量值或z-score

$$Z_{if} = \frac{x_{if} - m_f}{S_f}$$

## 4.3 几种距离计算方法

标准化与否, 不管区间标度变量描述的对象间的相异度通常基于每对对象间的距离计算.

1. 欧几里得距离:

$$d(i,j) = \sqrt{(x_{i1} - x_{j1})^2 + (x_{i2} - x_{j2})^2 + \cdots + (x_{in} - x_{jn})^2}$$

2. 曼哈顿(城市块)距离:

$$d(i,j) = |x_{i1} - x_{j1}| + |x_{i2} - x_{j2}| + \cdots + |x_{in} - x_{jn}|$$

3. 闵可夫斯基距离

$$d(i,j) = (|x_{i1} - x_{j1}|^P + |x_{i2} - x_{j2}|^P + \cdots + |x_{in} - x_{jn}|^P)^{\frac{1}{P}}$$

   , P为正整数,

4. 加权的欧几里得距离

$$d(i,j) = \sqrt{\omega_1|x_{i1} - x_{j1}|^2 + \omega_2|x_{i2} - x_{j2}|^2 + \cdots + \omega_n|x_{in} - x_{jn}|^2}$$

   - 距离公式的数学要求:
     1) $d(i,j) \geq 0$,
     2) $d(i,i) = 0$,
     3) $d(i,j) = d(j,i)$,
     4) $d(i,j) \leq d(i,h) + d(h,j)$

- 一般最常用的距离公式就是欧几里得距离, 如果是在二维空间中, 它反应了两点间的坐标距离.
- 不同的变量类型也有不同的相异度计算方法, 但是可以不用管他, 一般在信息检索, 文本文档聚类, 或者是对象-属性结构的非度量情况下, 都可以看成是一个向量. 那么, 这种情况下, 通常使用余弦度量, 如下:

5. 余弦度量

$$S(x,y) = \frac{x^t \cdot y}{\|x\|\|y\|}$$

$x^t$是x的转置, 分子上体现的是共同属性的相对拥有率, $\|x\|$是欧几里得范数. 本质上是向量x, y之间夹角的余弦值. 该值对于旋转和放大是不变的, 但对于平移和普通闲心变换却是可变的. 一种变种, Tanimoto距离:

$$S(x,y) = \frac{x^t \cdot y}{x^t \cdot x + y^t \cdot y - x^t \cdot y}$$

6. 相似度函数选择

- 相似度(距离)函数/规范化聚类分析的数据有很多方法, 度量的选择高度依赖于具体的应用.

## 4.4 主要聚类方法

1. k均值: 基于质心的技术
以k为输入参数, 把n个对象的集合分为k个簇, 使结果簇内的相似度高, 而簇间的相似度低, 簇的相似度是关于簇中对象的均值度量, 可看作簇的质心(centroid)或重心(center of gravity).
处理流程: 首先, 随机地选择k个对象, 每个对象代表一个簇的初始均值或中心, 对剩余的每个对象. 根据其与各个簇均值的距离, 将它指派到最相似的簇, 然后计算每个簇的新均值, 不断重复直到准则函数收敛. 这个准则函数一般是: 平方误差准则

$$\varepsilon = \sum_{i=1}^{k} \sum_{p \in C_i} |lvertp - m_i|^2$$

其缺点: 1) k值, 2) 非凸形状, 3) 离群点, 噪声

- 这里涉及到k的具体数值是几个?

另一个有趣策略, 首先采用层次凝聚算法决定结果簇的数目(即k值), 并找到一个初始聚类, 然后用迭代重定位来改进该聚类(有个迭代过程, 不断更新聚类的中心点).

2. k众数(k-modes方法)
扩展了k均值模式来聚类分类数据. 用簇的众数取代簇均值, 采用新的相异性度量处理分类对象, 采用基于频率的方法更新簇众数.

3. EM(Expectation-Maximization)
期望最大化, 每个对象按照权重(<-对象的隶属概率)指派到每个簇.

4. K中心点方法: 基于代表对象的技术 在每个簇中选出一个实际的对象来代表该簇, 其余的每个对象聚类到与其最相似的代表性对象所在的簇中(仍基于最小化所有对象与其对应的参照点之间的相异度之和的原则来执行, 使用绝对误差标准(absolute-error criterion)

$$E = \sum_{j=1}^{k} \sum_{p \in C_j} |P - O_j|$$

5. 基于层次的聚类方法

- 层次方法(一旦合并或分裂执行就不能修正, 一种解决方法是: 凝聚层次聚类和迭代重定位方法的结合)
  1) 凝聚层次聚类(AGENES)
  2) 分裂层次聚类(DIANA) 3) BIRCH: 利用层次方法的平衡迭代归约和聚类 4) ROCK: 分类属性的层次聚类算法. 5) chameleon: 利用动态建模的层次聚类算法. 动态建模确定簇对之间的相似度, 基于CURE和ROCK的缺点, Chameleon中簇的相似度依据簇中对象的互连度和簇的邻近度来评定, 即若两个簇的互连性都很高并且他们又靠的很近就将其合并.

6. DBSCAN: 一种基于高密度连通区域的基于密度的聚类方法.
   簇: 密度相连的点的最大集合.$\varepsilon$领域, 核心对象, 直接密度可达, 密度相连.
   基于密度的簇是基于密度可达性的最大的密度相连对象的集合, 不包含在任何簇中的对象, 认为是噪声.

7. OPTICS: 通过点排序识别聚类结构.
   为自动和交互的聚类分析计算一个增广的簇排序(cluster ordering), 代表了数据的基于密度的聚类结构. 次序选择根据最小的$\varepsilon$值密度可达的对象, 以便较高密度(较低$\varepsilon$维)的簇先完成.
   核心距离(core-distance, 是使P成为核心对象的最小$\varepsilon'$)和可达距离(reachability-distance, 是P的核心距离和p与q之间的欧式距离之间的较大值).

8. DENCLUE: 基于密度分布函数的聚类
   1) 影响函数(influence function): 描述据点在其领域内的影响. 2) 密度吸引点(density attractor): 全局密度函数的局部最大值.

9. 基于网格的方法
   一种多分辨率的网格数据结构, 它将对象空间量化为有限数目的单元形成网格结构, 所有聚类操作都在网格上进行.
   1) STING: 利用存储在网格单元中的统计信息 2) WaveCluster: 用小波变换方法聚类对象 3) CLIQUE: 高维数据空间中基于网格和密度的聚类方法. 下述...

10. 基于模型的聚类方法
    数据根据潜在的混合概率分布生成, 1) 期望最大方法(EM), 2) 概念聚类: 进行聚类, 后给出特征描述, COBWEB, 3) 神经网络方法, 自组织特征映射(self-organizing feature map, SOM)

## 4.5 聚类高维数据

当维度增加时, 通常只有少数的几维与某些簇相关, 但其他不相关维的数据可能会产生大量的噪声而屏蔽真实的簇. 随着维度的增加, 数据通常会变得更加稀疏, 因为数据点可能大多分布在不同维的子空间中.

1. 特征变换: Feature transformation methods
   主成分分析和奇异值分解, 把数据转换到一个较小的空间, 同时保持对象间原始的相对距离. 通过创建属性的线性组合来汇总数据, 可能发现数据中的隐藏结构. 仅仅适用于那些大部分维都与聚类任务相关的数据集.

2. 试图删除某些维: 属性子集选择(特征子集选择), 用于数据归约, 删除不相关的或荣誉的维(或属性).
   监督学习: 找出与已知类标号最相关的属性集.
   无监督学习: 熵分析(基于性质, 包含紧凑簇的数据值偏低).
   评估函数...
   子空间聚类, 属性子集选择的一种扩展, 在相同数据集的不同子空间中搜索簇群, –> 如何有效俄发现这些子空间簇?

3. CLIQUE
   其聚类过程开始于单维的子空间, 并且向上增长主高维的子空间.
   把每一维划分成网格状的结构, 并且根据每个网格单元包含点的数目来确定该网格单元是否稠密
   <– 基于密度和基于网格的聚类方法的一种集成.
   候选搜索空间的识别基于关联规则中使用的Apriori性质, k-1维不稠密 –> k维不稠密.
   随输入规模线性地伸缩, 当数据维数增长时具有良好的可伸缩性, 获得有意义的聚类结果依赖于正确的调整网格大小和密度阈值.

4. PROCLUS: 维归约子空间聚类方法(投影聚类)
   从高维的属性空间中寻找簇的初始近似开始, 对每维为每个簇赋值一个权重, 并且在下一轮迭代中使用这些更新的权重产生簇.
   曼哈顿分段距离(Manhattan Segmental distance), 即一组相关维的曼哈顿距离.
   PROCLUS三阶段, 初始化, 迭代和聚类改进, 依次如下:
   1) 使用贪心算法选择一组彼此距离很远的初始中心点, 确保每个簇都能由其中至少一个对象来代表
   2) 从归约后的集合(中心点)中随机选择k个中心点, 选择平均距离小于统计期望值的维的集合.
   3) 基于已发现的簇, 为每个中心点计算新的维, 把数据点重新分配给中心点并删除离群点.

## 4.6 基于频繁模式的聚类

搜索大型数据集合中频繁出现的模式, 项或对象的集合, 发现的频繁模式也可能预示簇, 非常适合于高维数据. 另一种聚类高维数据的有趣方法是基于维子集对象间的模式相似度.
pCluster: 通过微阵列数据分析中的模式相似性聚类,,,对维子集内聚模式(coherent pattern),,平移模式, 双聚类..

## 4.7 基于约束的聚类分析

允许用户指定一个焦点,,,指导挖掘算法发现用户感兴趣的那类"知识"

## 4.8 离群点分析

略. page 295

# 5 多关系/序列/数据流挖掘

## 5.1 多关系数据挖掘

- Multirelational data mining , MRDM.
- 之前所说的分类或聚类都是针对一个关系表来说的, 而在现实中的数据库中, 大大存在多个关系数据表, 如果对这多个表利用传统的单表挖掘方法必定要将所有表链接成一个大表才能在上面进行挖掘. 而多关系数据挖掘解决的问题就是对多个表不进行物理连接, 直接在多个表上进行挖掘的一种理论.
  多关系分类, 聚类, 频繁模式挖掘等等. 具体涉及到的知识也有很多.

## 5.2 数据流挖掘

- 数据流以不同的更新速度连续地流进和流出系统, 按时间顺序快速变化的, 海量的和潜在无限的, 在不规则中发现规则的东西. 它一般仅单遍扫描的, 联机的, 多层多维的处理和分析.
- 把数据流看成是自然界的水流, "你不可能两次踏进同一条河", 正确性和存储空间之间进行平衡, 近似而不是精确的结果, 也就是说, 满足一定精确度条件下使用最少的资源消耗获得结果. 主要涉及的方法有: 随机抽样(水库抽样, 无放回地选取s个元素的无偏随机样本), 滑动窗口模型(仅基于最近的数据作出决策, 在每个时刻t, 一个新的数据元素到来, 该元素在时刻t+w过期, 其中涉及到冲突, 窗口溢出处理, 衰减处理), 直方图(流中元素的频率分布), 多分辨率方法(数据归约, 分治策略, 平衡二叉树, 使用聚类方法将流数据组织成一个层次结构的树), 小波(用来构建输入信号的多分辨率层次结构,,,), 梗概(频率矩), 随机算法(拉斯维加斯算法, 蒙特卡罗算法)...page 308

## 5.3 序列挖掘

- 针对时间序列数据库, 其特点数据量超大, 快速, 具有时间信息. 具体由不同时间点的重复测量的值或事件的序列组成.
- 序列数据库, 记录带有或不带有具体的时间概念的有序元素或事件的序列组成, 序列模式挖掘就是发现频繁发生的有序时间或子序列模式.
- 时间序列数据库, 由不同时间重复测量得到的值或事件的序列组成, 这些值通常是在相等时间间隔测量.
- 序列数据库, 由有序事件序列组成任何数据库, 这些事件不一定具有具体的时间概念.
- 趋势分析, 时间序列建模, 发现趋势和离群点, 回归分析.
- 时间序列建模也归结为将时间序列分解为趋势(T), 周期性(C), 季节性(S)和无规则的运动(I), 即

$$\text{时间序列变量} Y = T \times C \times S \times I$$

- 简单列举一下序列模式挖掘算法, 不进行深入. page 326

1. GSP: 水平数据格式, $< sequence - ID : sequence of itemsets >$
2. SPADE: 垂直数据格式, $< itemset : sequence - ID, event - ID >$
3. Prefix Span: 模式增长方法, 不需要候选产生.

- 它们的基本思想仍然是基于Apriori性质, 即序列模式的每个非空子序列都是序列模式, Apriori的反单调性.
- 基于约束的序列模式挖掘,,,把约束推进到挖掘过程中, 由于没有用户或专家指定的约束, 挖掘可能产生大量无意义的模式.
- 约束的形式: 属性, 属性值之间的联系或结果模式中的聚集, 正则表达式, 限定搜索空间, 结果模式的有趣程度, 挖掘效率.

# 6 Markov模型

在马尔可夫链模型中, $x_l$的概率仅依赖于前一个状态$x_{l-1}$的值, 而不是它前面的整个序列的概率, 此为马尔可夫性质.

$$P(x) = P(x_l|x_{l-1})P(x_{l-1}|x_{l-2})\cdots P(x_2|x_1)P(x_1) = P(x_1)\prod_{i=1}^{l} P(x_i|x_{i-1}$$

$P(x_i|x_{i-1})$是$a_{x_{i-1}x_i}$的转移概率, 则

$$P(x) = \prod_{i=1}^{l} a_{x_{i-1}x_i}$$

todo: 待补充.....

# 7 潜在语义索引

LSI: Latent Semantic Indexing. 把原始的向量空间转换成潜在语义空间, 文档就在转换后的语义空间上进行表示和比较.

$$X = U_r \sum_r V_r^T \approx U_k \sum_k V_k^T$$

选择的是最具代表性的特征.
本质上是矩阵论中的奇异值分解.

# 8 杂七杂八的东西

## 8.1 文本挖掘

1. 文档选择: 对选择相关文档指定约束条件, 布尔检索模型
2. 文档秩评定: 基于不同的数学基础, 查询中的关键词和文档中的关键词进行匹配, 根据匹配查询的程序给每个文档打分. 目的是根据一些信息, 如词在该文档和整个集合的频率, 计算出得分近似估计文档的相关程度....

## 8.2 PageRank

从许多优质的网页链接过来的网页, 必定还是优质网页.
3个要点: 1) 反向链接数(单纯的意义上的受欢迎程度指标, 2) 反向链接是否来自推荐度高的页面(有根据的受欢迎指标), 3) 反向链接原页面的链接数(被选中的几率指标)
(被许多页面链接的)受欢迎的页面必定是优质的页面.
(汇集着许多推荐的)好的页面所推荐的页面必定也是同样好的页面.
与感觉在被胡乱链接的链接相比, 被少数挑选出的链接肯定是优质的链接.

HITS算法, 权威页面(authority, 权威值), Hub网页(hub, 中心值)

## 8.3 Vector Space Model

向量空间模型, 将文档和查询都表示成对应于所有关键词的高维空间中的向量并使用适当的相似性度量计算查询向量与文档向量之间的相似度. <– 评定文档的秩.

## 8.4 文本索引技术

1. 倒排索引, 对同义词, 多义词处理比较麻烦.
2. 特征文件, 存储DB中每个文档的特征记录的文件位.
   多对一映射, 经检索-> 分析->词根处理和检索
   散列索引技术和重叠编码技术将词表编码为位串表示.

## 8.5 文本检索技术

相关内容来自一个PPT, 忘了名字

1. SMART系统: 基于VSM的文本信息检索系统
2. Okapi系统: 基于概率检索模型
3. Lemur Toolkit系统
4. Lucene: 全文索引软件包. 一个基于Lucene实现的Search Engine, nutch.

## 8.6 分布式Web信息检索技术研究

综述性质, 分布式信息检索主要过程:
1) 文档集合划分
2) 集合选择
3) 单文档集合检索
4) 结果合并
ljljlj