

数据挖掘名词解释/简答题、计算题复习

一、题型

判断题！丢分最多了

计算题

简答题，例如 PCA 怎么做？

二、复习提纲

1. 数据挖掘：数据挖掘是从巨量数据中获取正确的、新颖的、潜在有用的、最终可理解的模式的非平凡过程
2. KDD: Knowledge Discovery in Data
3. 数据仓库：支持管理决策过程的面向主题的、集成的、长期的、稳定的数据集合
4. OLAP 和 OLTP？主要回答有颜色的即可
 - a) OLTP：联机事务处理
 - b) OLAP：联机分析处理

	OLTP	OLAP
users	clerks, IT professionals	managers, executives
function	day-to-day operations	decision support
DB design	application-oriented	subject-oriented
data	current, up-to-date detailed, flat relational isolated	historical, summarized, multidimensional integrated, consolidated
access	read/write	mostly read
unit of work	short, simple transaction	complex query
number of accessed records	tens	millions
number of users	thousands	hundreds
DB size	GB to high-order GB	>=TB
metric	transaction throughput	query throughput, response time

5. 多维数据库组织：星型、雪花型、事实星座型
6. 概念层级：定义一个映射序列，将底层概念集映射到较高层、更一般的概念
7. 模式层次结构：形成数据库模式中属性的全序或偏序的概念分层
8. 集合-分组层次结构：将给定维或属性的值离散化或分组来定义概念分层
9. OLAP 操作：
 - a) 上卷、下钻、切片和切块、转轴
 - b) 聚集函数：分布的 (count、sum)，代数的 (avg)，整体的 ()
10. A data cube measure is a numeric function that can be evaluated at each point in the data cube space
11. ROLAP (Rational OLAP, 关系数据库) 和 MOLAP (Multidimensional OLAP, 稀疏矩阵)
 - a) ROLAP: Use relational or extended-relational DBMS to store and manage warehouse data
 - i. 优化技术，慢，但良好的拓展性
 - b) MOLAP: Array-based multidimensional storage engine (sparse matrix techniques)
 - i. 多路数组聚合，快
 - c) HOLAP (Hybrid OLAP, ROLAP 拓展性+MOLAP 快速)、Specialized SQL Servers
12. 预计算：部分 cuboids 的预计算较好：trade-off

a) Partial materialization: trade-off

• How many cuboids are there in an n-D data cube (lattice) ?

– If there is no hierarchy

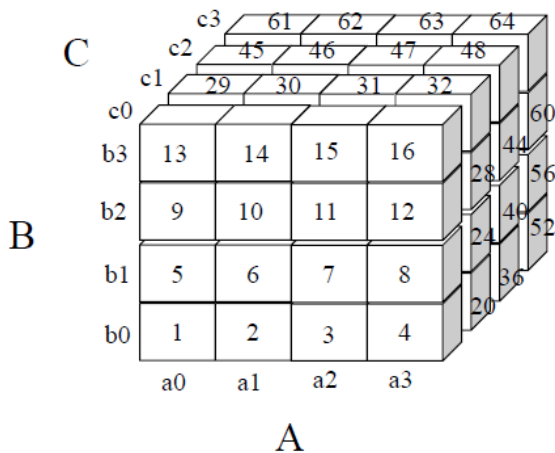
$$T = 2^n$$

– If each dimension is associated with L_i levels of hierarchy

$$T = \prod_{i=1}^n (L_i + 1)$$

for 10 dimensions each with 4 Levels, $T = 5^{10} \approx 9.8 \times 10^6$

13. 多路数组聚合：计算？——考试会考



Suppose the size of the array for each dimension, A, B, and C, is 40, 400, and 4000

What is the best traversing order to do multiway aggregation?



升序排序最节省空间，即：BC，AC，AB 顺序计算，仅需 $100 \times 1000 + 40 \times 1000 + 40 \times 400 = 156000$

14. OLAP 和 DM 的区别？目标、功能、数据

- a) 目标：OLAP：迭代、DM：自动化
- b) 功能：OLAP：用户驱动的数据总结和比较、DM：广泛功能
- c) 数据：OLAP：Data warehouse、DM：不止 Data warehouse

15. 为什么合并 OLAP 和 DM？

Why integrating OLAP with data mining?



- High quality of data in data warehouses
 - DW contains integrated, consistent, cleaned data
- Available information processing infrastructure surrounding data warehouses
 - ODBC, OLEDB, Web accessing, service facilities, reporting and OLAP tools
- OLAP-based exploratory data analysis
 - Select portions of relevant data, analyze at different granularities, visualization tools
- On-line selection of data mining functions
 - Flexibility to select data mining functions and swap data mining tasks dynamically

16. AOI: 面向属性归约
- a) 首先用数据库查询收集任务相关的数据, 然后通过考察任务相关数据中每个属性的不同值的个数进行泛化
 - b) 框架: Data Focusing + Data Generalization(attribute removal + attribute generalization) + Presentation
 - c) 属性太多就删除(较高层概念用其他属性表示或没有泛化操作算子, 如名字、电话号码等)或泛化(大量不同值且有泛化操作算子), 元组太多就泛化
17. t-weight, d-weight 计算? ——考试会考~
- a) t-weight: 类特征描述如果 X 在 target_class 中, 则 X 满足 condition_i 的可能性是 w_i
 - b) d-weight: 类比较描述如果 X 满足 condition_i, 则 X 在 target_class 中的可能性为 w_i
18. d-weight 是必须的, 但是 t-weight 不必须
19. 数据预处理: **数据清洗、数据约简、数据集成、数据转换**
- a) 数据清洗: 填充缺失值、平滑噪声、修正不一致数据
 - i. binning: 等宽、等深(平均数、中位数、桶界限)
 - b) 数据约简: 简化数据表示, 同时还能获取相似结果
 - c) 数据集成: 集成数据库、data cubes 等, 处理冗余
 - d) 数据转换: 正则化、平滑、总结、概念级爬升 (Generalization)、属性创建
 - i. 正则化:
 - 1. min-max
 - a) 优点: preserves the relationship among the original data values
 - b) 缺点: may encounter "out of boundary" error
 - 2. z-score
 - a) 优点: particularly useful when the actual min and max of the attribute value are unknown, or when there are outliers dominating the min-max normalization
 - b) 缺点: change the original data
 - 3. decimal scaling
20. 信息增益: 计算? ——考试会考, 结合决策树看
21. 降维:
- a) PCA? 会了吗
 - b) 流形、小波
22. 数值规约
- a) 直方图: 等深、等宽、最小方差、maxdiff
 - b) 聚类
 - c) 采样
 - d) 离散化
 - e) 描述性数据概括:
 - i. 均值、中值、众数、midrange $((\min + \max)/2)$ 、分位数、散度
 - ii. 箱图、分位数图会看了吗?
 - iii. 分位数图和分位数-分位数图的区别?
 - 1. 分位数图是一种观察单变量数据分布的方法
 - 2. 分位数-分位数图对着另一个对应的分位数, 绘制一个单变量分布的分位数, 使得用户可以观察从一个分布到另一个分布是否有漂移
23. PCA、LDA 和 NCA
24. 关联规则挖掘: 在给定的数据集中寻找项之间的有趣关系

- a) Body->Head[support, confidence]
- b) $\text{support}(A \rightarrow B) = P(A \text{ and } B)$
- c) $\text{confidence}(A \rightarrow B) = P(B|A)$
- 25. 强规则: rules 满足 min_sup 和 min_conf
- 26. k-项集: itemset 包含 k 个 item
- 27. itemset 满足 min_sup, 就是一个频繁项集
- 28. 关联规则挖掘流程: ——会考计算
 - a) 找频繁项集: Apriori、FP-growth
 - i. Apriori: 连接、剪枝, 计算? 伪代码中为什么? 理解了吗?
 - 1. 先验知识: all non-empty subsets of a frequent itemset must also be frequent
 - 2. 如果 I1I3I4I5 要由 I1I3I5 和 I1I4I5 生成, 那么它一定首先被 I1I3I4 和 I1I3I5 生成, 但是实际上它并没有, 说明 I1I3I4 一定不是频繁项集, 同时也说明 I1I3I4I5 肯定也不是频繁项集了 (逻辑关系理清, 就知道为什么算法里面要求前 k 项相同, 仅最后一项不同才可以判断了)

✓ for example, without those constraints, {I1, I3, I5} and {I1, I4, I5} will be joined to {I1, I3, I4, I5}

SETM algorithm works in this way [Houtsma & Swami, Research Report at IBM Almaden Research Center 1993]

✓ however, in Apriori, if {I1, I3, I4, I5} is a frequent itemset, it will be generated from {I1, I3, I4} and {I1, I3, I5}. Otherwise it will **NOT** be generated

- 3. AIS 和 Apriori 的区别!
- 4. AprioriTid
- 5. DHP
- 6. Apriori 的优化:
 - a) 基于散列的计数: 散列项集到对应的桶中
 - b) 事务压缩: 压缩进一步迭代扫描的事务数量
 - c) 划分: 为找候选项集划分数据 (全局频繁项集至少要是分区的一个分区的频繁项集)
 - d) 抽样: 对给定数据的一个子集上挖掘 (近似结果)
 - e) 动态项集计数: 在扫描的不同点添加候选项集
- ii. FP-growth: 计算? 画图等等, 生成的结果?
 - 1. Apriori 的缺点:
 - a) it is costly to handle a **huge number of candidates**
 - b) it is tedious **to repeatedly scan the database** and check a large set of candidates by pattern matching
 - 2. 为什么 FP-growth 高效?
 - a) The mining process works on a set of pattern-bases and conditional FP-trees that are usually much **smaller than DB (only scan the DB twice)**
 - b) The mining operations consist of mainly **prefix count adjustment, counting local frequent items**, and **pattern fragment concatenation** which is much less costly than generation and test of a very large number of candidate patterns
 - 3. 如果有分支, 就不断抽取条件模式基和条件 FP-tree
- b) 由频繁项集产生强关联规则: 即对所有频繁项集 I, 产生所有非空子集 s, 当 $\text{sup}(I)/\text{sup}(s) > \text{min_conf}$

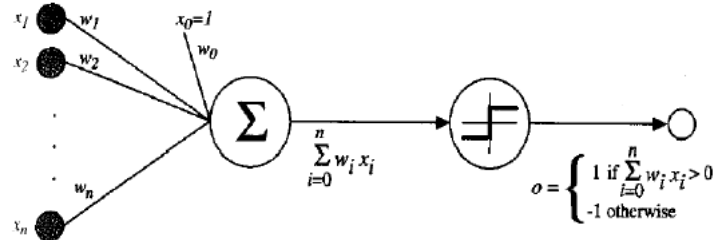
时输出 $s \rightarrow l-s$

29. 闭频繁项集:
 - a) 闭: 一个 itemset X 和一个 data set S , 如果没有真超集 Y 使得 Y 和 X 在 S 中有相同的 support 计数, 则 X 是闭的 (无论再加一个什么东西, 当前项集的支持度都会下降 (但还可能是频繁项集))
 - b) 闭频繁项集: X 既 closed 又 frequent
 - c) 闭频繁项集包含所有信息, 包括频繁项集和支持度计数
30. 极大频繁项集: X 频繁, 且没有真超集 Y 使得 Y 在 S 中频繁 (即 X 是最大的频繁集, 再加什么都不频繁了)
 - a) 极大频繁项集只包含所有频繁项集, 但是支持度计数不确定
31. 闭频繁项集和极大频繁项集的判断? 会判断了吗
32. 多层关联规则挖掘: 每种方法是什么? 优缺点?
 - a) 相同 min_sup
 - b) Level-by-Level 独立
 - c) Level-cross filtering by k-itemset
 - d) Level-cross filtering by single item
 - e) Controlled level-cross filtering by single item
33. 多维关联规则挖掘
 - a) k-谓词集: 包含 k 个连接谓词的集合
 - b) 静态离散化: 预定义的概念层次结构对量化属性 (如 age、income、price) 进行离散化
 - c) 量化关联规则: 根据数据分布动态分到 bin 中 (可动态组合), 找频繁谓词集, 然后聚类
 - d) 基于距离的关联规则: 找间隔或聚类, 然后找最经常出现的 cluster 的 group
 - i. 用 density threshold 替换 support
 - ii. 用 degree of association 替换 confidence
34. 对支持度、置信度的批评: 未必有真正的价值, 因为它是因果, 不是关系
35. 所以有了关联分析, 用 lift 替换 conf
 - a) $\text{lift}(A,B)=P(A \cup B)/(P(A)P(B))$, $=1$ 独立, <1 负相关, >1 正相关
36. 泛化的评估
 - a) 分类: 精度、损失、Precision、Recall、F1、AUC 公式?
 - b) 回归: MSE
 - c) 排序: Ranking loss (有多少个倒序)、MAP、NDCG 公式?
37. 方法
 - a) 留出法, 优点? 缺点?
 - b) 交叉验证法, 优点? 缺点?
 - c) 留一法
 - i. 优点: 交叉验证的优点+适用于小数据集
 - ii. 缺点: 大数据集的时候开销很大+留一法的验证集不能代表整个数据集 (分布不同)。例如一种算法: 把类别预测为出现次数最多的类。现在: 正类负类一样多。用留一法, 验证的正确率一定是 0%! 这说明留一法不一定好的。
38. 贝叶斯错误率
39. 判别式和生成式
40. 决策树——考试会考计算!
 - a) 停止划分的条件 (同一类, 没有属性供划分, 没有样本落到节点)
 - b) 信息增益: 公式? 偏好拥有大量取值的属性
 - c) 信息增益率: 公式? 偏好小取值的属性

- d) 剪枝：为什么剪枝？预剪枝和后剪枝分别是怎么回事？后剪枝更精确但计算代价更大
- e) 连续值问题？缺失值处理？增量式方法（动态调整树）？属性创建（旋转坐标轴）

41. 神经网络

- a) 感知机：无法解决异或问题



- b) BP 算法
 - i. 保证局部最小值
 - ii. 全局最小？随机初始化和随机梯度下降等
- c) 神经网络的表达能力
 - i. 2 层单元 \rightarrow 所有布尔函数
 - ii. 2 层单元，其中 1 层是 sigmoid 隐含层和 1 层线性层 \rightarrow 有界连续函数
 - iii. 3 层单元，2 层 sigmoid 隐含层 \rightarrow 任意函数，找不到，证明不了 $P=NP$

42. SVM

- a) margin 边界？两个类别的距离，目标是最大化最小距离
- b) SVM 可以保证求到函数的最优值，但是求不到问题的最优值，原因是存在很多很多 ϕ 可以作为核函数，同样不能证明 $P=NP$

43. 贝叶斯分类器：计算——考试会考


- a) 类条件独立性假设
- b) 会不会计算了？
- c) 拉普拉斯修正？怎么计算？自己手算一遍

44. 贝叶斯网

- a) 有向无环图，条件概率表

45. 集成学习

- a) 好而不同：所有集成学习器的平均错误-它们之间的差异——平均错误越小（平均性能越好），学习器之间的差异越大，集成学习的误差就越小（学习效果越好）
- b) 并行的方法：Bagging，减少 variance
 - i. bootstrap 抽样，每次抽样作为一个训练集，剩下的做验证集
 - ii. 分类：一人一票投票，取票数最高
 - iii. 回归：平均
 - iv. 效果：各子模型独立，可以显著降低学习器之间的 variance，bagging 选择的是具有一定相关的子模型，因此可以降低 variance；使用同种模型，bagging 后的 bias 接近子模型的 bias，因此不能显著降低偏差
- c) 串行的方法：Boosting，减少 bias
 - i. weighted combination：因为每个 Learner 关注的相关，而后面的 Learner 可能更关注偏题难题怪题，所以如果一人一票的话后面的 Learner 可能都不会关心前面比较简单的样本了，加权会更好一些
 - ii. 效果：最小化损失函数，降低偏差；但因为序列化的强相关策略，boosting 子模型的线性组合就不能显著降低方差

46. KNN: 分类、回归方法方法, lazy learning
- a) 降低对离群点的敏感性? k-中心点算法, 但是 k 中心点算法复杂度更高, 需要取舍
47. 预测模型的评估函数
- a) 什么情况下 MSE 不理想? 情况是什么?
 - i. 回归问题的时候离群点会产生很大的偏差
 - b) ϵ -不敏感性: 在 epsilon 里面的才考虑, 避免离群点的影响
48. 距离度量的要求: 非负性、自反性、对称性、三角不等式
- a) 闵可夫斯基距离: 欧氏距离、曼哈顿距离
 - i. 基本假设: 每一维的重要性相同
 - b) 加权的闵可夫斯基距离: 权重反应不同属性的重要性关系
 - c) 马氏距离: cov 协方差矩阵, 也描述了属性的协方差
 - d) 0-1 数据: 汉明距离、sim、Jaccard、Dice
 - e) 名词性: 分成 binary 属性、VDM
 - f) 复杂结构:
 - i. 分布: KL 散度、交叉熵
 - ii. 树、图: 核
49. 划分聚类
- a) 表示一个簇的方法: 平均或周围的数据
 - b) 代表方法: k-means、k-medoids、k-modes
 - c) k-means: cluster 由这个 cluster 的平均代表
 - i. 为什么聚类可以得到还算满意的结果? 因为我们目标是类内距离足够小, 类间距离足够大, 而聚类结果能够在当前的类别数目中获得每一个簇内距离相对最小的结果聚类欲最小化的目标函数 $d = \sum_{j=1}^K \sum_{x \in m_j} (x_i - m_j)^2$ 每次更换新的中心, m_j 就变了, 我们能够让 $(x_i - m_j)^2$ 更小收敛性: 不断降低且下界 0, 所以收敛
 - ii. 缺点: 存在局部极小 (因为确定点属于哪个 cluster 这一步不是 convex 的)
 - d) k-medoids: cluster 由中心附近的一个点来代表
50. 层次化聚类: 保持历史记录、延伸到两个极端 (只有一个 cluster 和 N 个 cluster)
- a) AGNES: Bottom up, 两个 cluster 之间最小欧几里得距离结合
 - b) DIANA: Top down, cluster 内最大欧几里得距离拆开
- 

这两个就可以merge起来
- c) 问题: 簇的距离

这个是缺点, 也可能是优点

 - i. 最近两点距离
 - 1. 优点: 可发现非球形的簇, 序不敏感
 - 2. 缺点: 方差敏感, 离群点敏感, 不容易分割
 - ii. 最远两点距离
 - 1. 优点: 方差不敏感, 离群点不敏感, 分割均匀
 - 2. 缺点: 倾向于产生球形聚类, 同样远的时候对连接顺序敏感
 - iii. 均值距离: 两个簇均值的距离
 - iv. 平均距离: 两个簇点距离的均值
51. 密度聚类, 关键: 定义邻居和阈值
- a) DBSCAN: 种子点, 直接密度可达, 密度可达, 密度连接 (通过 P 密度可达的两个点是密度连接的) 的定义
52. 模型聚类

- a) GMM, EM 算法: 希望找到带有隐变量的概率模型的极大似然估计
 - i. E: 估计给定数据集 D 和模型参数 θ^k 的时候对隐变量 Z 的期望
 - ii. M: 用极大似然估计求最大的 θ
 - iii. 收敛: 每次 F 都增加, 并且上界 L
 - iv. EM 需要确定的: 似然函数、模型参数、隐变量、期望、目标函数
- b) SOM: 每次计算, 总有一个神经元在数据点附近, 然后它牵扯的点就会向数据靠近。不断迭代, 所有神经元都会集中到数据里面

53. 离群点类型: 全局离群点、情境离群点 (依赖条件的离群)、集体离群点 (一些对象作为整体偏离)?

54. 离群点方法: 学习、假设 (统计、估计、聚类、信息论)

55. 统计的离群点检测方法

- a) 参数化:
 - i. 单变量: 3σ , 箱图
 - ii. 多变量: 转化成单变量、用卡方检验 (大的 outlier)
- b) 非参数化
 - i. 直方图、核密度估计

56. 基于估计的离群点检测方法

- a) 基于距离
 - i. DB-outlier: 距离当前点 o' 小于等于 r 的点的数目占总点数是不是过少

$$\frac{|\{o' \mid \text{dist}(o, o') \leq r\}|}{|D|} \leq \pi$$

- 1. 计算代价高
- ii. CELL: grid 的思想且对角线 $r/2$, 周围 1 的宽度为 1, 2 的宽度为 2

1. 对角线 $r/2$? 为了方便找到 $\leq r$ 的点的数量
is the number of objects in level-1
因为只用了周围红框里的点都让他不是 outlier 了, 即使再加上边上比红框多出部分的点, 它就更不可能是一个 outlier 了

• **Level-1 pruning rule:** if $a + b_1 > [\pi n]$, then every object o in C is NOT a DB(r, π)-outlier

因为用了 2 方框里面的所有点都不够, 即使用绿色圆里面的点也肯定更不够了, 所以 C 里面的点就更是 outlier 了

• **Level-2 pruning rule:** if $a + b_1 + b_2 < [\pi n] + 1$, then all objects in C is DB(r, π)-outliers

不管 C 里面怎么样, 只要离别人都太远, 即 b_2 里点的数目不够多, 那就说明这里头都是 outlier

2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2
2	2	1	1	1	2	2	2
2	2	1	C	1	2	2	2
2	2	1	1	1	2	2	2
2	2	2	2	2	2	2	2
2	2	2	2	2	2	2	2

- b) 基于密度 (相对密度)?
 - i. CELL 只考虑了全局视角, 没考虑本地属性

57. 基于聚类的离群点检测方法

- a) 基本假设: outlier 不属于聚类, outlier 离 cluster 最近的点距离很大, 稀疏小簇内的点都是 outlier

58. 基于分类的离群点检测方法

- a) 观察: 严重的不平衡性, 同时这也是挑战
- b) one-class SVM:
 - i. v-SVM: 意思是把所有正常点投影到超平面, 让所有正常点都远离原点
 - ii. SVDD: 把正常点限制在一个半径小的球中

59. 情境离群点: 转换成传统的 outlier 检测, 或建模正常的行为或上下文来查找 outlier

60. 集体离群点：探索数据的结构
61. 高维数据离群点：
- a) 方法一：降维，然后再子空间里面算距离，最后用基于估计的方法找
 - b) 方法二：把特征空间切成小块，对每个小块计算离群点分数找离群点
 - c) 观察：Outliers are few and different. Thus, when randomly split the space into small region, an outlier is more likely to be ISOLATED
 - d) iForest:
 - i. 思想：类似决策树，在空间中轴平行地划分，看看要划分多少次才能把这个点和别的点分开。划的越多，就越不容易分开就越可能不是 outlier；反之容易划分的更可能是 outlier
 - ii. 方法：
 - 1. 下采样，然后随机选择属性划分，直到找到 isolation
 - 2. 每次找都有一个 path length，多次平均，就能计算出离群点分数，从而找到离群点
62. 表示文本：词袋模型
- a) Binary、Frequency、TFIDF
 - b) 特点：高维、稀疏、非负、边际信息
63. 预处理：停用词处理、形态转化、标点符号去除、标签去除、WebData 的主页面
64. 文档聚类：欧氏距离没有考虑文本的特点
- a) cosine 相似度：意思是看每一维有没有相似来度量相似性，而如果用欧氏距离的话可能因维度之间范围不一样而差很多
 - b) centroid，可以找到主题词：群集中的低频单词会被投射出来，并且每个群集的词汇量只有一部分被重新训练。其余的单词称为主题词，这些低频词汇可能会导致长尾而导致稀疏，会误导结果，需要过滤掉，留下来的那些主题词就可以用来代表文档了
 - c) 概率的方法：EM
65. 协同聚类：二分图（一边文档一边词）求图割，n cluster 就割到 n 个子图，同时去掉最少的边，剩下的就是聚类
66. 主题建模
- a) 潜在语义分析 (LSA)：相似单词会在相似文本中出现
 - i. 保留 Top-K 的奇异值分解
 - ii. 优点：降维、克服同义和多义的问题：即虽然词汇不同，但表达的语义相似
 - b) 概率建模近似 (PLSA)：EM
 - c) LSA 和 PLSA 关系？
 - i. 相同：LSA 中左奇异矩阵是 topic 和文档的关系，右奇异矩阵是 word 和 topic 的关系，中间是 topic 的概率；PLSA 中乘积是 given topic 的情况下文档的概率和 given topic 情况下词的概率乘一起
 - ii. 不同：
 - 1. LSA 中 P_k 是列向量的正交矩阵，而 PLSA 写成向量不正交
 - 2. PLSA 中 P_k 可以用来推断主题的主题词，但 LSA 中不可以
 - d) PLSA 和 EM 区别？
 - i. PLSA 根据隐变量选择文档-词矩阵的位置，EM 根据隐变量生成文档-词矩阵的每一行
67. 文本分类
- a) cosine 相似度
 - b) LSA+cosine 相似度：如果太稀疏，就先 LSA 再 cosine，让它紧致
 - c) Centroid-based (clustering+cosine 相似度)：每个文档类别中聚类，再在这些类别中找到代表的 centroids，然后用这些代表点算 kNN

nearest neighbor of

- d) Rocchio's method, 问题在于特殊 case: each cl 会判定成正类, 但实际上应该是负类
- e) 伯努利朴素贝叶斯: 认为词和词之间没有关系
 - i. 缺点: 可能会被没有出现的词所主导, 而非那些已经出现的词, 因为所有文档里面, 每个单词不出现的话更经常只是单纯地不出现, 而非这里认为是某种机制存在而导致他不出现
- f) 多项朴素贝叶斯: 认为从词典中多次有放回采样单词
 - i. 没有出现的词在这里完全不考虑了! 认为文章是从这些相关的词表中不停采用词来写作的
- g) 线性支持向量机: 支持向量机的好处来源于文档词表示也是稀疏的, 通常用线性核已经足够了
 - i. OP1 和 OP2 的区别?
 1. 关系: 一一对应
 2. OP1 要求每一个样本都要有一个 Loss, 会慢, 而 OP2 把所有 Loss 综合考虑, 因此 OP2 更快