信息检索与数据挖掘

第11章 文本聚类

5月18日周一 第12章 Web搜索
5月22日周五 图像分类的算法思想【第13章 多媒体信息检索 & 第14章 其他应用简介】
5月25日周一 总复习
5月28日周四晚 答疑【不讲课】
6月1日周一 考试

5月18日周一 第12章 Web搜索
5月22日周五 图像分类的算法思想【第13章 多媒体信息检索 & 第14章 其他应用简介】
5月25日周一 总复习
5月28日周四晚 答疑【不讲课】
6月1日周一 考试

中国科学技术大学教学日历表
（2015年春、夏季学期）

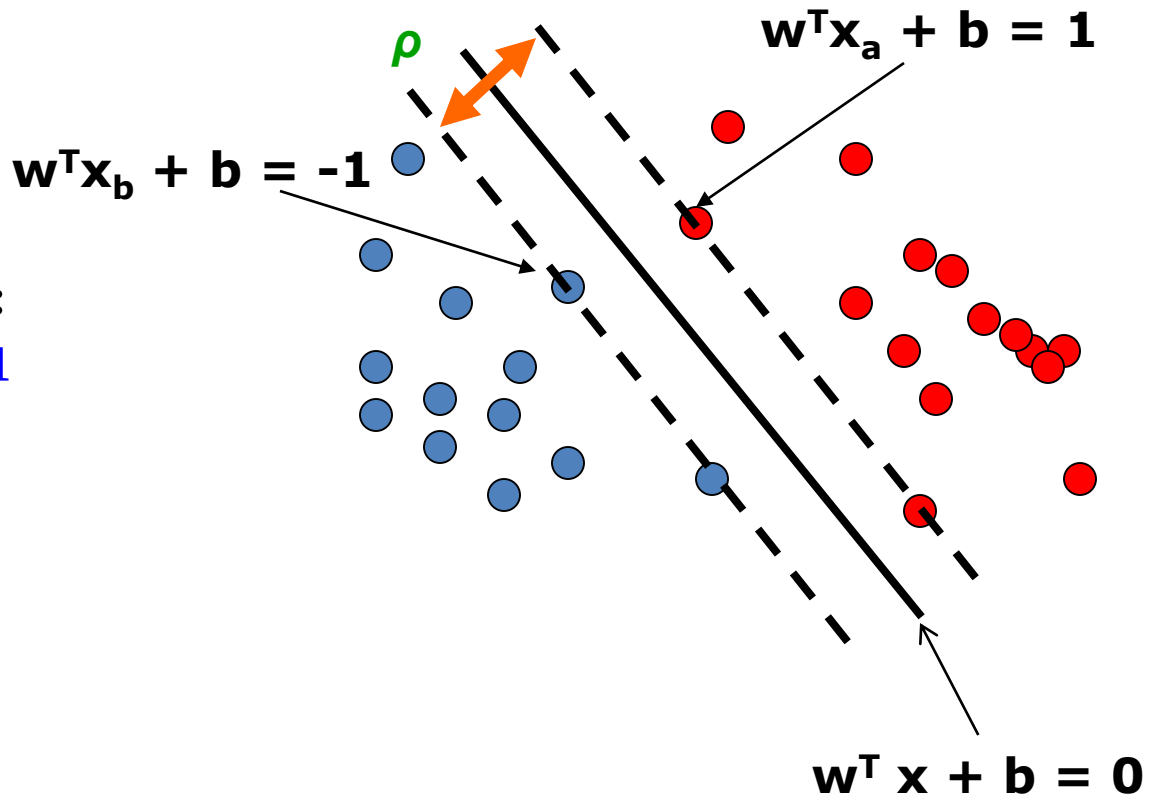
月	周	星期日		星期一		星期二		星期三		星期四		星期五		星期六	
三	一	1	开学注册	2	上课	3		4		5		6		7	
	二	8	妇女节	9		10		11		12		13		14	
	三	15		16		17		18		19		20		21	
	四	22		23		24		25		26		27		28	
四	五	29		30		31		1		2		3		4	
	六	5	清明节	6		7		8		9		10		11	
	七	12		13		14		15		16		17		18	
	八	19		20		21		22		23		24		25	
	九	26		27		28		29		30		1	劳动节	2	
五	十	3		4	青年节	5		6		7		8		9	
	十一	10		11		12		13		14		15		16	
	十二	17		18		19		20		21		22		23	
	十三	24		25		26		27		28		29		30	
六	十四	31		1		2		3		4		5		6	
	十五	7		8		9		10		11		12		13	
	十六	14		15		16		17		18		19		20	端午节

考试

答疑

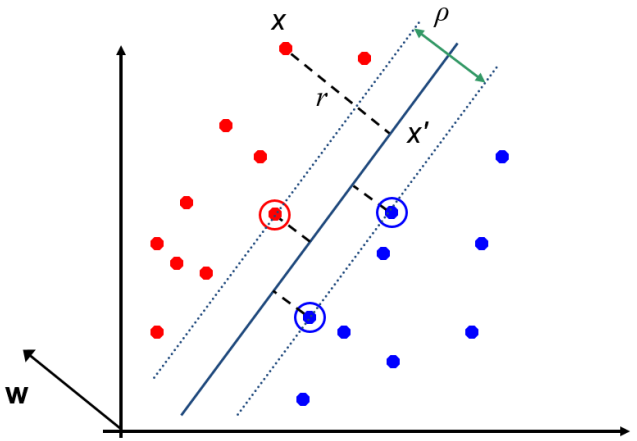
回顾：Linear Support Vector Machine (SVM)

- **Hyperplane**
  $\mathbf{w}^T \mathbf{x} + b = 0$
- **Extra scale constraint:**
  $\min_{i=1,\ldots,n} |\mathbf{w}^T \mathbf{x}_i + b| = 1$
- **This implies:**
  $\mathbf{w}^T(\mathbf{x}_a - \mathbf{x}_b) = 2$
  $\rho = \|\mathbf{x}_a - \mathbf{x}_b\|_2 = 2/\|\mathbf{w}\|_2$



分类器的几何间隔: 中间空白带的最大宽度，
该空白带可以用于将两类支持向量分开

回顾：SVM



- **线性SVM**

Find **w** and *b* such that 通过拉格朗日对偶问题求解
$\mathbf{\Phi(w)} = \frac{1}{2}\mathbf{w}^T\mathbf{w}$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1$

- **SVM用于非线性问题**
  - Soft Margin

Find **w** and *b* such that 原优化问题进行松弛
$\mathbf{\Phi(w)} = \frac{1}{2}\mathbf{w}^T\mathbf{w} + C\Sigma\xi_i$ is minimized and for all $\{(\mathbf{x_i}, y_i)\}$
$y_i(\mathbf{w}^T\mathbf{x_i} + b) \geq 1 - \xi_i$ and $\xi_i \geq 0$ for all i

  - Non-linear SVMs

$$K(\vec{x}_i, \vec{x}_j) = \vec{x}_i^{\,T}\vec{x}_j$$

$$K(\vec{x}_i, \vec{x}_j) = \Phi(\vec{x}_i)^T\Phi(\vec{x}_j)$$

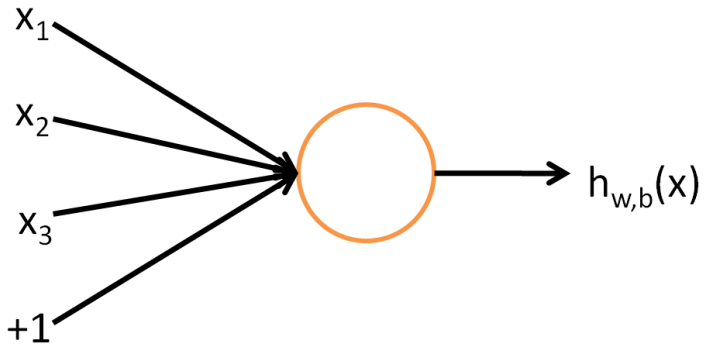
$$f(\vec{x}) = sign\left(\sum_i \alpha_i y_i K(\vec{x}_i, \vec{x}) + b\right)$$

核技巧使高维特征空间中内积计算简单化

$$K(\vec{x}, \vec{z}) = (1 + \vec{x}\cdot\vec{z})^d$$

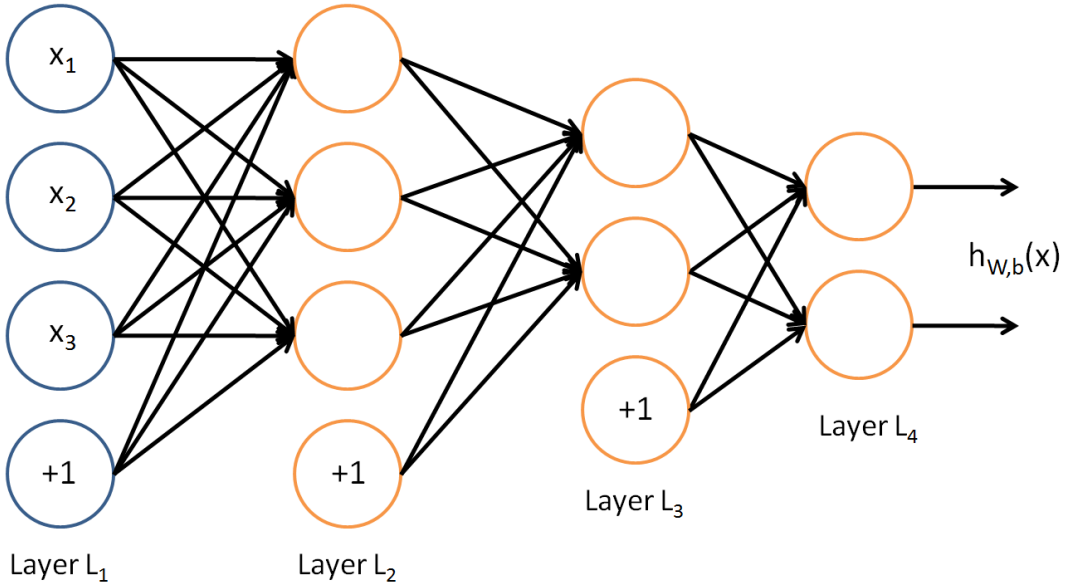
$$K(\vec{x}, \vec{z}) = e^{-(\vec{x}-\vec{z})/(2\sigma^2)}$$

回顾：人工神经网络



- 神经元
  - 输入、输出、激励函数
- 神经网络
  - 训练：获取各层的权重参数$W^{(l)}$和偏置$b^{(l)}$
  - 误差反向传播算法 backpropagation algorithm

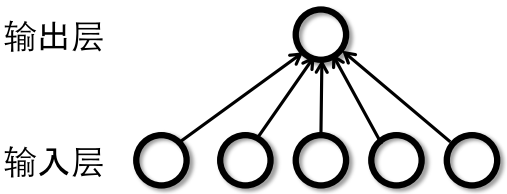
$$z^{(l+1)} = W^{(l)} a^{(l)} + b^{(l)}$$
$$a^{(l+1)} = f(z^{(l+1)})$$



回顾：机器学习技术的两次浪潮

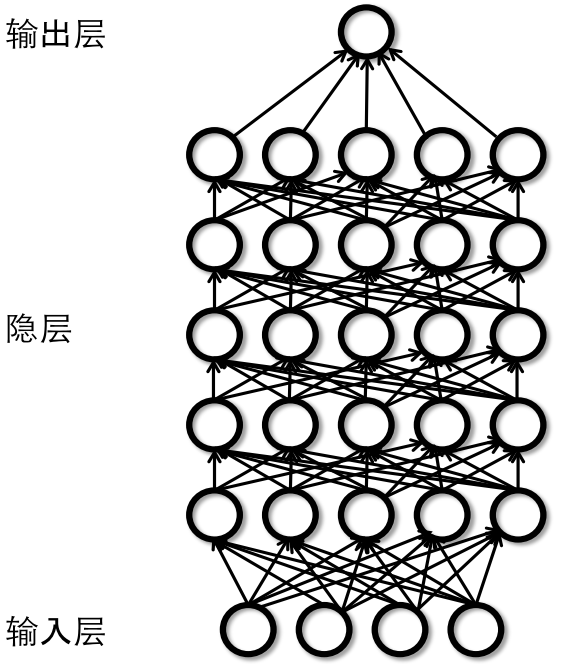
deep learning采用了神经网络相似的分层结构，系统由包括输入层、隐层（多层）、输出层组成的多层网络。传统神经网络中，采用的是back propagation的方式进行，然而对于一个deep network（7层以上），残差传播到最前面的层已经变得太小，出现所谓的gradient diffusion（梯度扩散）。

←问题的出现，BP不再合适
需要新的训练方法



不含隐层的浅层学习模型

人的视觉系统的信息处理是分级的。从低级的边缘特征，到形状或者目标的部分等，再到更高层，整个目标、目标的行为等



含多个隐层的深度学习模型

1990 第一次浪潮
浅层学习

2010 第二次浪潮
深度学习

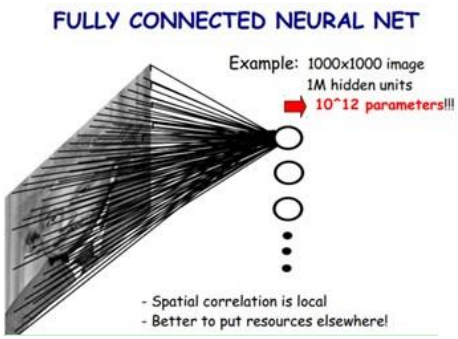
Source 《深度学习和多媒体搜索》，余凯，百度技术副总监， CCF 2013.09

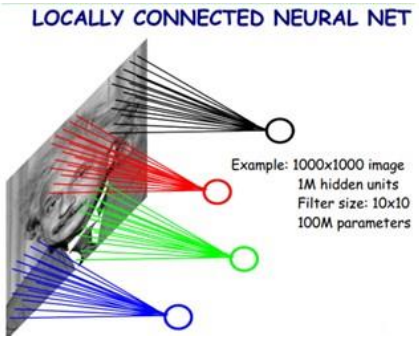
回顾：深度学习

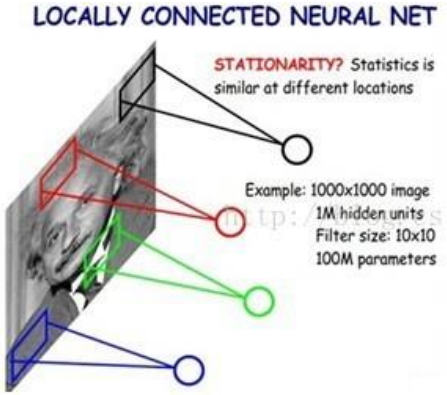
- **浅层学习是机器学习的第一次浪潮**
  - Back Propagation算法
- **深度学习是机器学习的第二次浪潮**
  - 强调了模型结构的深度
  - 突出了特征学习的重要性
- **常见的深度学习模型/算法**
  - 受限波尔兹曼机（Restricted Boltzmann Machine, RBN）
  - 深度置信网络（Deep Belief Networks, DBN）
  - 卷积神经网络（Convolutional Neural Network）
  - 堆栈式自动编码器（Stacked Auto-encoders）

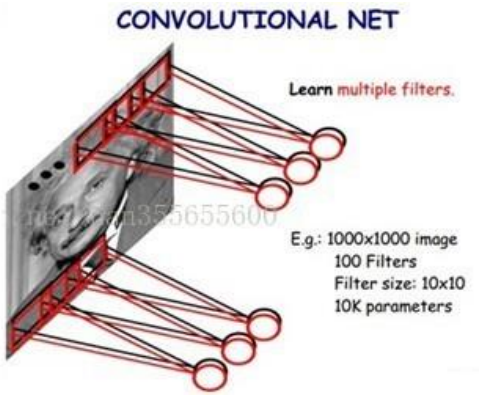
回顾： Convolutional Neural Networks (CNN)

- **Convolution Layers 卷积层**
- **Sub-sampling Layers 子采样层**
- **Local Receptive Rields 局部感受野**
- **Shared Weights 权值共享**

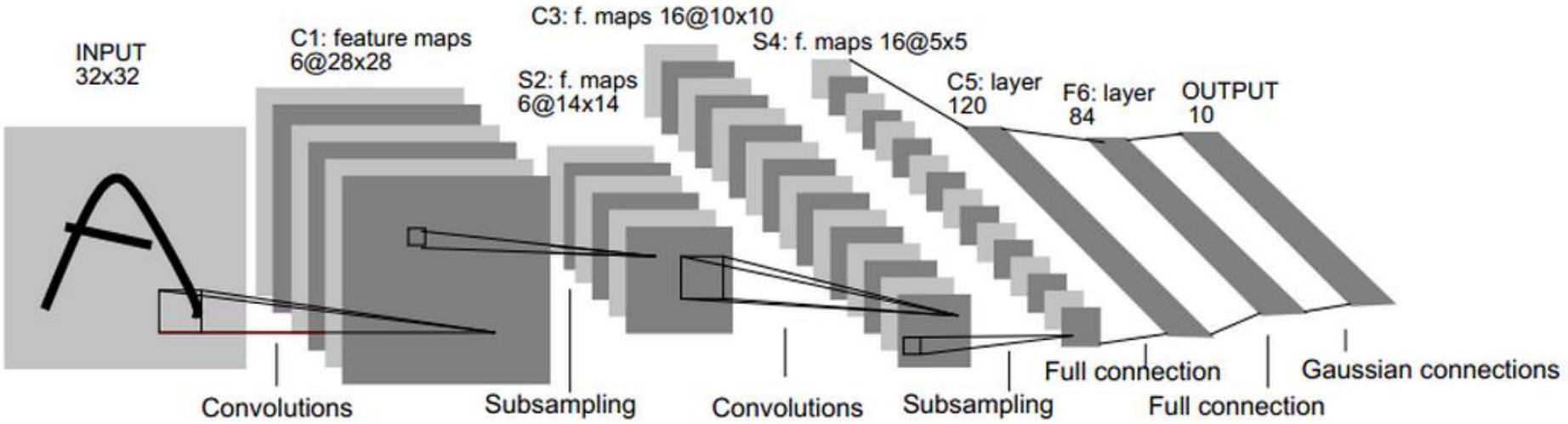








Convolutional Neural Networks (CNN)
LeNet-5文字识别系统的卷积神经网络



- C1层是一个卷积层，由6个特征图Feature Map构成（每个特征图对应一种卷积核）。特征图中每个神经元与输入中5*5的邻域相连。
- S2层是一个下采样层，有6个14*14的特征图。特征图中的每个单元与C1中的2*2邻域相连接。
- C3层也是一个卷积层，它同样通过5x5的卷积核去卷积层S2，然后得到的特征map就只有10x10个神经元，但是它有16种不同的卷积核，所以就存在16个特征map了。
- S4层是一个下采样层，由16个5*5大小的特征图构成。每个单元与C3中2*2邻域相连接。
- C5层是一个卷积层，有120个特征图。
- F6层计算输入向量和权重向量之间的点积，再加上一个偏置。
- 输出层由欧式径向基函数（Euclidean Radial Basis Function）单元组成，每个单元代表一个类别（如果要识别0-9数字的话，需要10个节点）。

Gradient Based Learning Applied to Document Recognition, Yann LeCun等, 1998

课程内容

- 第1章 绪论
- 第2章 布尔检索及倒排索引
- 第3章 词项词典和倒排记录表
- 第4章 索引构建和索引压缩
- 第5章 向量模型及检索系统
- 第6章 检索的评价
- 第7章 相关反馈和查询扩展
- 第8章 概率模型
- 第9章 基于语言建模的检索模型
- 第10章 文本分类
- 第11章 文本聚类
- 第12章 Web搜索
- 第13章 多媒体信息检索
- 第14章 其他应用简介

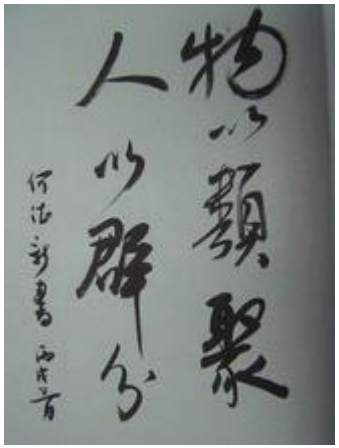
本讲内容：文本聚类

- **聚类概述**
  - 什么是聚类？在IR中如何用聚类？聚类的几个术语
- **K-均值聚类算法**
  - K-均值聚类中的基本准则
  - K-均值算法中簇的个数
- **聚类评价**
  - purity、NMI（Normalized Mutual Information，)、RI（Rand Index)、F measure）
- **基于模型的聚类**
- **层次聚类简介**
  - 层次聚类的簇相似度计算
  - 四种HAC算法：单连接、全连接 、组平均、质心法

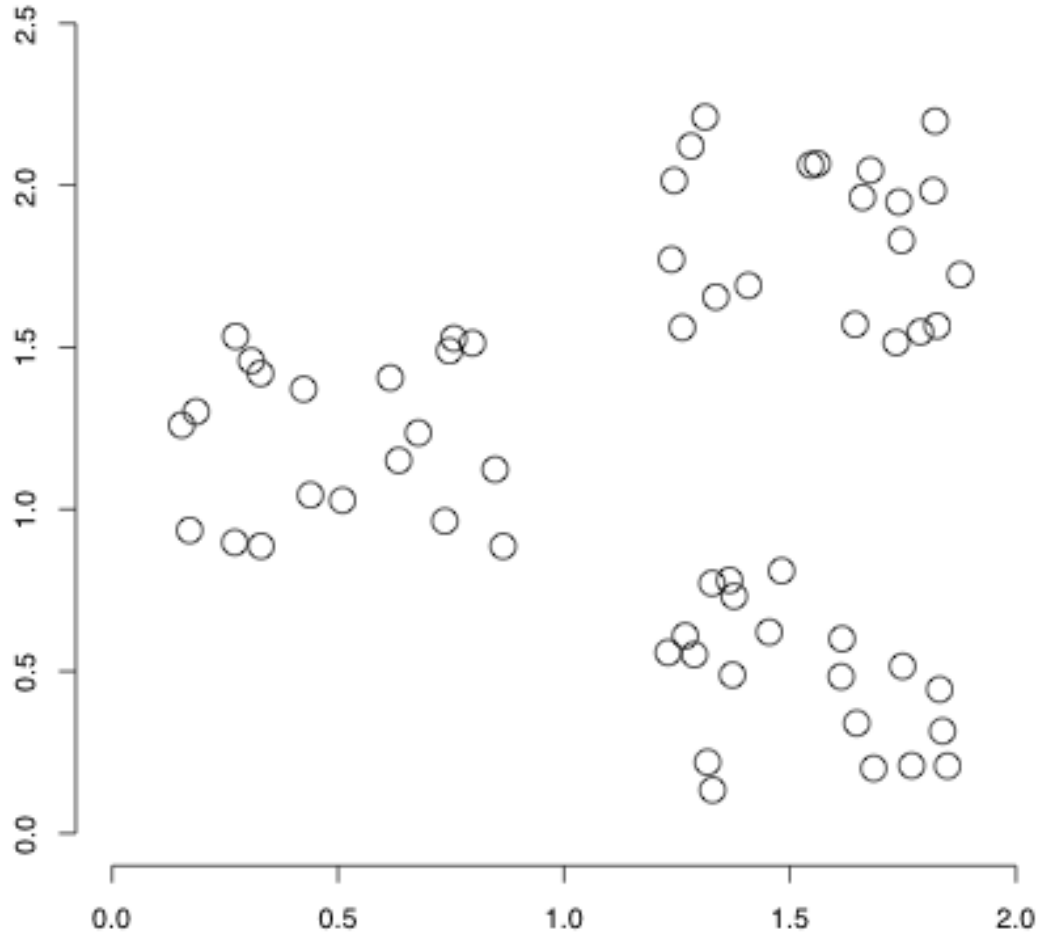
聚类(Clustering)的定义

- **(文档)聚类是将一系列文档按照相似性聚团成子集或者簇(cluster)的过程**
- **簇内文档之间应该彼此相似**
- **簇间文档之间相似度不大**

- **聚类是一种最常见的无监督学习(unsupervised learning)方法**
- **无监督意味着没有已标注好的数据集**



一个具有清晰簇结构的数据集



•聚类算法的一个关键输入是距离计算方法。图中，计算距离时采用的是二维平面上的距离计算方法。基于这种距离计算方法在图中得出了三个不同的簇。在文档聚类当中，距离计算方法往往采用欧氏距离。

•不同的距离计算方法会导致不同的聚类效果。因此，距离的计算方法是影响聚类结果的一个重要因素。

分类 vs. 聚类

- 分类：有监督的学习
- 聚类：无监督的学习
- 分类：类别事先人工定义好，并且是学习算法的输入的一部分
- 聚类：簇在没有人工输入的情况下从数据中推理而得
  - 但是，很多因素会影响聚类的输出结果：簇的个数、相似度计算方法、文档的表示方式，等等

乍看起来，聚类和分类的区别并不大，两种任务都会将文档分到不同的组中。然而，这两个问题之间存在着**本质的差异**。分类是监督学习的一种形式，其目标是对人类赋予数据的类别差异进行学习或复制。而在以聚类为重要代表的无监督学习当中，并没有这样的人来对类别的差异进行引导。

聚类假设

- 聚类假设：在考虑文档和信息需求之间的相关性时，同一簇中的文档表现互相类似。
- 聚类在IR中的所有应用都直接或间接基于上述聚类假设
- **Van Rijsbergen的原始定义: “closely associated documents tend to be relevant to the same requests”**（彼此密切关联的文档和同一信息需求相关）

聚类假设所表达的是，如果簇中某篇文档和查询需求相关，那么同一簇中的其他文档也和查询需求相关。这是因为聚类算法将那些共有很多词项的文档聚在一起。聚类假设实质上就是第14 章的邻近假设。两种情况下，我们都认为内容相似的文档在相关性上的表现也相似。

C. J. van Rijsbergen

C. J. "Keith" van Rijsbergen (Cornelis Joost van Rijsbergen) (born 1943) is a professor of computer science and the leader of the Glasgow Information Retrieval Group based at the University of Glasgow. He is one of the founders of modern Information Retrieval and the author of the seminal monograph Information Retrieval and of the textbook The Geometry of Information Retrieval.

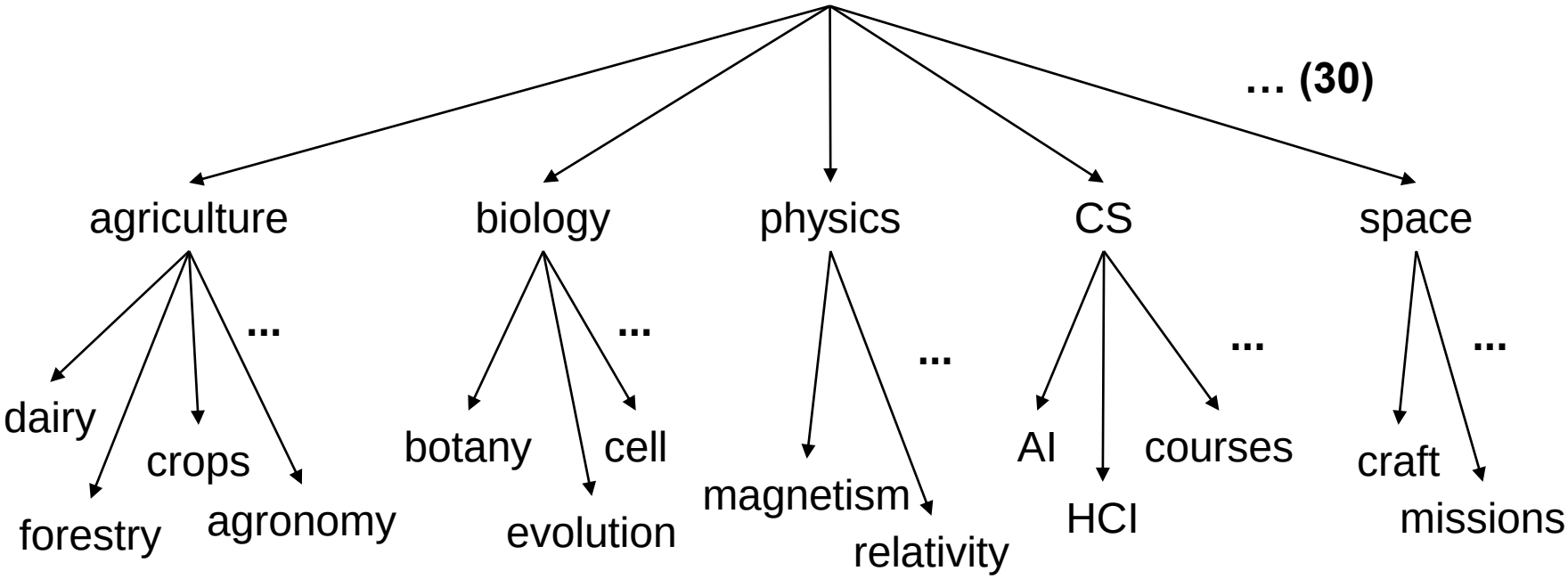
In 2003 he was inducted as a Fellow of the Association for Computing Machinery. In 2004 he was awarded the Tony Kent Strix award. In 2006, he was awarded the Gerard Salton Award for Quantum haystacks.

http://en.wikipedia.org/wiki/C._J._van_Rijsbergen
http://www.dcs.gla.ac.uk/~keith/

聚类在IR中的应用

应　　用	聚类对象	优　　点
搜索结果聚类	搜索结果	提供面向用户的更有效的展示
“分散—集中”界面	文档集和文档子集	提供了另一种用户界面，即不需要人工输入关键词的搜索界面
文档集聚类	文档集	提供了一种面向探索式浏览的有效的信息展示方法
基于语言建模的IR文档集	文档集	提高了正确率和/或召回率
基于聚类的检索	文档集	加快了搜索的速度

Yahoo! 目录式检索的效果



Hierarchy *isn't* clustering but *is* the kind of output you want from clustering

百度搜索“jagular”返回的结果

https://www.baidu.com/s?wd=jagular&rsv_spt=1&issp=1&f=8&rsv_bp=1&rs

jaguar_百度搜索

转换 选择

百度 jagular 百度一下 添加百度到桌面，搜索更便捷！

网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»

百度为您找到相关结果约410,000个 搜索工具

以下为您显示“jaguar”的搜索结果。仍然搜索：jagular

捷豹(Jaguar)中国官方网站 预约试驾全部车型 推广链接

捷豹致力于打造高性能豪华轿车,以诱人设计,卓越性能和精湛工艺著称.立即登录官网了解更多捷豹屡获殊荣的车型设计和造车之道!

XJL 80周年典藏版 XF 80周年典藏版 捷豹XE 捷豹F-TYPE 预约试

www.jaguar.com.cn 2015-05 V3 - 评价

jaguar_百度翻译

jaguar
英 ['dʒægjʊə(r)]
美 ['dʒæg,wɑr, 'dʒægju,ɑr]

n. <动>（中、南美洲的）美洲虎
名词复数:jaguars
[例句] Jaguar financial a canadian investment bank smells complacency.
捷豹财政这家加拿大投资银行则洋洋得意。
更多示例用法>>

fanyi.baidu.com

JAGUAR_百度百科

捷豹（Jaguar）：是英国的一家豪华汽车生产商，车标为一只正在跳跃前扑的“美洲豹”形象，矫健勇猛，形神兼备，具...
品牌文化 捷豹创始人 捷豹理念 在售车型 经典车型 更多>>
查看“JAGUAR”全部2个含义>>
baike.baidu.com/ 2015-05-08

【捷豹】报价|图片|参数配置|口碑-汽车之家

类型	车系				
轿车	捷豹XJ	捷豹XF	捷豹XE	捷豹S-Type	全部(5款)
跑车	捷豹F-TYPE	捷豹XK			

查看全部7款汽车>>
www.autohome.com.cn 2015-05-14

捷豹客服电话

400-820-8055

百度搜索 有V有保障

其他人还搜 展开


尹喜地
3000万购布加迪威龙


萨林
就是今天Saleen的前身


史上最牛跑车
王健仿造兰博基尼跑车


美洲虎
非洲北豹猫科全能冠军

相关汽车 汽车类 展开


edonis
基于布加迪打造的超跑


lincoln
福特汽车厂的传世佳作


青年莲花汽车
综合性汽车工业集团


奥迪a9
豪华运动系轿车


积架
英国豪华的名牌产品


派拉蒙掠夺者
世上最不可阻挡的车辆


迈巴赫exelero
运动跑车极品


特纳蒙塔纳
西班牙战斗机设计超跑

车迷进阶小知识

关注点	关注热度
jaguar考虑进军小型车市场	★★★★★
jaguar将投产三款全新suv	★★★★★
jaguar属于哪个国家产	★★★★★

谷歌搜索“jagular”返回的结果

https://www.baidu.com/s?wd=jagular&rsv_spt=1&issp=1&f=8&rsv_bp=1&rs

jaguar_百度搜索

转换 选择

Bai du 百度 jagular 百度一下 添加百度到桌面，搜索更便捷！

网页 新闻 贴吧 知道 音乐 图片 视频 地图 文库 更多»

https://www.google.com.hk/search?newwindow=1&safe=strict&hl=zh-CN&gl=c

jaguar - Google 搜索

转换 选择

+你 搜索 图片 地图 Play YouTube 新闻 Gmail 更多

Google jaguar

网页 图片 地图 新闻 视频 更多 搜索工具

找到约 171,000,000 条结果 （用时 0.31 秒）

jaguar.com.cn - 捷豹(Jaguar)中国官方网站

广告 www.jaguar.com.cn/

捷豹XF,XJ,XE,F-TYPE,F-PACE 全系重磅升级,敬邀尊享试驾!

相关搜索： jaguar汽车价格 jaguar 价格

捷豹汽车_英国豪华汽车品牌_跑车品牌-Jaguar中国官方网站

www.jaguar.com.cn/

历经时代变更,捷豹(jaguar)始终致力于为冰冷的机械赋注灵魂,创造具有无可复制的极致奢华的美洲豹汽车,以诱人设计、卓越性能、精湛工艺和创新精神著称.

捷豹XE - 捷豹F-TYPE - 全部车型 - 捷豹XJ

Jaguar

www.jaguar.com/ 翻译此页

Official worldwide web site of Jaguar Cars. Directs users to pages tailored to country-specific markets and model-specific websites.

jaguar的图片搜索结果 举报图片

有关“jaguar”的更多图片

捷豹

汽车制造商

捷豹汽车有限公司是英国的一家豪华汽车生产商，总部起初座落于英格兰考文垂的布朗兰，后迁至考文垂的惠特利。 维基百科

首席执行官： 施韦德

创立于： 1922 年 9 月 4 日，英国布莱克浦

JAGUAR

反馈

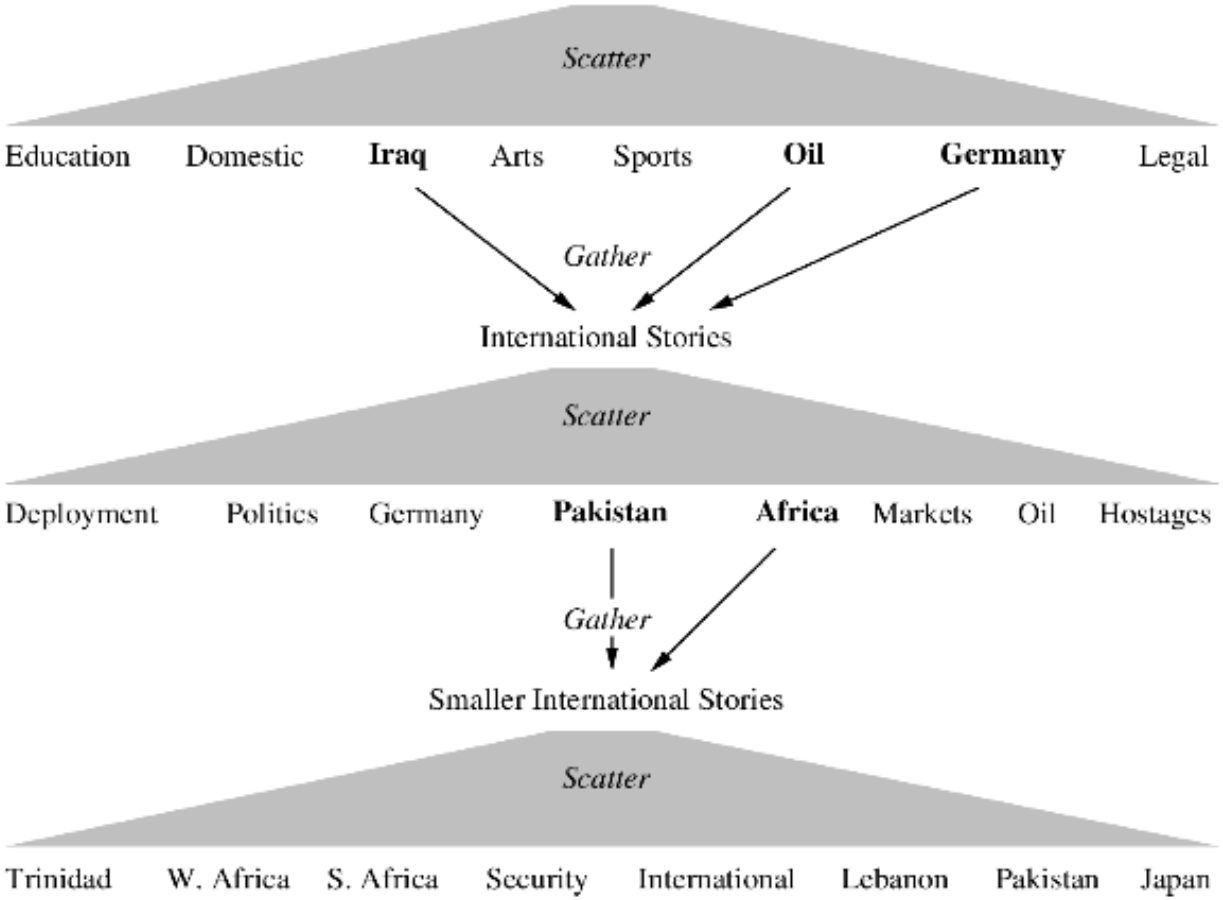
搜索结果的聚类：更好地浏览

返回结果中前面的文档并没有覆盖jaguar 作为动物的那个词义。但是用户很容易通过在Clustered Results 面板中点击Cat 簇得到该类结果

如果对结果有分类....

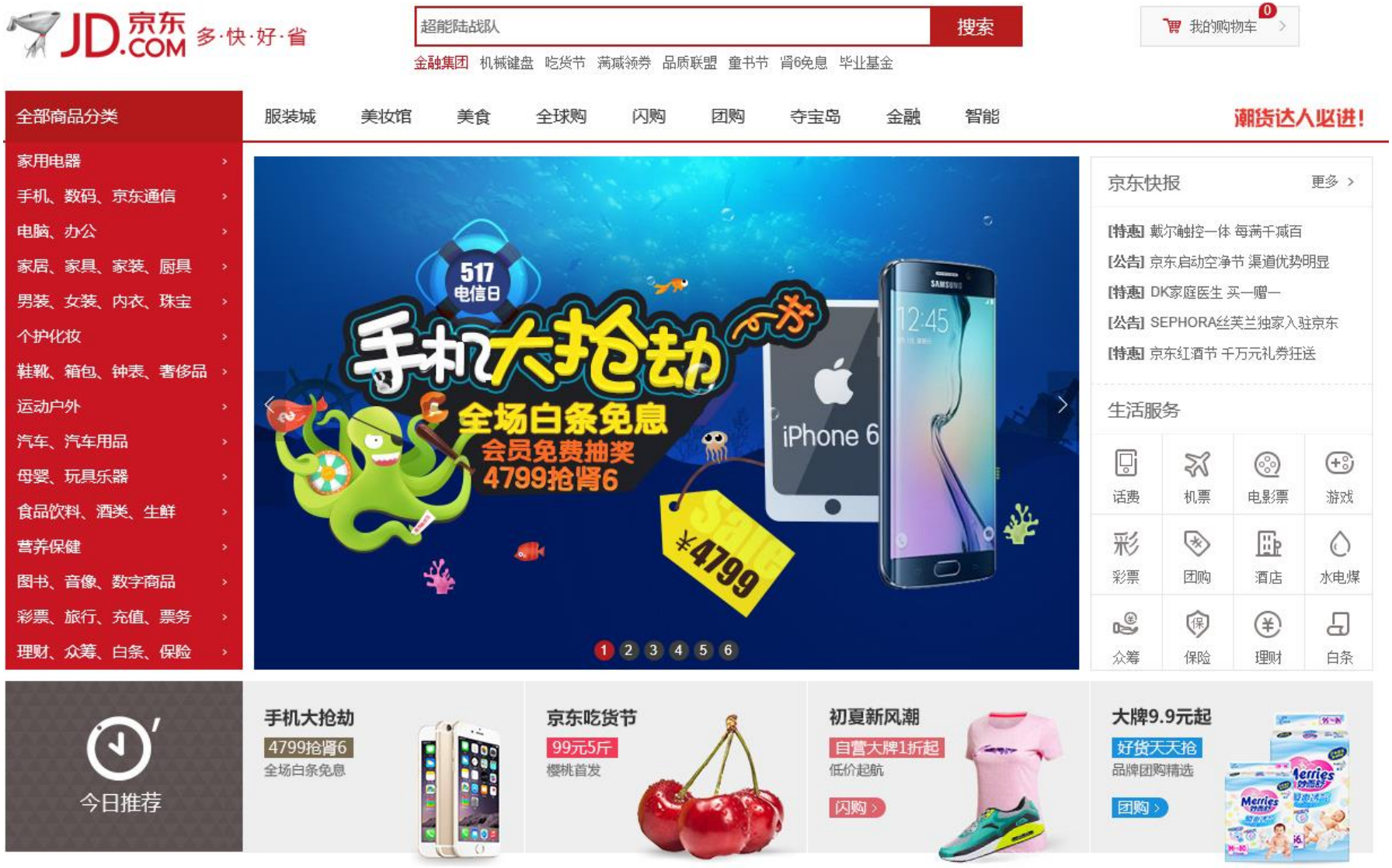
朱自清《荷塘月色》

分散-集中(Scatter-Gather)

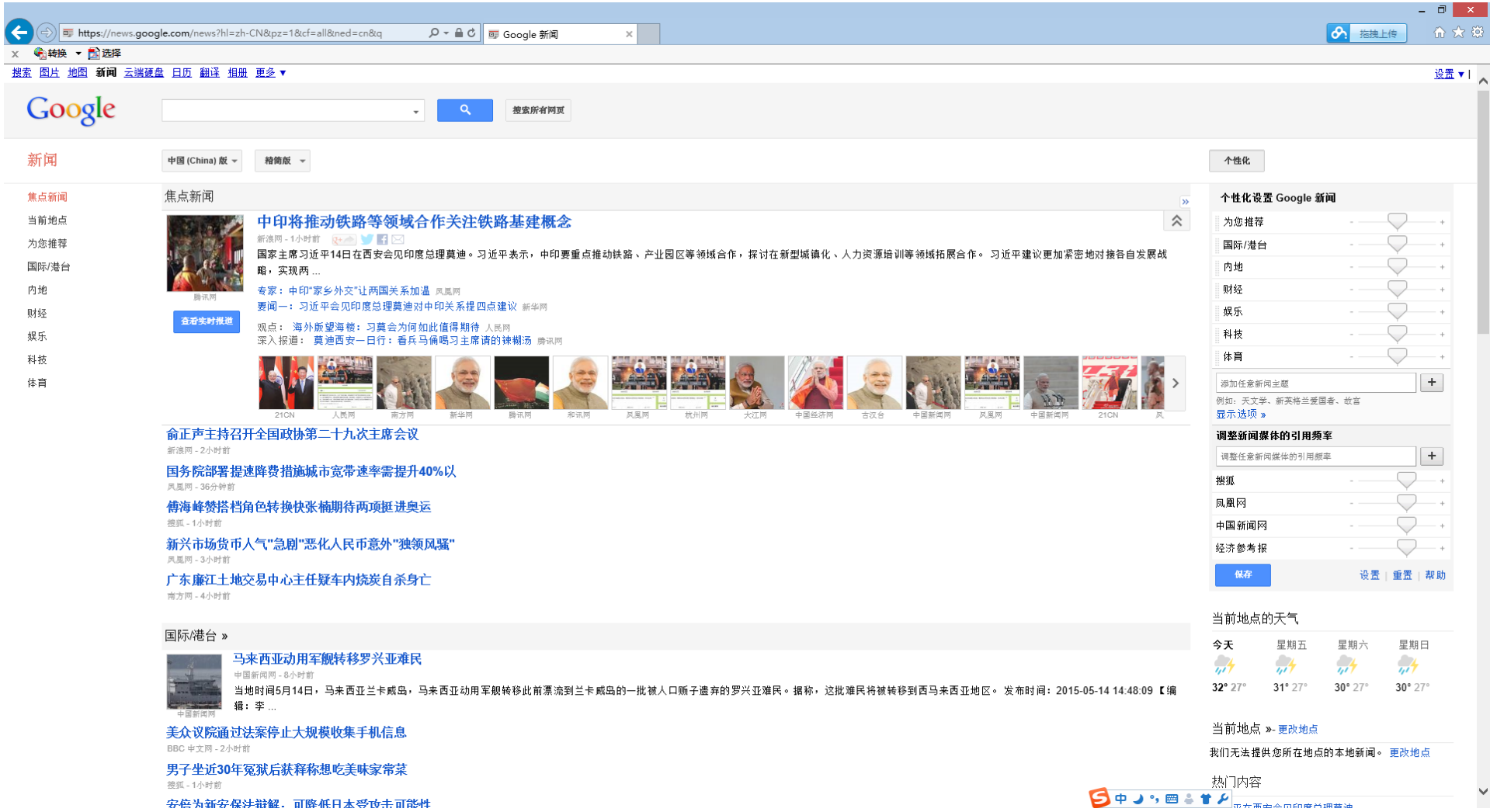


一个New York Times 的新闻报道集聚类（“ 分散” ）成8 个簇（顶行）。用户手工将其中三个簇“ 集中” 成一个小的文档集International Stories，然后再次进行分散操作。重复上述过程直至找到包含相关文档的簇（比如Trinidad）为止

全局浏览的例子: 京东


JD.京东.COM 多·快·好·省
超能陆战队
搜索
金融集团 机械键盘 吃货节 满减领券 品质联盟 童书节 肾6免息 毕业基金
我的购物车
全部商品分类
服装城 美妆馆 美食 全球购 闪购 团购 夺宝岛 金融 智能
潮货达人必进!
家用电器
手机、数码、京东通信
电脑、办公
家居、家具、家装、厨具
男装、女装、内衣、珠宝
个护化妆
鞋靴、箱包、钟表、奢侈品
运动户外
汽车、汽车用品
母婴、玩具乐器
食品饮料、酒类、生鲜
营养保健
图书、音像、数字商品
彩票、旅行、充值、票务
理财、众筹、白条、保险
517 电信日
手机大抢劫
全场白条免息
会员免费抽奖
4799抢肾6
iPhone 6
¥4799
京东快报 更多
[特惠] 戴尔触控一体 每满千减百
[公告] 京东启动空净节 渠道优势明显
[特惠] DK家庭医生 买一赠一
[公告] SEPHORA丝芙兰独家入驻京东
[特惠] 京东红酒节 千万元礼券狂送
生活服务
话费 机票 电影票 游戏
彩票 团购 酒店 水电煤
众筹 保险 理财 白条
今日推荐
手机大抢劫
4799抢肾6
全场白条免息
京东吃货节
99元5斤
樱桃首发
初夏新风潮
自营大牌1折起
低价起航
闪购
大牌9.9元起
好货天天抢
品牌团购精选
团购

全局浏览的例子: Google News



文档聚类用于提高召回率

- 可以实现将文档集中的文档进行聚类
- 当文档d和查询匹配时，也返回包含d的簇所包含的其它文档
- 我们希望通过上述做法，在输入查询“car“时，也能够返回包含 “automobile”的文档
- 由于聚类算法会把包含 “car”的文档和包含“automobile”的文档聚在一起
- 两种文档都包含诸如 “parts”、 “dealer”、“mercedes”和“road trip”之类的词语

聚类要解决的基本问题

- **Representation for clustering**
  - Document representation
    - Vector space? Normalization?
      - Centroids aren't length normalized
  - Need a notion of similarity/distance
- **How many clusters?**
  - Fixed a priori?
  - Completely data driven?
    - Avoid "trivial" clusters - too large or small
      - If a cluster's too large, then for navigation purposes you've wasted an extra user click without whittling down the set of documents much.

聚类的形式化描述

- **硬扁平聚类的目标可以定义如下：**
  - 给定
  - (i) 一系列文档$D = \{d_1, ..., d_N\}$，
  - *(ii)* 期望的簇数目K，
  - *(iii)*用于评估聚类质量的目标函数（objective function），
  - 计算一个分配映射$\gamma : D \rightarrow \{1, ..., K\}$，该分配下的目标函数值极小化或者极大化。大部分情况下，我们要求γ 是一个满射，也就是说，K 个簇中的每一个都不为空。
- **目标函数通常基于文档的相似度或者距离来定义。下面我们将看到，*K*-均值算法的目标是最小化文档和其所在簇的质心的平均距离。**

扁平聚类 vs. 层次聚类

- **扁平算法**
  - 通过一开始将全部或部分文档随机划分为不同的组
  - 通过迭代方式不断修正
  - 代表算法：K-均值聚类算法
- **层次算法**
  - 构建具有层次结构的簇
  - 自底向上(Bottom-up)的算法称为凝聚式(agglomerative)算法
  - 自顶向下(Top-down)的算法称为分裂式(divisive)算法

硬聚类 vs. 软聚类

- **硬聚类(Hard clustering): 每篇文档仅仅属于一个簇**
  - **很普遍并且相对容易实现**
- **软聚类(Soft clustering): 一篇文档可以属于多个簇**
  - **对于诸如浏览目录之类的应用来说很有意义**
  - **比如，将 胶底运动鞋 (sneakers) 放到两个簇中：**
    - **体育服装(sports apparel)**
    - **鞋类(shoes)**
  - **只有通过软聚类才能做到这一点**

扁平算法

- 扁平算法将N篇文档划分成K个簇
- 给定一个文档集合及聚类结果簇的个数K
- 寻找一个划分将这个文档集合分成K个簇，该结果满足某个最优划分准则
- 全局优化：穷举所有的划分结果，从中选择最优的那个划分结果
  - 无法处理
- 高效的启发式方法: K-均值聚类算法

小结：聚类

- **(文档)聚类是将一系列文档按照相似性聚团成子集或者簇(cluster)的过程。是一种最常见的无监督学习(unsupervised learning)方法**
- **聚类假设：在考虑文档和信息需求之间的相关性时，同一簇中的文档表现互相类似。**
- **聚类要解决的基本问题**
  - Representation for clustering
  - How many clusters?
- **扁平聚类 vs. 层次聚类**
- **硬聚类 vs. 软聚类**

本讲内容：文本聚类

- **聚类概述**
  - 什么是聚类？在IR中如何用聚类？聚类的几个术语
- **K-均值聚类算法**
  - K-均值聚类中的基本准则
  - K-均值算法中簇的个数
- **聚类评价**
  - purity、NMI（Normalized Mutual Information，)、RI（Rand Index)、F measure）
- **基于模型的聚类**
- **层次聚类简介**
  - 层次聚类的簇相似度计算
  - 四种HAC算法：单连接、全连接 、组平均、质心法

K-均值聚类算法思想

- **K-均值聚类算法中的每个簇都定义为其质心向量**
- **划分准则： 使得所有文档到其所在簇的质心向量的平方和最小**
- **质心向量的定义：**

$$\vec{\mu}(\omega) = \frac{1}{|\omega|} \sum_{\vec{x} \in \omega} \vec{x}$$

  - 其中 ω 代表一个簇
- **通过下列两步来实现目标优化：**
  - 重分配(reassignment): 将每篇文档分配给离它最近的簇
  - 重计算(recomputation): 重新计算每个簇的质心向量

或许是最著名的聚类算法。算法十分简单，但是在很多情况下效果不错。是文档聚类的默认或基准算法

K-均值聚类中的文档表示

- **向量空间模型**
  - 同基于向量空间的分类一样，这里我们也采用欧氏距离的方法来计算向量之间的相关性
  - 欧氏距离与余弦相似度差不多等价(如果两个向量都基于长度归一化，那么欧氏距离和余弦相似度是等价的)
- **然而，质心向量通常都没有基于长度进行归一化**

K-均值聚类中的目标函数

- 一个衡量质心对簇中文档的代表程度的指标是**RSS**（***Residual Sum of Squares***，残差平方和），即所有向量到其质心距离的平方和：

$$\mathrm{RSS} = \sum_{k=1}^{K} \mathrm{RSS}_k \qquad \mathrm{RSS}_k = \sum_{x \in \omega_k} |\vec{x} - \vec{\mu}(\omega_k)|^2$$

- **RSS** 是**K-**均值算法的目标函数，我们的目的就是要是让这个函数取最小值。由于**N** 是固定
- 的，最小化**RSS** 也就等价于最小化平方距离，而平方距离度量的正是质心对文档的代表能力。

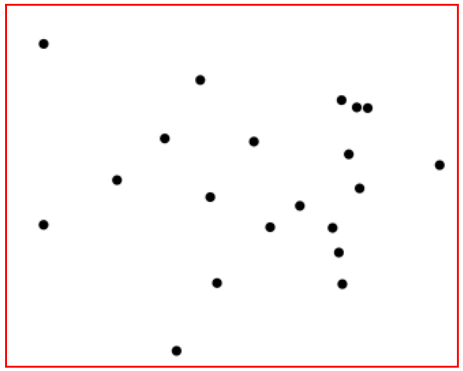
K-均值聚类算法

K-MEANS$(\{\vec{x}_1, \ldots, \vec{x}_N\}, K)$

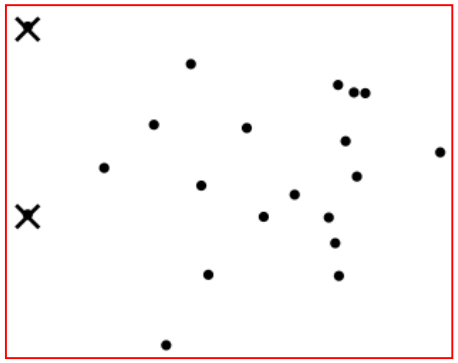
1 $(\vec{s}_1, \vec{s}_2, \ldots, \vec{s}_K) \leftarrow$ SELECTRANDOMSEEDS$(\{\vec{x}_1, \ldots, \vec{x}_N\}, K)$
2 **for** $k \leftarrow 1$ **to** K
3 **do** $\vec{\mu}_k \leftarrow \vec{s}_k$
4 **while** stopping criterion has not been met
5 **do for** $k \leftarrow 1$ **to** K
6 **do** $\omega_k \leftarrow \{\}$
7 **for** $n \leftarrow 1$ **to** N
8 **do** $j \leftarrow \arg\min_{j'} |\vec{\mu}_{j'} - \vec{x}_n|$
9 $\omega_j \leftarrow \omega_j \cup \{\vec{x}_n\}$ *(reassignment of vectors)*
10 **for** $k \leftarrow 1$ **to** K
11 **do** $\vec{\mu}_k \leftarrow \frac{1}{|\omega_k|} \sum_{\vec{x} \in \omega_k} \vec{x}$ *(recomputation of centroids)*
12 **return** $\{\vec{\mu}_1, \ldots, \vec{\mu}_K\}$

K-均值算法的stopping criterion

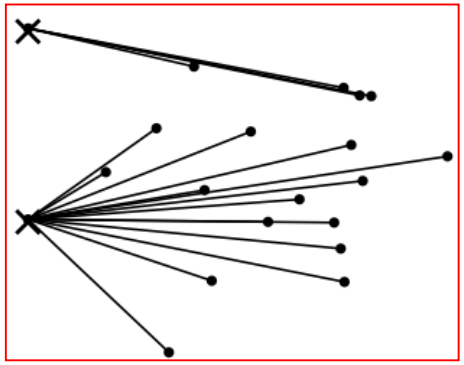
- **K-均值算法中可以采用如下终止条件。**
  - 当迭代一个固定次数 I 后停止。该条件能够限制聚类算法的运行时间，但是有些情况下，由于迭代次数不足，聚类结果的质量并不高。
  - 当文档到簇的分配结果（即划分函数γ）不再改变后停止。除了某些情况下会使算法陷入局部最优外，该停止条件通常会产生较好的聚类结果，但是运行时间不宜太久。
  - 当质心向量$\rightarrow\mu_k$ 不再改变后停止。这等价于函数γ 不再改变。



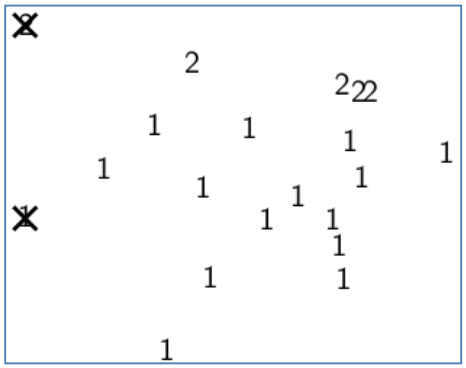
拟聚类文档



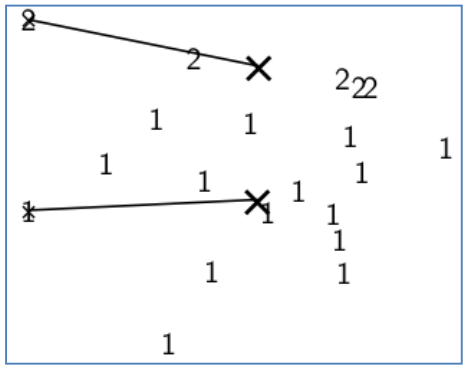
随机选择两个种子(K=2)



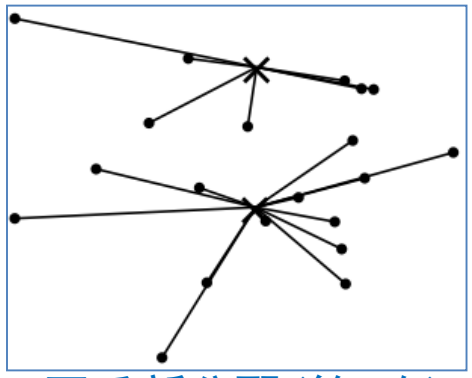
分配(第1次)



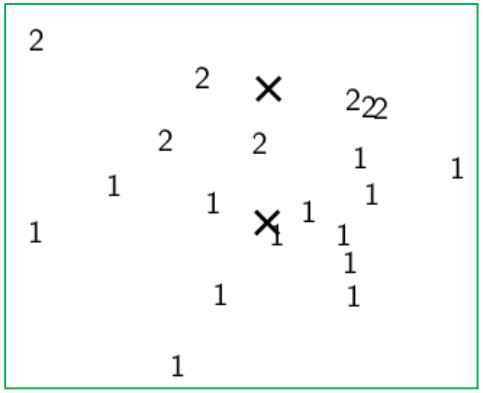
分配结果



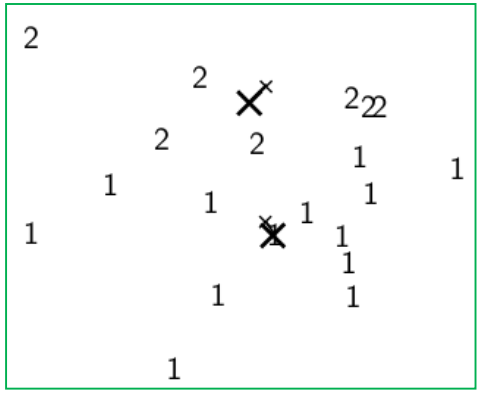
重新计算质心向量



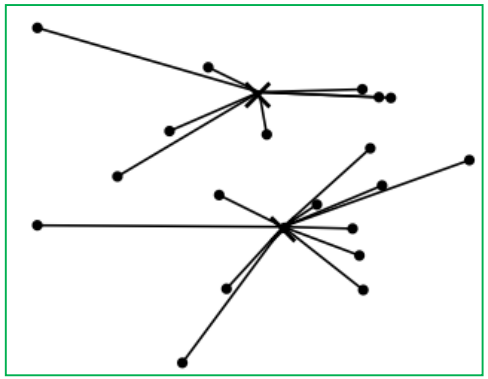
再重新分配(第2次)



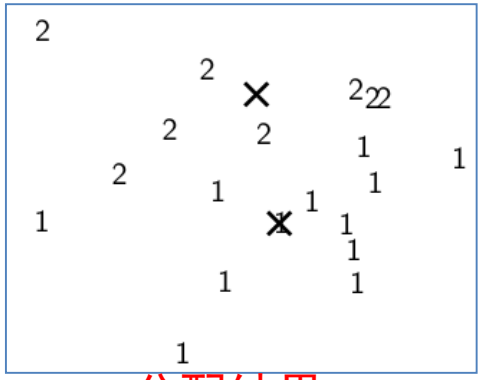
分配结果



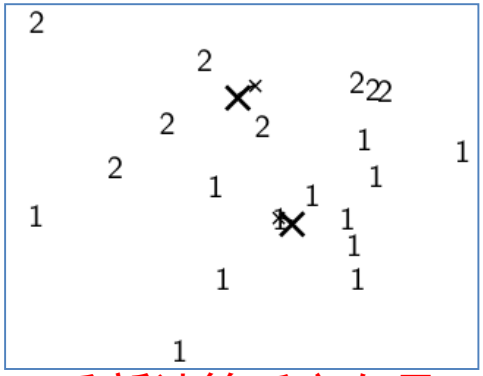
重新计算质心向量



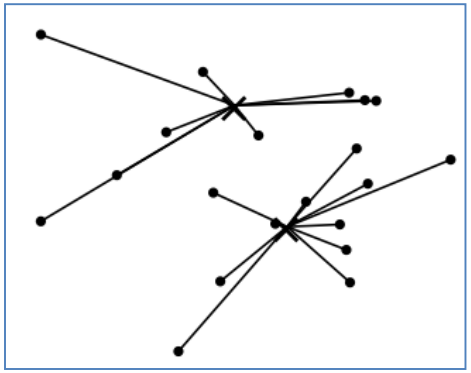
再重新分配(第3次)



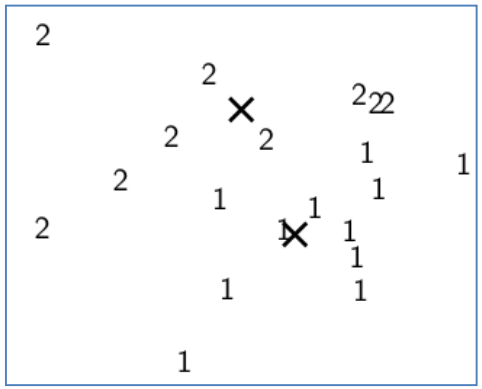
分配结果



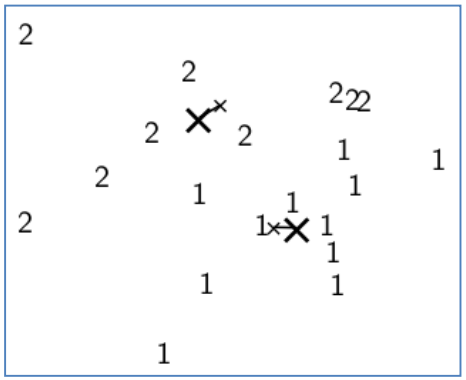
重新计算质心向量



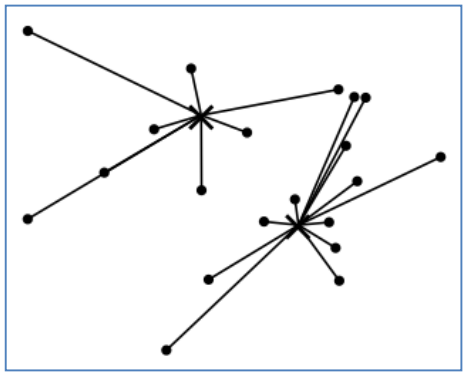
再重新分配(第4次)



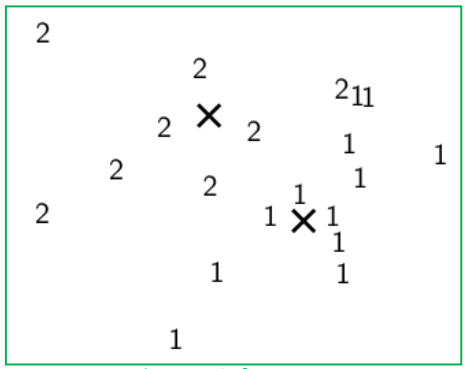
分配结果



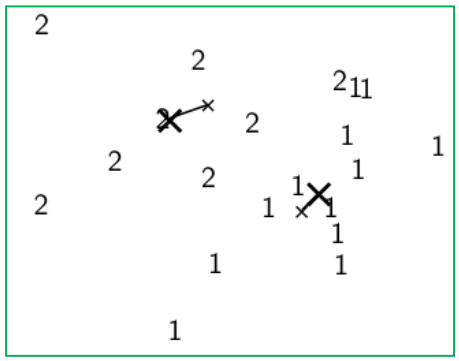
重新计算质心向量



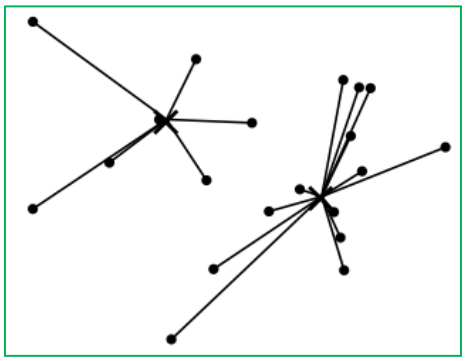
再重新分配(第5次)



分配结果

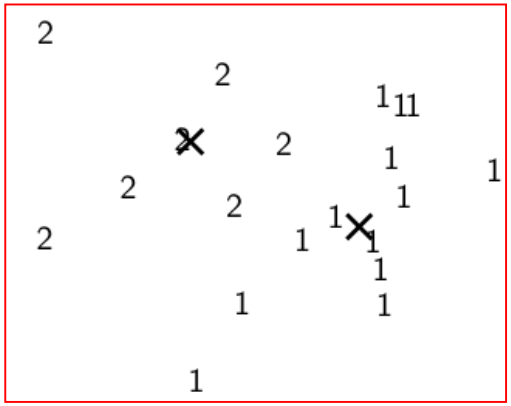


重新计算质心向量

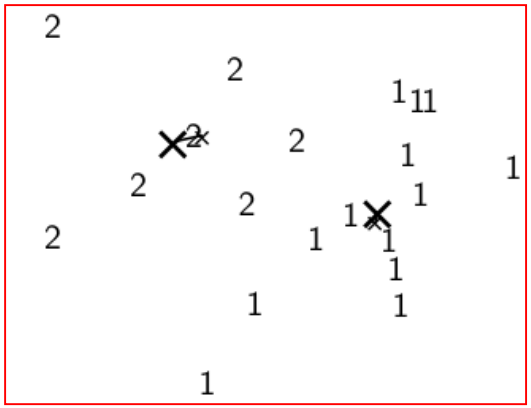


再重新分配(第6次)

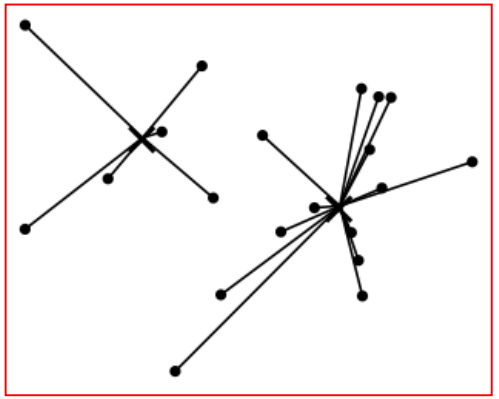
质心向量和分配结果最终收敛



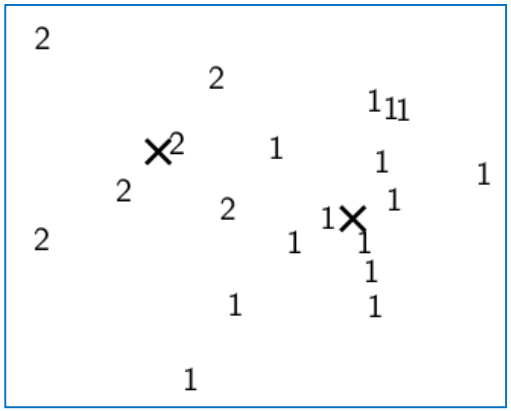
分配结果



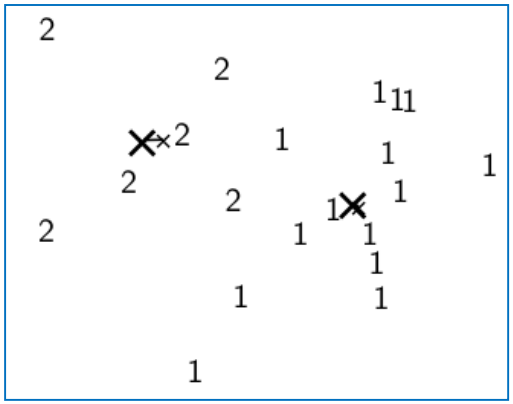
重新计算质心向量



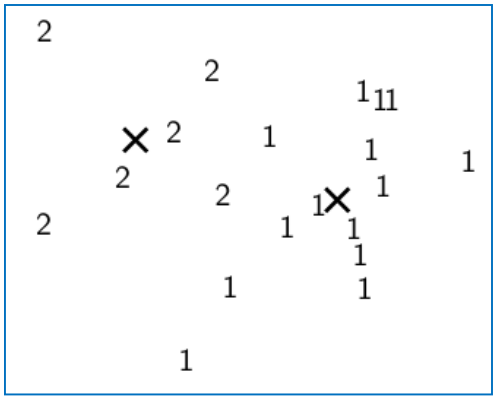
再重新分配(第7次)



分配结果



重新计算质心向量



质心向量和分配结果最终收敛

K-均值聚类算法一定会收敛: 证明

- **RSS = 所有簇上的文档向量到(最近的)质心向量的距离平方和的总和**
- **每次重新分配之后RSS会下降**
  - 这是因为每个向量都被移到离它最近的质心向量所代表的簇中
- **每次重新计算之后RSS也会下降**
  - 参见p250公式16-8、16-9、16-10
- **可能的聚类结果是有穷的**
- **因此：一定会收敛到一个固定点**

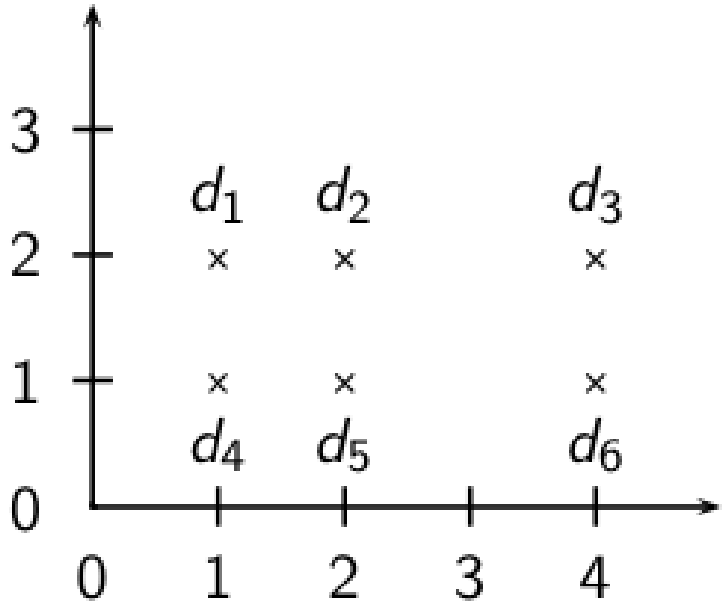
但是不知道达到收敛所需要的时间!

K-均值聚类算法的最优性

- **K-均值聚类算法的最优性**
  - 收敛并不意味着会达到全局最优的聚类结果!
  - 这是K-均值聚类算法的最大缺点之一
  - 如果种子选的不好，最终的聚类结果可能会非常糟糕

•K=2情况下的最优聚类结果是什么？
•对于任意的种子d_i 、d_j ,我们是否都会收敛于该聚类结果？

对于种子d2 和d5，K-均值算法最后收敛为{{d1, d2, d3}, {d4, d5, d6}}
对种子d2 和 d3，收敛结果为{{d1, d2,d4, d5}, {d3, d6}}，这是K=2时的全局最优值



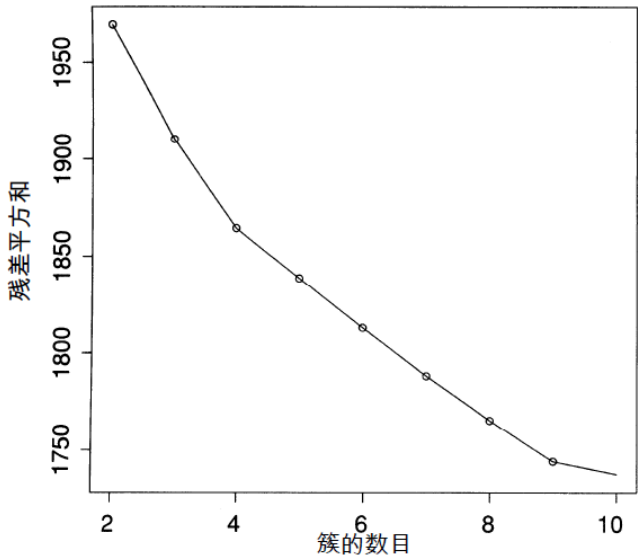
K-均值聚类算法的性能

- **K-均值聚类算法的时间复杂度**
  - 计算两个向量的距离的时间复杂度为O(M)
  - 重分配过程: O(KNM) (需要计算 KN 个文档-质心的距离)
  - 重计算过程： O(NM) (在计算质心向量时，需要累加簇内的文档向量)
  - 假定迭代次数的上界是 I
  - 整体复杂度: O(IKNM) – 线性

对于某个固定的迭代次数I，K-均值算法的时间复杂度与迭代次数、簇的数目、向量的数目及向量的维数之间都是线性关系

K-均值算法中簇的个数

- 对于大多数扁平聚类算法来说，簇的个数K 往往是算法的输入参数。一个很自然的想法显然是选择使目标函数最优的K 值，也就是使RSS 极小化的K 值。将所有可能输出K 个聚类结果的RSS 中的最小值记为$RSS_{min}(K)$，则$RSS_{min}(K)$会随着K 的增大而单调递减，当K=N 时，$RSS_{min}(K)$会取最小值0，其中N 是所有文档的数目。



拐点的结果具有一定的代表性

K-均值算法中，估计出的RSSmin 是簇数目的函数。该聚类中，总共对1203 篇Reuters-RCV1 的文档进行了聚类。RSSmin曲线分别在簇数目为4和9的两个拐点处变平坦。聚类中的文档从China、Germany、Russia 及Sports 等4 个类中选取，因此，K=4 的聚类结果接近Reuters-RCV1 中的分类

第二种确定簇数目的准则是对每个新簇给予一定的惩罚

- **基本思路:从1个簇开始 (K = 1)→不断增加簇 (= 不断增大K)→对每个新的簇增加一个惩罚项。在惩罚项和RSS之间折中,选择满足最佳折中条件的 K**
- **目标函数**
  - 给定聚类结果，定义文档的代价为其到质心向量的(平方)距离（失真率）
  - 定义全部失真率 RSS(K) 为所有文档代价的和
  - 然后：对每个簇一个惩罚项 λ
  - 于是，对于具有 K 个簇的聚类结果，总的聚类惩罚项为 Kλ
  - 定义聚类结果的所有开销为失真率和总聚类惩罚项的和：
  - RSS(K) + Kλ
  - 选择使得 (RSS(K) + Kλ) 最小的K值
  - 当然，还要考虑较好的λ值 . . .

小结： K-均值算法

- **K-均值聚类中的文档表示：向量空间模型**
- **准则：RSS = 所有簇上的文档向量到(最近的)质心向量的距离平方和的总和 最小**
- **收敛。但并不意味着会达到全局最优的聚类结果（受种子影响）**
- **K-均值聚类算法的时间复杂度：O(IKNM) – 线性**
- **K-均值算法中簇的个数：**
  - RSSmin(K)会随着K 的增大而单调递减，但拐点的结果具有一定的代表性
  - 从1个簇开始 (K = 1)→不断增加簇 (= 不断增大 K)→对每个新的簇增加一个惩罚项。

本讲内容：文本聚类

- **聚类概述**
  - 什么是聚类？在IR中如何用聚类？聚类的几个术语
- **K-均值聚类算法**
  - K-均值聚类中的基本准则
  - K-均值算法中簇的个数
- **聚类评价**
  - purity、NMI（Normalized Mutual Information，)、RI（Rand Index)、F measure）
- **基于模型的聚类**
- **层次聚类简介**
  - 层次聚类的簇相似度计算
  - 四种HAC算法：单连接、全连接 、组平均、质心法

怎样判断聚类结果的好坏?

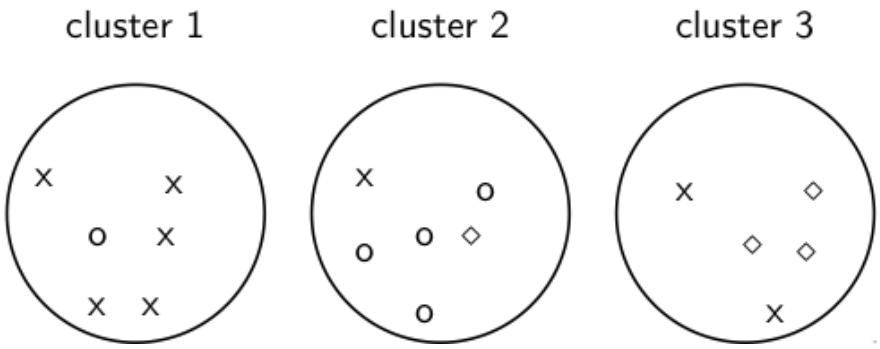
- **内部准则(Internal criteria)**
  - 将簇内高相似度（簇内文档相似）及簇间低相似度（不同簇之间的文档不相似）的目标进行形式化后得到的一个函数。一个内部准则的例子： K-均值算法的RSS值
- **但内部准则往往不能评价聚类在应用中的实际效用**
- **替代方法：外部准则(External criteria)**
  - 按照用户定义的分类结果来评价，即对一个分好类的数据集进行聚类，将聚类结果和事先的类别情况进行比照，得到最后的评价结果
  - 四种衡量聚类质量的外部准则：纯度（purity）、NMI（Normalized Mutual Information，归一化互信息)、RI（Rand Index，兰德指数)、F值（F measure）

外部准则1：纯度

- 纯度 $$\text{purity}(\Omega, C) = \frac{1}{N}\sum_{k}\max_{j}|\omega_k \cap c_j|$$

- $\Omega = \{\omega_1, \omega_2, \ldots, \omega_K\}$ 是簇的集合
- $C = \{c_1, c_2, \ldots, c_J\}$ 是类别的集合
- 对每个簇 ω_k：找到一个类别c_j，该类别包含ω_k中的元素最多，为n_{kj} 个，也就是说ω_k的元素最多分布在c_j中
- 将所有 n_{kj} 求和，然后除以所有的文档数目

- 计算示例



$\max_j |\omega_1 \cap c_j| = 5$ (class x, cluster 1);
$\max_j |\omega_2 \cap c_j|$ (class o, cluster 2);
$\max_j |\omega_3 \cap c_j| = 3$ (class ⋄, cluster 3)
纯度为 $(1/17) \times (5 + 4 + 3) \approx 0.71$

外部准则2：归一化互信息

$$NMI(\Omega,\mathrm{C})=\frac{I(\Omega,\mathrm{C})}{[H(\Omega)+H(\mathrm{C})]/2}$$

$$I(\Omega,\mathrm{C})=\sum_k\sum_j P(\omega_k\cap c_j)\log\frac{P(\omega_k\cap c_j)}{P(\omega_k)P(c_j)}$$

$$=\sum_k\sum_j\frac{|\omega_k\cap c_j|}{N}\log\frac{N|\omega_k\cap c_j|}{|\omega_k||c_j|}。$$

其中，$P(\omega_k)$、$P(c_j)$及$P(\omega_k\cap c_j)$分别是一篇文档属于ω_k、c_j及$\omega_k\cap c_j$ 的概率。

$$H(\Omega)=-\sum_k P(\omega_k)\log P(\omega_k)$$

$$=-\sum_k\frac{|\omega_k|}{N}\log\frac{|\omega_k|}{N}。$$

外部准则3：兰德指数

- 以将聚类看成是一系列的决策过程，即对文档集上所有 N(N−1)/2 个文档对进行决策。当且仅当两篇文档相似时，我们将它们归入同一簇中。TP（True-positive，真阳性）决策将两篇相似文档归入一个簇，而TN（True-negative，真阴性）决策将两篇不相似的文档归入不同的簇。在此过程中会犯两类错误： FP 决策会将两篇不相似的文档归入同一簇，而FN 决策将两篇相似的文档归入不同簇。RI 计算的是正确决策的比率，它实际上就是在8.3 节中提到的精确率（accuracy）：

$$RI = \frac{TP+TN}{TP+FP+FN+TN}$$

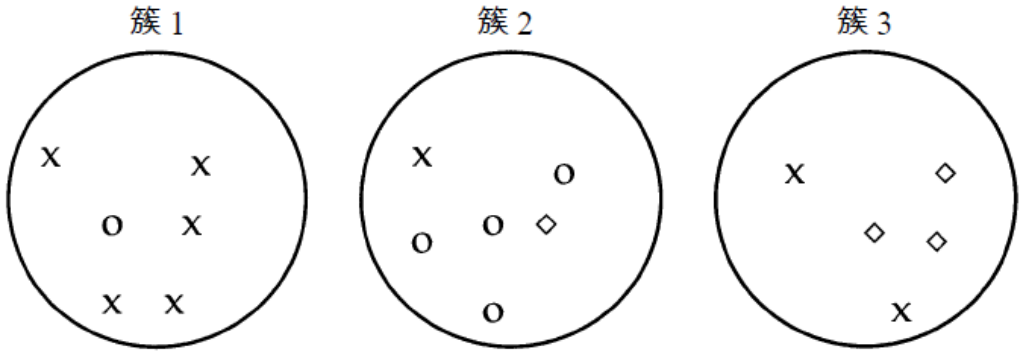
外部准则4：F值

- 可以使用8.3 节讨论的F 值来度量聚类结果，并通过设置β > 1 以加大对FN 的惩罚，此时实际上也相当于赋予召回率更大的权重。

$$P=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FP}} \qquad R=\frac{\mathrm{TP}}{\mathrm{TP}+\mathrm{FN}} \qquad F_{\beta}=\frac{(\beta^{2}+1)PR}{\beta^{2}P+R}$$

小结：聚类算法的评价

- 内部准则：簇内高相似度及簇间低相似度的函数表达
- 外部准则：将聚类结果和事先的类别情况进行比照，得到最后的评价结果



所有指标都从0 (非常差的聚类结果) 到 1 (完美聚类)

表16-2　图16-4中聚类算法中的四种外部评价指标

	purity	NMI	RI	F_5
下界	0.0	0.0	0.0	0.0
最大值	1.0	1.0	1.0	1.0
图16-4中的值	0.71	0.36	0.68	0.46

本讲内容：文本聚类

- **聚类概述**
  - 什么是聚类？在IR中如何用聚类？聚类的几个术语
- **K-均值聚类算法**
  - K-均值聚类中的基本准则
  - K-均值算法中簇的个数
- **聚类评价**
  - purity、NMI（Normalized Mutual Information，)、RI（Rand Index)、F measure）
- **基于模型的聚类**
- **层次聚类简介**
  - 层次聚类的簇相似度计算
  - 四种HAC算法：单连接、全连接 、组平均、质心法

从向量空间→基于模型

- **聚类要解决的基本问题**
  - Representation for clustering
    - Document representation
    - Need a notion of similarity/distance
  - How many clusters?

- **K-均值聚类**
  - 文档表示为向量、几何距离
- **基于模型的聚类**
  - 文档由模型生成

回顾：语言模型

怎样由文档生成语言模型？

- 一元语言模型（**unigram language model**）：

$$P_{uni}(t_1t_2t_3t_4) = P(t_1)P(t_2)P(t_3)P(t_4)$$

- 问题：已知样本D，求其模型M_D的参数$P(w|M_D)$。

$$P(d) = \frac{L_d!}{tf_{t_1,d}!tf_{t_2,d}!\cdots tf_{t_M,d}!} P(t_1)^{tf_{t_1,d}} P(t_2)^{tf_{t_2,d}} \cdots P(t_M)^{tf_{t_M,d}}$$

$$\vec{\theta}_D = (\theta_1, \theta_2, ..., \theta_L)$$
$$= (P(w_1|M_D), P(w_2|M_D), ..., P(w_L|M_D))$$

M_D的参数求解

$$\vec{\theta}_D^{\,*} = \arg\max_{\vec{\theta}_D} P(D|\vec{\theta}_D)$$

回顾：LM中相关性的表示

- **文档和查询表示决定了相关性的表示**
  - 表示为词项的集合→相关度为布尔运算结果
  - 表示为向量→相关度为向量的余弦相似度
  - 表示为随机变量→相关度为随机变量(二值或非二值)

- **灵活多样的概率表示**
  - 相关的概率：$P(R=1|Q=q,D=d)$ →$P(D=d|R=1,Q=q)$
  - 查询生成的概率：$P(Q=q|D=d)$ → $P(Q=q|M=M_d)$
  - 文档生成的概率：$P(D=d|Q=q)$ → $P(D=d|M=M_q)$

$O(R|Q=q,D=d)$

$$R(d;q) = LK(M_d \| M_q) = \sum_{t \in V} P(t|M_q) \log \frac{P(t|M_q)}{P(t|M_d)}$$

回顾：NB分类器

- **LM**

$$P(d \mid q) \propto P(d) \prod_{t \in q} P(t \mid M_d)$$

- **NB分类器**

$$P(c|d) \propto P(c) \prod_{1 \le k \le n_d} P(t_k|c)$$

- **NB的c_{map}**

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}(t_k|c)$$

- **MLE估计**

$$\hat{P}(c) = \frac{N_c}{N} \qquad \hat{P}(t \mid c) = \frac{T_{ct}}{\sum_{t' \in V} T_{ct'}}$$

基于模型的聚类

将模型的参数标记为 Θ，在 K-均值算法中，$\Theta=\{\vec{\mu}_1,\ldots,\vec{\mu}_K\}$

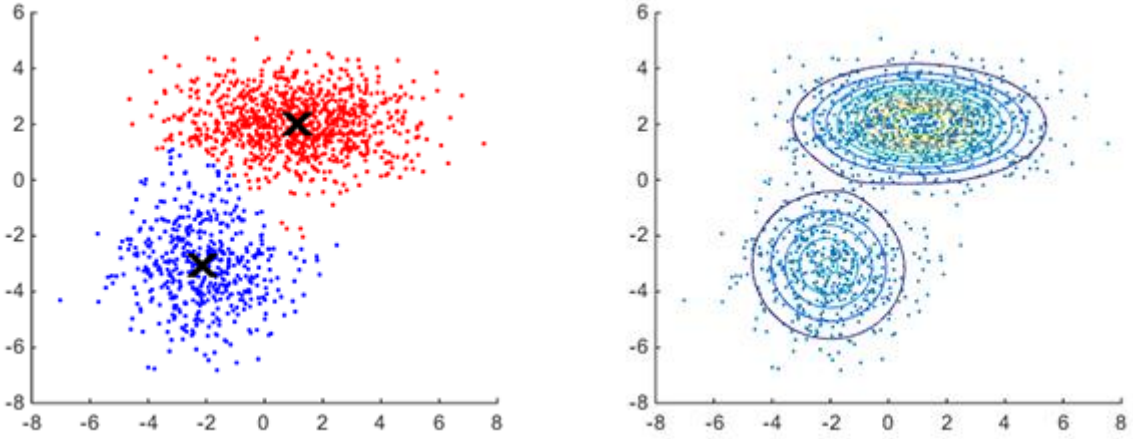
$$\Theta = \arg\max_{\Theta} L(D \mid \Theta) = \arg\max_{\Theta} \log \prod_{n=1}^{N} P(d_n \mid \Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} \log P(d_n \mid \Theta)$$

K个类别对应$\Theta = \{\Theta_1,\ldots, \Theta_K\}$， 而$\Theta_K$对应一个类别的一组参数

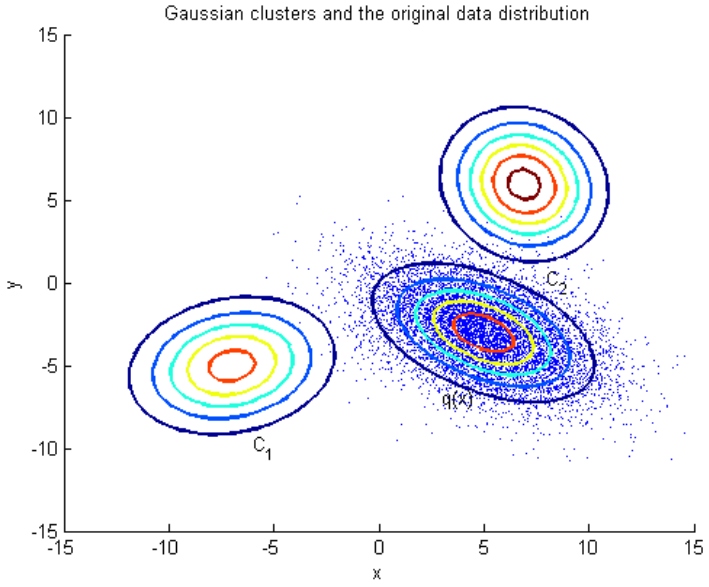
其中，$L(D|\Theta)$是度量聚类结果质量好坏的目标函数。给定具有同样簇数目的两个聚类结果，选择具有更大$L(D|\Theta)$值的结果。

在文本分类中，选择某篇具体文档的生成似然最大的那个类。
在这里，选择生成给定文档集的似然最大的聚类方法Θ。

模型示例：GMM, Gaussian mixture models



基于模型的聚类（Model-based clustering）方法假定数据产生自某个模型并试图从数据中恢复出该模型，这个恢复出的模型即定义了簇及文档到簇的归属。



多元贝努利混合分布

EM 是一个最大化$L(D|\Theta)$的迭代算法，可应用于不同类型的概率建模中。这里使用的概率分布模型是多元贝努利混合分布

$$P(d \mid \omega_k;\Theta) = \left(\prod_{t_m \in d} q_{mk}\right)\left(\prod_{t_m \notin d}(1-q_{mk})\right)$$

其中，$\Theta = \{\Theta_1,\ldots,\Theta_K\}$，$\Theta_k = (\alpha_k, q_{1k},\ldots,q_{Mk})$，$q_{mk} = P(U_m=1|\omega_k)$是模型的参数。$P(U_m=1|\omega_k)$是簇$k$中的文档包含词项$t_m$的概率。$\alpha_k$是簇$\omega_k$的先验概率，即在对$d$没有任何先验知识的情况下$d$属于$\omega_k$的概率。于是，混合模型如下：

$$P(d \mid \Theta) = \sum_{k=1}^{K} \alpha_k \left(\prod_{t_m \in d} q_{mk}\right)\left(\prod_{t_m \notin d}(1-q_{mk})\right)$$

该模型中，生成一篇文档的过程如下：首先以概率α_k选择一个簇ω_k，然后按照参数q_{mk}生成文档的词项。

Expectation Maximization (EM) 算法
用EM算法迭代求解模型参数

EM 在步骤上和*K*-均值算法类似，它在E 步（*Expectation Step*，期望）和从步（*Maximization Step*，最大化)这两步之间交替迭代，计算的参数是α_k和q_{mk}。

在M步中，可以按照如下方式对参数q_{mk}和α_k进行计算

$$\text{M步：}\ q_{mk} = \frac{\sum_{n=1}^{N} r_{nk} I(t_m \in d_n)}{\sum_{n=1}^{N} r_{nk}} \quad \alpha_k = \frac{\sum_{n=1}^{N} r_{nk}}{N}$$

其中，如果$t_m \in d_n$，那么$I(t_m \in d_n) = 1$，否则为0。r_{nk}表示d_n到簇k 的软分配概率。

$$\text{E步：}\ r_{nk} = \frac{\alpha_k \left(\prod_{t_m \in d_n} q_{mk}\right)\left(\prod_{t_m \notin d_n} (1 - q_{mk})\right)}{\sum_{k=1}^{K} \alpha_k \left(\prod_{t_m \in d_n} q_{mk}\right)\left(\prod_{t_m \notin d_n} (1 - q_{mk})\right)}$$

小结：基于模型的聚类

- **NB分类器**

$$c_{\text{map}} = \arg\max_{c \in \mathbb{C}} \hat{P}(c|d) = \arg\max_{c \in \mathbb{C}} \hat{P}(c) \prod_{1 \le k \le n_d} \hat{P}(t_k|c)$$

- **基于模型的聚类**

$$\Theta = \arg\max_{\Theta} L(D|\Theta) = \arg\max_{\Theta} \log \prod_{n=1}^{N} P(d_n|\Theta) = \arg\max_{\Theta} \sum_{n=1}^{N} \log P(d_n|\Theta)$$

$$P(d|\Theta) = \sum_{k=1}^{K} \alpha_k \left(\prod_{t_m \in d} q_{mk} \right) \left(\prod_{t_m \notin d} (1 - q_{mk}) \right)$$

- **Expectation Maximization (EM) 算法迭代求解模型参数**

本讲内容：文本聚类

- **聚类概述**
  - 什么是聚类？在IR中如何用聚类？聚类的几个术语
- **K-均值聚类算法**
  - K-均值聚类中的基本准则
  - K-均值算法中簇的个数
- **聚类评价**
  - purity、NMI（Normalized Mutual Information，)、RI（Rand Index)、F measure）
- **基于模型的聚类**
- **层次聚类简介**
  - 层次聚类的簇相似度计算
  - 四种HAC算法：单连接、全连接 、组平均、质心法

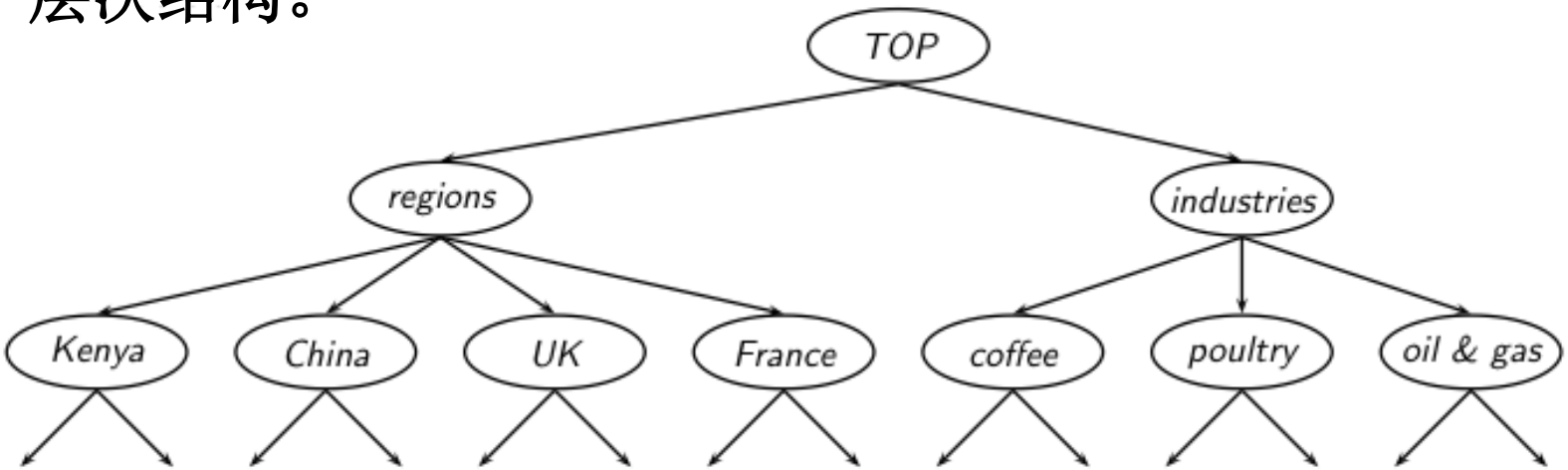
层次聚类

当效率因素非常重要时，我们选择扁平聚类算法。而当扁平算法的问题（如结构信息不足、簇数目需要预先定义、聚类结果非确定性）需要加以考虑时，我们则采用层次算法

- **扁平聚类**
  - 优点：概念简单、速度快
  - 缺点：算法返回的是一个无结构的扁平簇集合，它们需要预先定义簇的数目，并且聚类结果具有不确定性
- **层次聚类（hierarchical clustering）**
  - 输出一个具有层次结构的簇集合，因此能够比扁平聚类输出的无结构簇集合提供更丰富的信息。
  - 层次聚类不需要事先指定簇的数目，并且大部分用于IR中的层次聚类算法都是确定性算法。
  - 当然，层次聚类在获得这些好处的同时，其代价是效率降低。最普遍的层次聚类算法的时间复杂度至少是文档数目的平方级，而K-均值算法的时间复杂度是线性的。

层次聚类的目标

- **层次聚类的目标是生成类似于前面提到的Reuters目录的一个层次结构。**

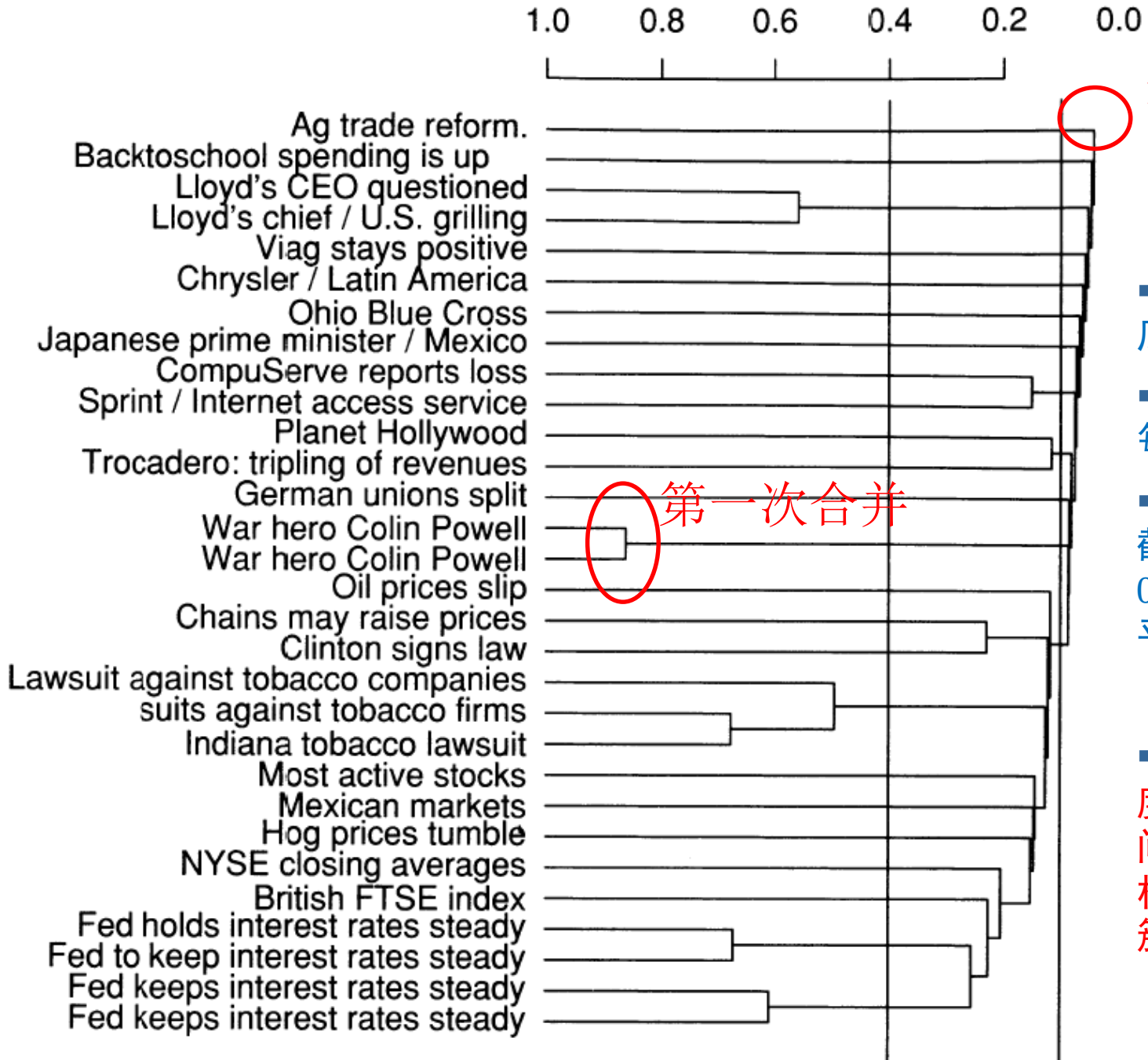


- **这个层次结构是自动创建的，可以通过自顶向下（分裂式 divisive）或自底向上（凝聚 agglomerate）的方法实现。**
  - 自底向上的算法一开始将每篇文档都看成是一个簇，然后不断地对簇进行两两合并，直到所有文档都聚成一类为止。
  - 而自顶向下的方法则首先将所有文档看成一个簇，然后不断利用某种方法对簇进行分裂直到每篇文档都成为一个簇为止。

层次凝聚式聚类 (HAC)

- **在IR 领域，HAC 方法的使用比自顶向下方法更普遍。最著名的自底向上的方法是层次凝聚式聚类(hierarchical agglomerative clustering，HAC)**

- **一开始每篇文档作为一个独立的簇**
- **然后，将其中最相似的两个簇进行合并**
- **重复上一步直至仅剩一个簇**
- **整个合并的历史构成一个二叉树**
- **一个标准的描述层次聚类合并历史的方法是采用树状图(dendrogram)**



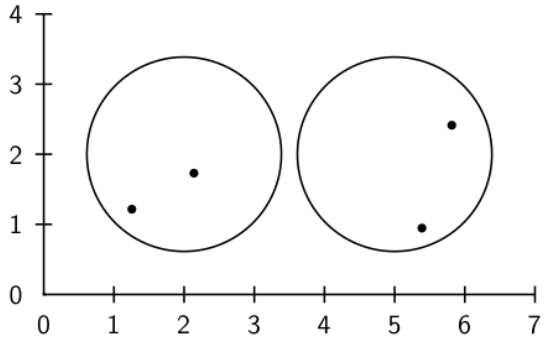
- 合并的历史可以从底往上生成
- 水平线上给出的是每次合并的相似度
- 我们可以在特定点截断 (比如 0.1 或 0.4) 来获得一个扁平的聚类结果

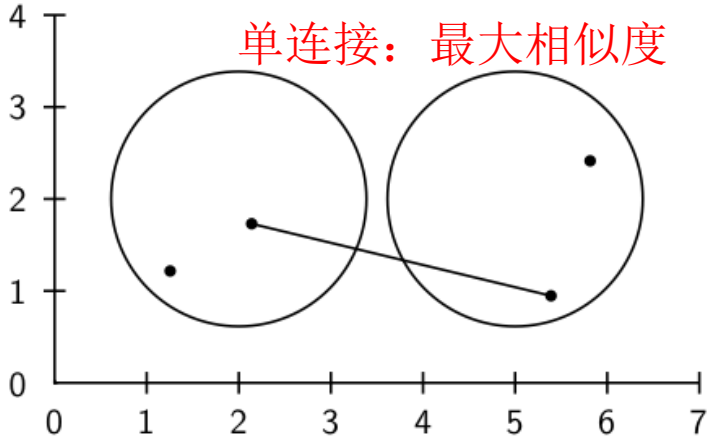
- 相似度：此前相似度都定义在文档之间，现在我们假设相似度定义在两个簇之间

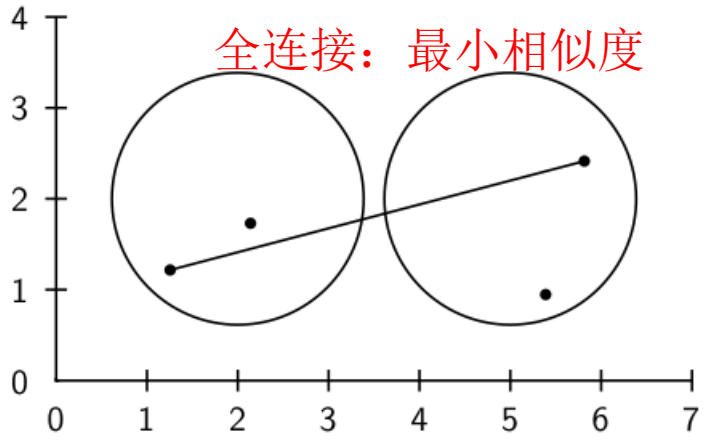
关键问题：如何定义簇相似度

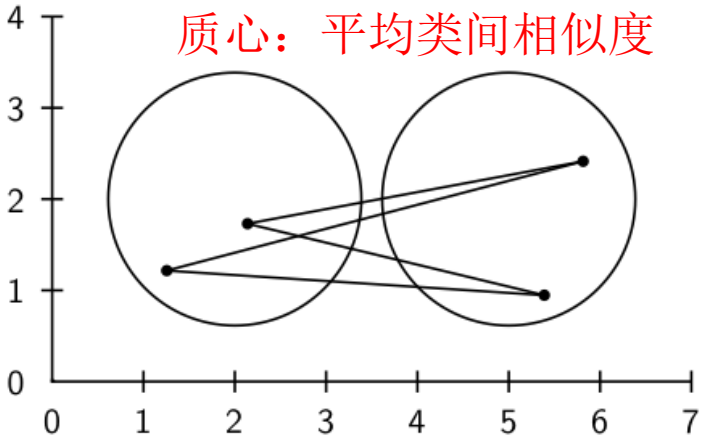
- **单连接(Single-link): 最大相似度**
  - 计算任意两篇文档之间的相似度，取其中的最大值
- **全连接(Complete-link): 最小相似度**
  - 计算任意两篇文档之间的相似度，取其中的最小值
- **质心法: 平均的类间相似度**
  - 所有的簇间文档对之间相似度的平均值 (不包括同一个簇内的文档之间的相似度)
  - 这等价于两个簇质心之间的相似度
- **组平均(Group-average): 平均的类内和类间相似度**
  - 所有的簇间文档对之间相似度的平均值 (包括同一个簇内的文档之间的相似度)

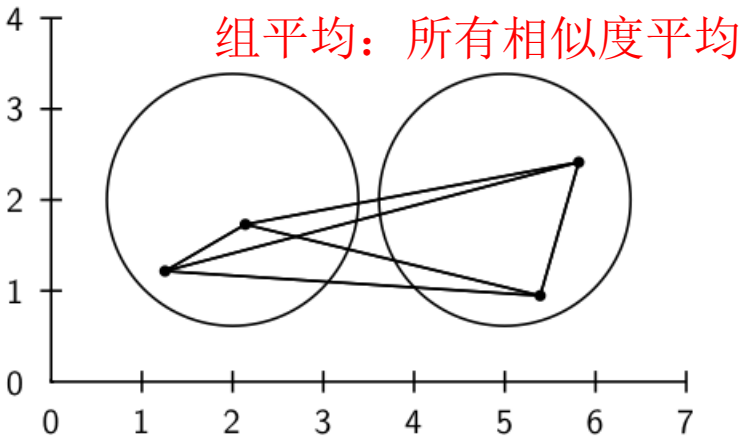
四种簇相似度示例1


0 1 2 3 4 5 6 7
0 1 2 3 4

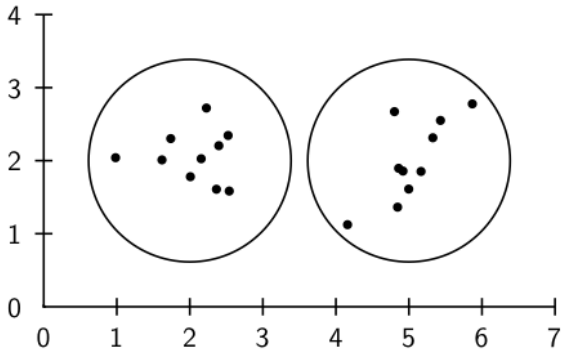

单连接：最大相似度
0 1 2 3 4 5 6 7
0 1 2 3 4


全连接：最小相似度
0 1 2 3 4 5 6 7
0 1 2 3 4

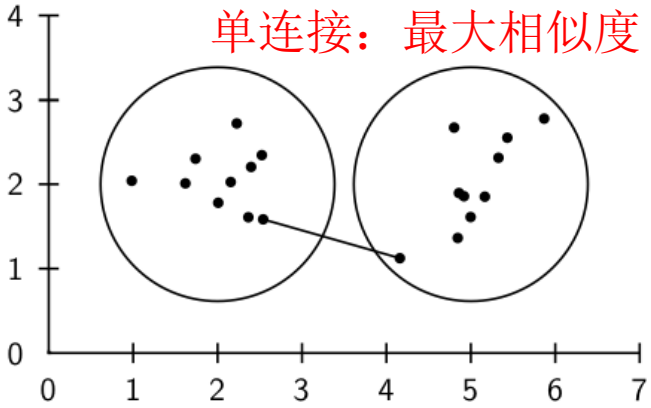

质心：平均类间相似度
0 1 2 3 4 5 6 7
0 1 2 3 4


组平均：所有相似度平均
0 1 2 3 4 5 6 7
0 1 2 3 4

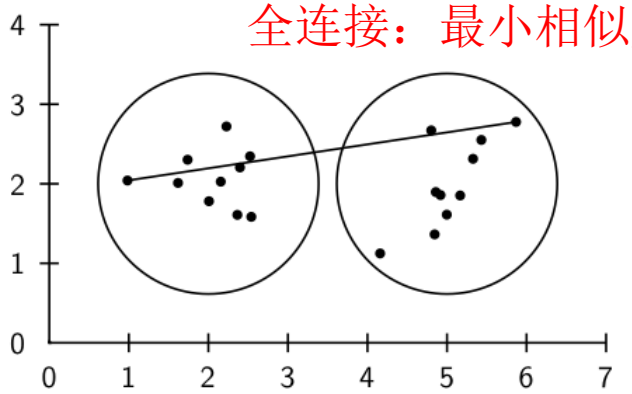
四种簇相似度示例2



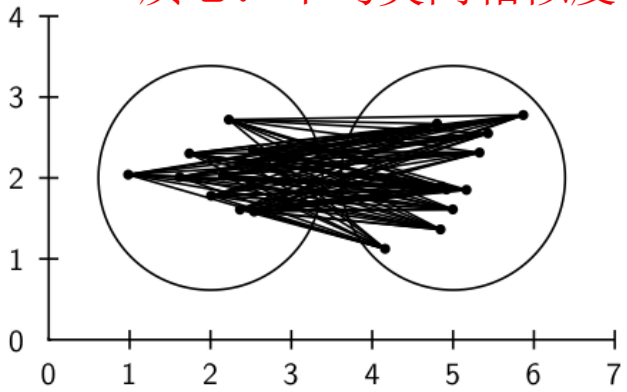
单连接：最大相似度



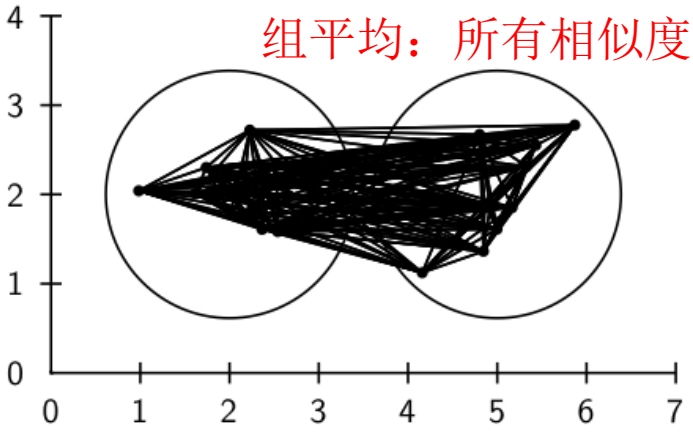
全连接：最小相似度



质心：平均类间相似度



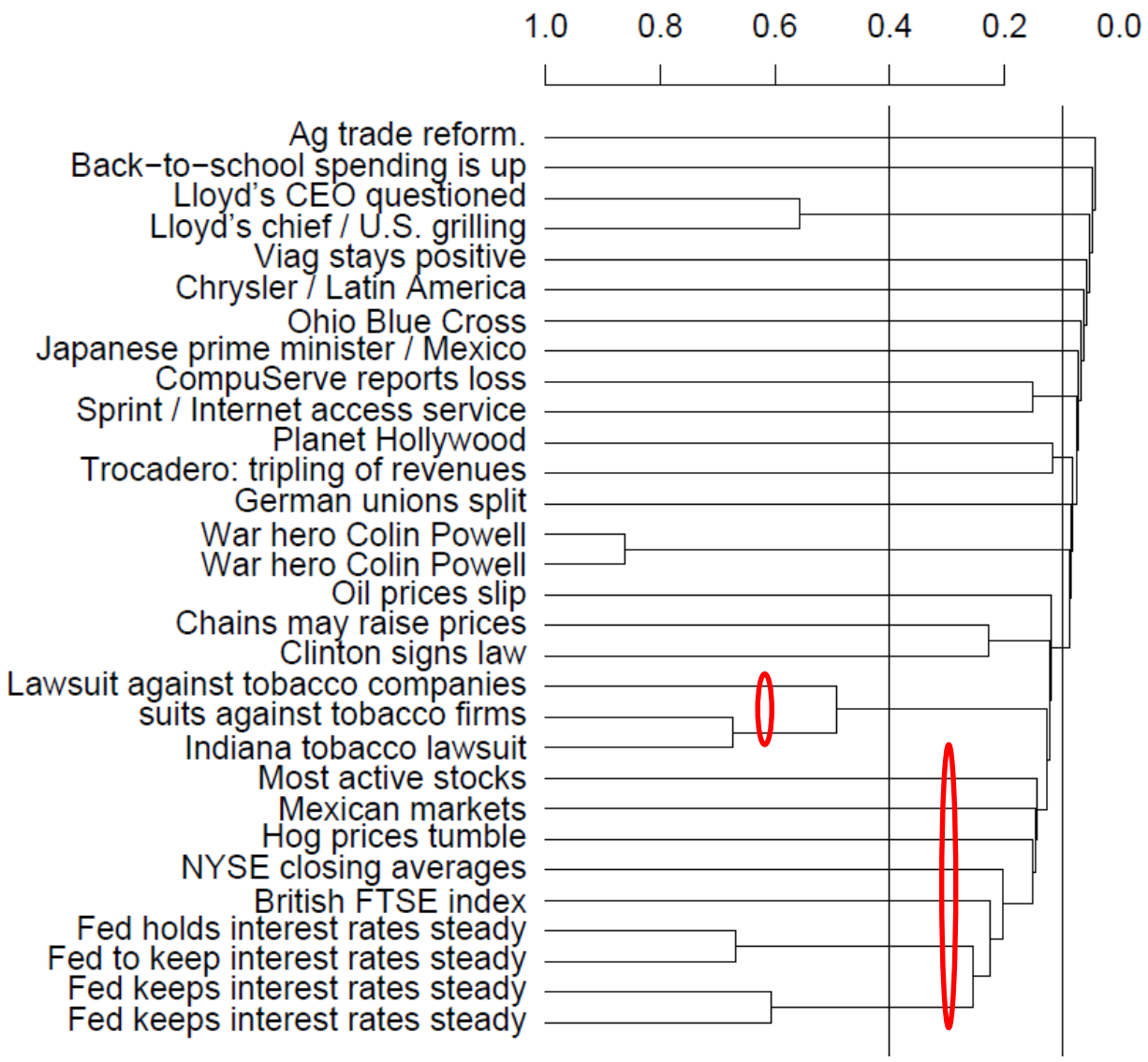
组平均：所有相似度平均



单连接/全连接算法

- 在单连接聚类（**single-link clustering**或**single-linkage clustering**）中，两个簇之间的相似度定义为两个最相似的成员之间的相似度。这种单连接的合并准则是局部的，即它仅仅关注两个簇互相邻近的区域，而不考虑簇中更远的区域和簇的总体结构。
- 在全连接聚类（**complete-link clustering** 或 **complete-linkage clustering**）中，两个簇之间的相似度定义为两个最不相似的成员之间的相似度，这也相当于选择两个簇进行聚类，使得合并结果具有最短直径。全连接聚类准则是非局部的，聚类结果中的整体结构信息会影响合并的结果。这种聚类实际上相当于优先考虑具有较短直径的紧凑簇，而不是具有长直径的松散簇，当然这种做法可能会对离群点较为敏感，比如某个远离中心的文档会显著增加候选簇的直径从而完全改变最后的聚类结果。

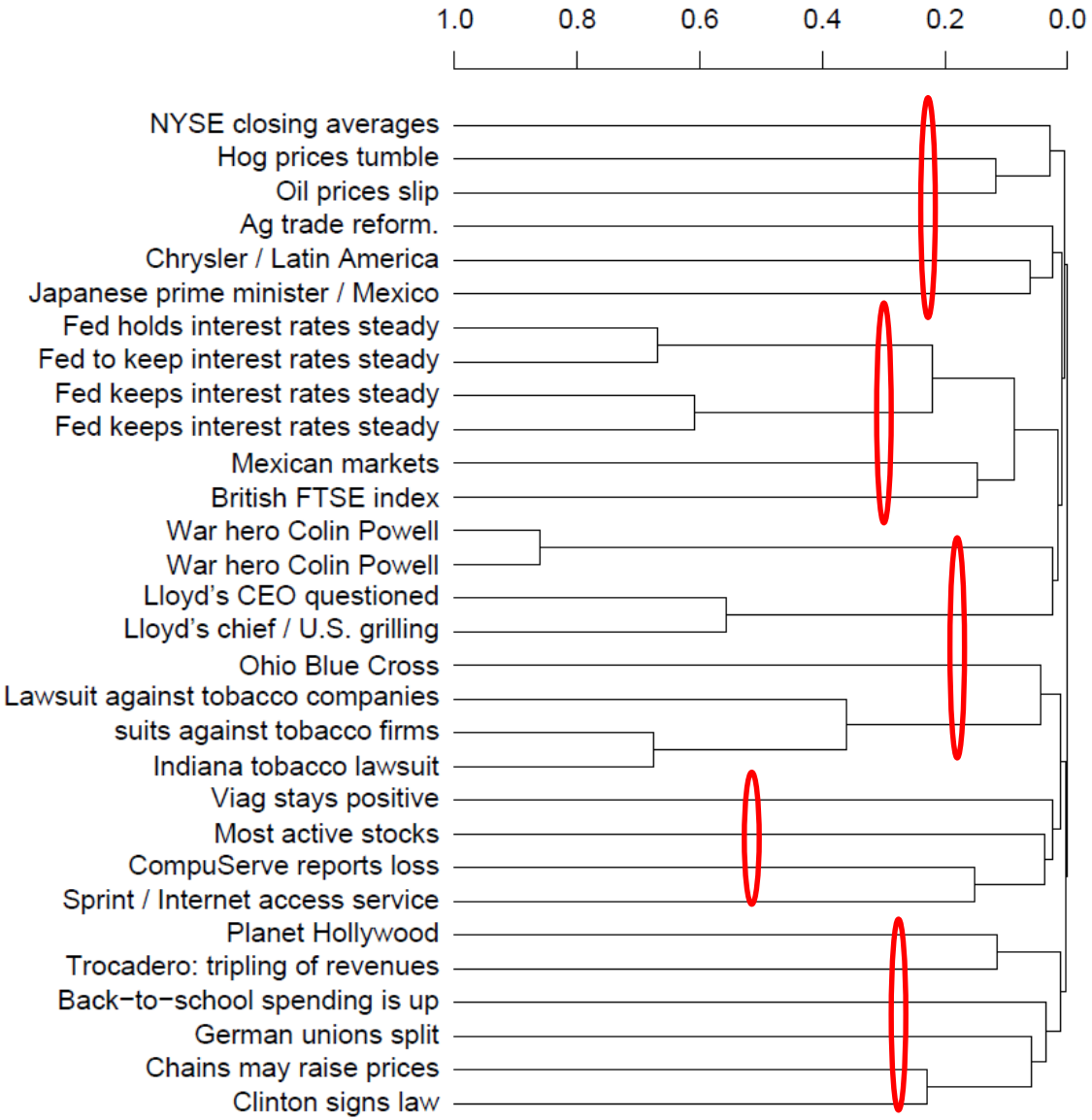
单连接算法产生的树状图



局部结构

- 注意：很多很小的簇（1 或 2 个成员）加入到一个大的主簇上面去
- 不存在2个簇或者3个簇的非常均衡的结果

全连接算法产生的树状图



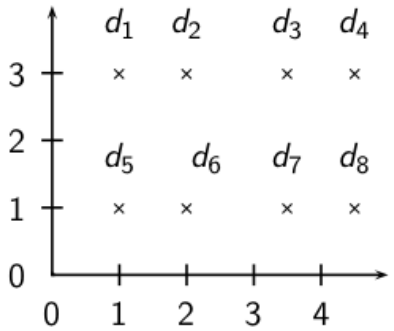
整体结构

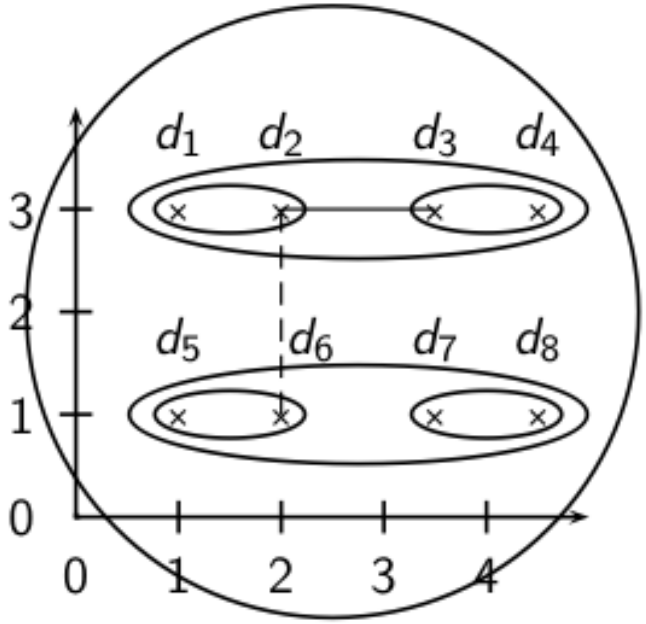
- 比单连接算法产生的树状图均衡得多
- 我们可以生成一个2个簇的结果，每个簇大小基本相当

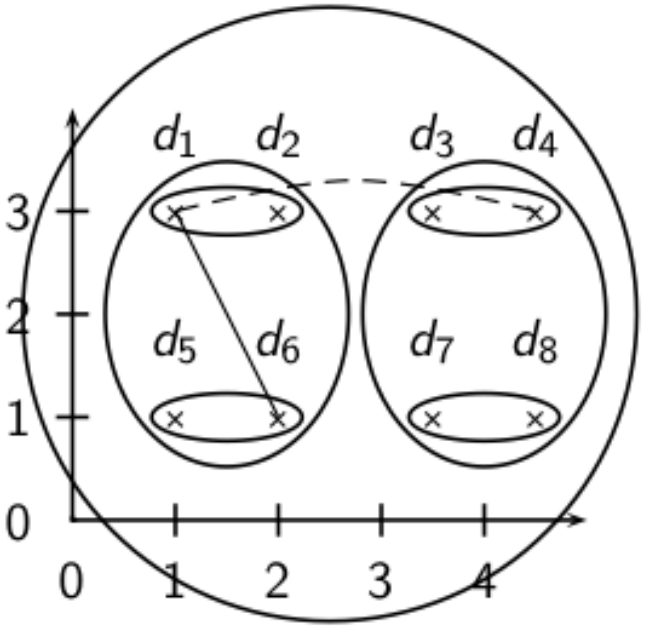
采用单连接和全连接方法进行聚类示例

左图：在单连接聚类中，上面两个簇的相似度为d2 和d3 的相似度（用实线表示），左边两个簇的相似度为d2 和d6 的相似度（用虚线表示），前者显然大于后者；

右图：在全连接聚类中，上面两个簇的相似度为d1 和d4 的相似度（用虚线表示），左边两个簇的相似度为d1 和d6 的相似度（用实线表示），前者显然小于后者



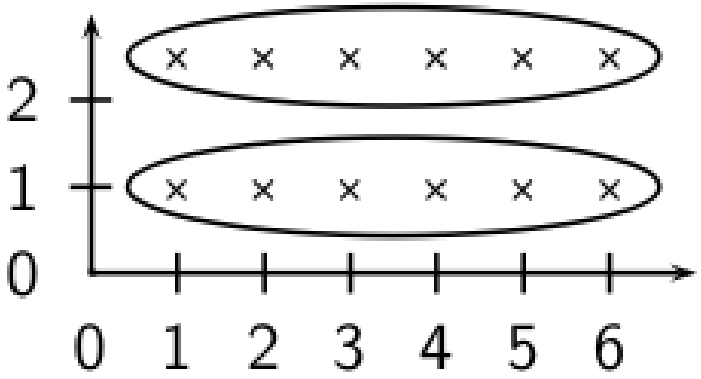


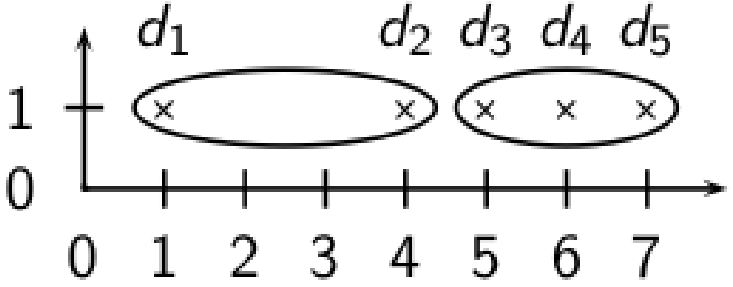


单连接和全连接聚类方法的缺点

- 单连接和全连接聚类方法将簇质量的计算过程简化成两个文档的单一相似度计算，其中在单连接方法中计算的是两篇最相似的文档之间的相似度，而在全连接方法中计算的是两篇最不相似的文档之间的相似度。
- 仅仅根据两篇文档来计算显然不能完全反映出簇中的文档分布情况，因此，这两种聚类方法产生的结果簇往往不是非常理想。

单连接和全连接聚类方法缺点示例





- 单连接方法的链化(Chaining)现象
  - 单连接聚类算法往往产生长的、凌乱的簇结构。对大部分应用来说，这些簇结构并不是所期望的。
- 全连接法: 对离群点非常敏感
  - 全连接聚类将d2和它的正确邻居分开----这显然不是我们所需要的
  - 出现上述结果的最主要原因是存在离群点 d1
  - 这也表明单个离群点的存在会对全连接聚类的结果起负面影响
  - 单连接聚类能够较好地处理这种情况

组平均凝聚式算法(GAAC)

- **GAAC（Group-average Agglomerative Clustering，组平均凝式聚类）通过计算所有文档之间的相似度来对簇的质量进行计算，因此可以避免在单连接和全连接准则中只计算一对文档相似度的缺陷。GAAC 也被称为组平均聚类（group-average clustering）或平均连接聚类（average-link clustering）。GAAC 可以计算所有文档之间相似度的平均值SIM-GA，其中 也包括来自同一簇的文档。当然，这种自相似度在这里并没有使用。计算公式如下：**

$$\text{SIM-GA}(\omega_i,\omega_j)=\frac{1}{(N_i+N_j)(N_i+N_j-1)}\sum_{d_m\in\omega_i\cup\omega_j}\sum_{d_n\in\omega_i\cup\omega_j,d_n\neq d_m}\vec{d}_m\cdot\vec{d}_n$$

其中，→d 是文档d的长度归一化向量，· 是内积运算符，N_i 和N_j 分别是ω_i 和ω_j 中的文档数目。

质心法HAC

- 簇相似度为所有簇间文档对之间相似度的平均值
- 一个原始的粗糙实现方法效率不高 ($O(N^2)$)，但是上述定义相当于计算两个簇质心之间的相似度：

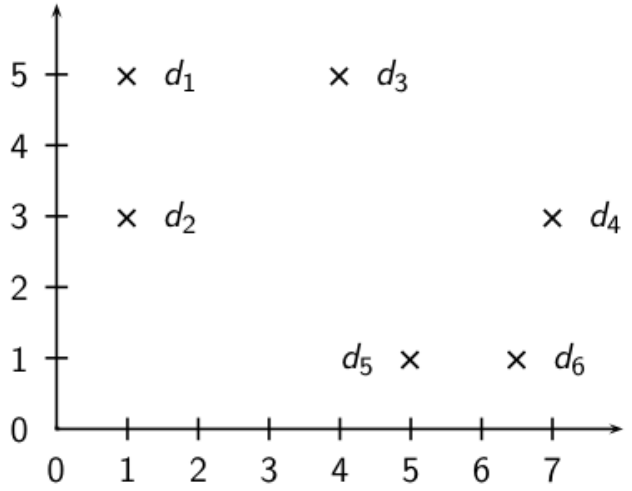
$$\text{SIM-CENT}(\omega_i, \omega_j) = \vec{\mu}(\omega_i) \cdot \vec{\mu}(\omega_j)$$

$$= \left(\frac{1}{N_i} \sum_{d_m \in \omega_i} \vec{d}_m \right) \cdot \left(\frac{1}{N_j} \sum_{d_n \in \omega_j} \vec{d}_n \right)$$

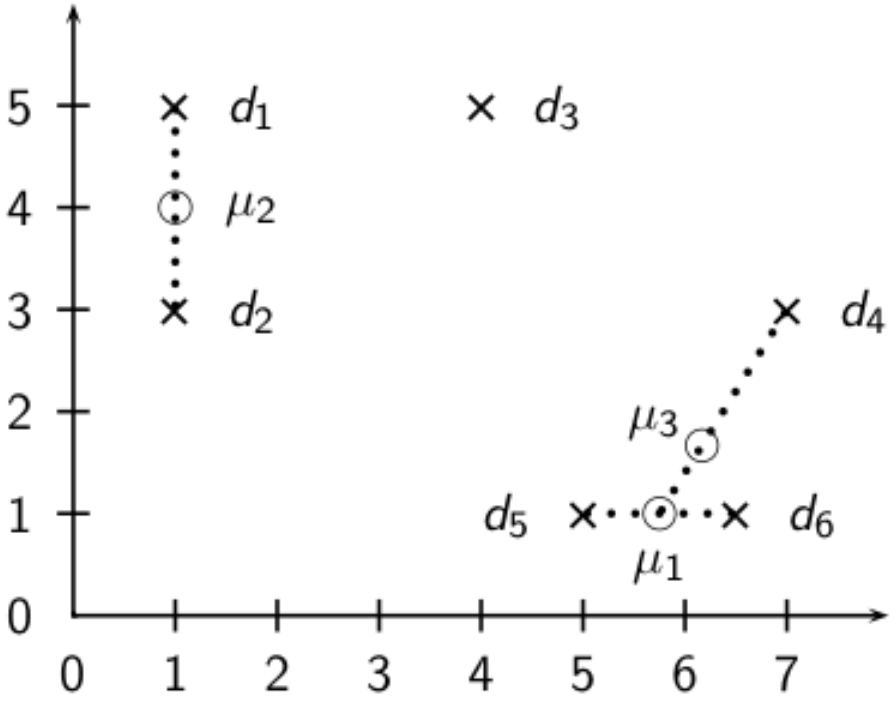
$$= \frac{1}{N_i N_j} \sum_{d_m \in \omega_i} \sum_{d_n \in \omega_j} \vec{d}_m \cdot \vec{d}_n$$

- 这也是质心HAC名称的由来
- 注意：这里是内积计算，而非余弦相似度

采用质心法进行聚类



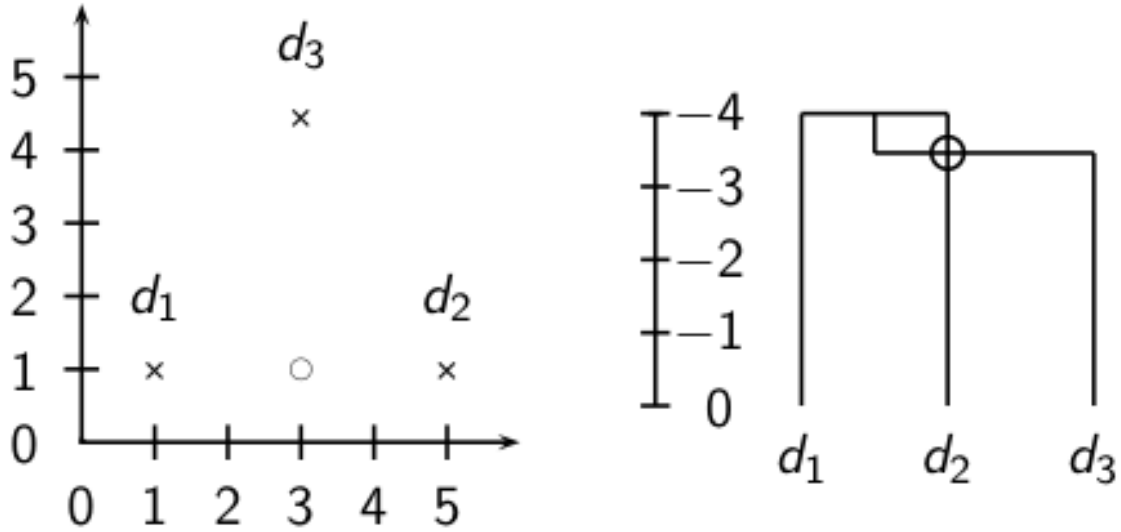
质心聚类中的三次迭代过程。每次迭代合并质心距离最近的两个簇



- 前两次迭代中，由于*<d5,d6>*和*<d1,d2>*具有最高的质心相似度，所以迭代后形成质心为*μ1* 的簇*{d5,d6}*及质心为*μ2* 的簇*{d1,d2}*。
- 在第三次迭代中，最高的质心相似度在*μ1* 和*d4* 之间，因此产生以*μ3* 为质心的簇*{d4,d5,d6}*。

相似度颠倒现象

- 与其他三种HAC 算法相比，质心聚类方法不是单调的，可能会发生相似度的颠倒现象。也就是说聚类过程中相似度值有可能会下降。



- 在相似度颠倒过程中，合并过程中相似度会增加，导致“颠倒”的树状图。图中，第一次合并 $(d_1 \cup d_2)$ 的相似度是-4.0，第二次合并的相似度 $((d_1 \cup d_2) \cup d_3) \approx -3.5$.

到底使用哪一个HAC聚类算法？

- 由于存在相似度颠倒，不使用质心法
- 由于GAAC不会受限于链化，并且对离群点不敏感，所以大部分情况下，GAAC都是最佳选择
- 然而，GAAC只能基于向量表示来计算
- 对于其他文档表示方法(或者如果仅仅提供了文档对之间的相似度)时，使用全连接方法
- 有些应用中适合用单链算法 (比如，Web搜索中的重复性检测，判断一组文档重复并不受那些离它们较远的文档所影响)

四种HAC算法的比较

方法	结合相似度	时间复杂度	是否最优?	备注
单连接	簇间文档的最大相似度	$\Theta(N^2)$	yes	链化效应
全连接	簇间文档的最小相似度	$\Theta(N^2 \log N)$	no	对离群点敏感
组平均	所有文档相似度的平均值	$\Theta(N^2 \log N)$	no	大部分应用中的最佳选择
质心法	所有簇间相似度的平均值	$\Theta(N^2 \log N)$	no	相似度颠倒


谢谢大家！