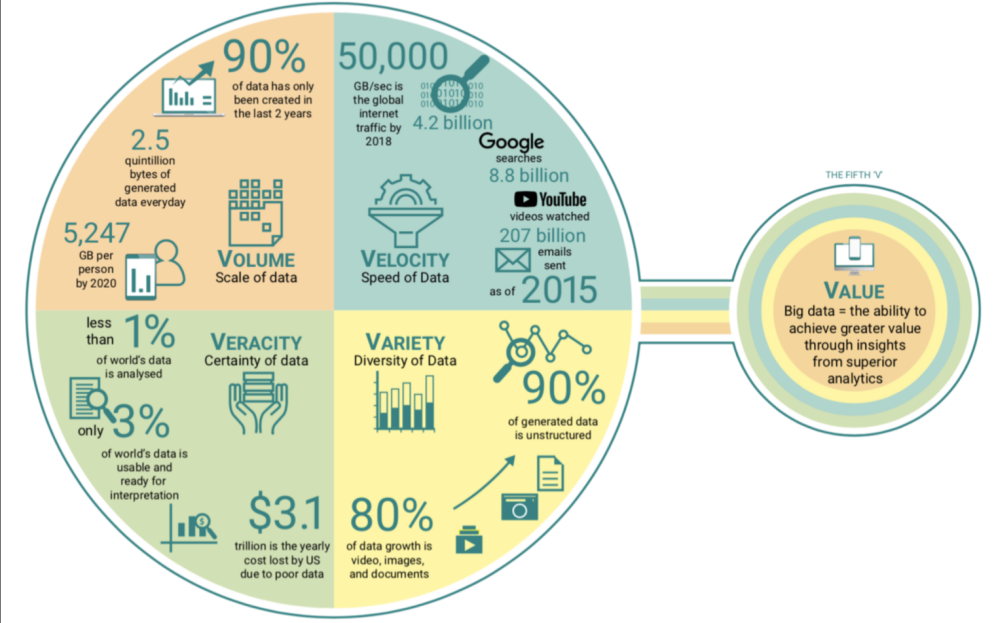
1



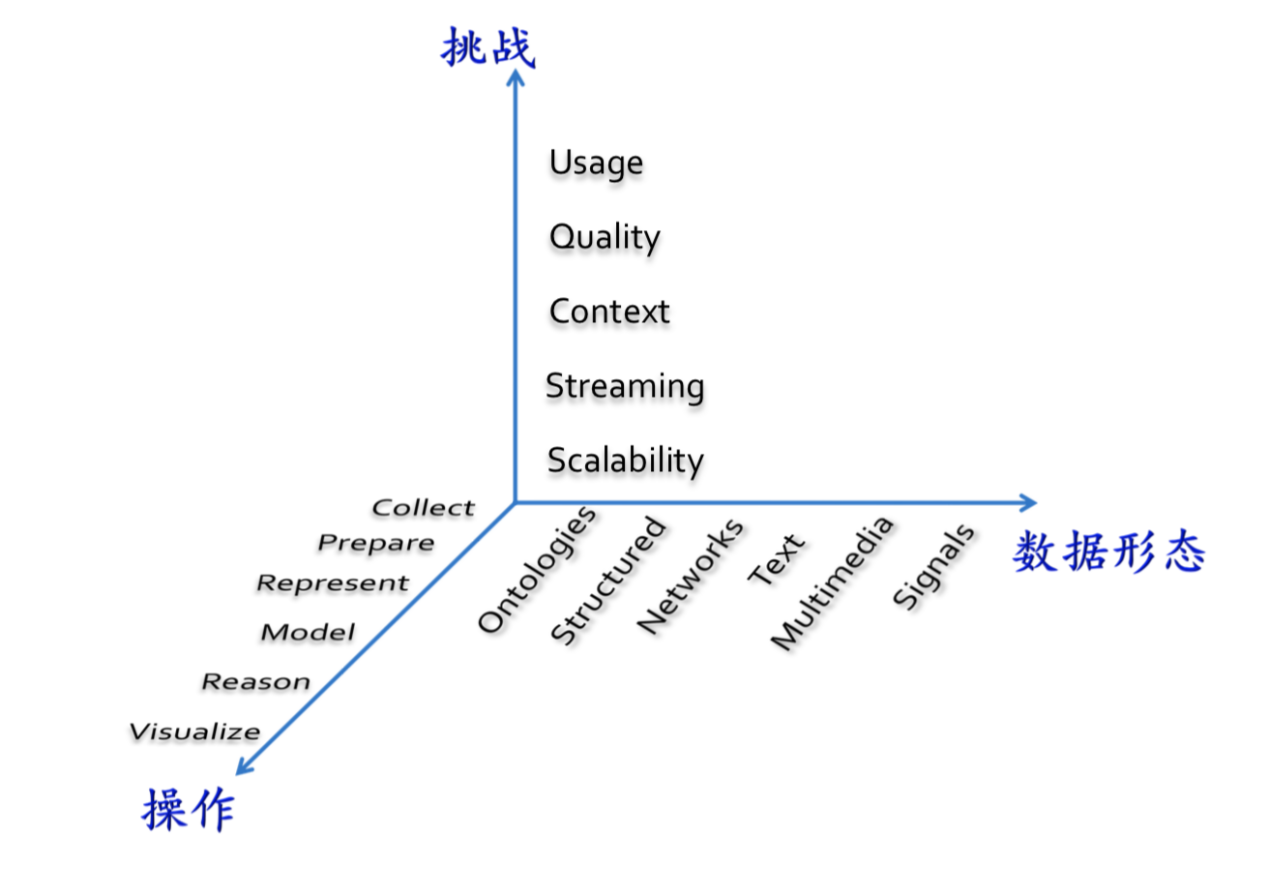
Variety volume velocity veracity -- value

给定大量数据，发现 ***有效性，可用性，出乎意料，可理解性***的模型

数据挖掘的常见任务：

* 描述类方法：找出人类可理解的模式来描述数据，聚类
* 预测类方法：使用一些变量来预测，推荐系统

数据挖掘的风险？



2

Hadoop是一个开发和运行大规模数据分析程序的软件平台。在大量普通服务器组成的集群中对海量数据进行分布式计算。

主要模块：

* Hadoop Common
* Hadoop distributed File System (HDFS)
* Hadoop YARN
* Hadoop MapReduce(Map and Reduce)

Hadoop采用了分布式存储方式，提高了读写速度，并扩大了存储容量。采用MapReduce 来整合分布式文件系统上的数据，可保证分析 和处理数据的高效。与此同时，Hadoop 还采用存储冗余数据的方式保 证了数据的安全性。

Hadoop中HDFS 的高容错特性，以及它是基于Java 语言开发的 ，这使得Hadoop可以部署在低廉的计算机集群中，同时不限于某个操 作系统。Hadoop 中HDFS 的数据管理能力，MapReduce 处理任务时 的高效率，以及它的开源特性，使其在同类的分布式系统中大放异彩， 并在众多行业和科研领域中被广泛采用。

|  |  |  |
| --- | --- | --- |
|  | 传统关系型数据库 | MapReduce |
| 数据大小 | GB | PB |
| 访问 | 交互型和批处理 | 批处理 |
| 更新 | 多次读写 | 一次写入多次读取 |
| 结构 | 静态模式 | 动态模式 |
| 集成度 | 高 | 低 |
| 伸缩性 | 非线性 | 线性 |

RDBMS适合点查询和更新，MapReduce适合批处理

Hadoop 是一个能够对大量数据进行分布式处理的软件框架，并且是以一种 可靠、高效、可伸缩的方式进行处理的，具有以下优点:

1. 可靠的，维护多个工作数据副本
2. 高效的，***并行***方式工作，加快处理速度，可伸缩
3. 成本低，高容错，可以兼容廉价服务器群
4. Linux平台运行，Java编写
5. 支持多种编程语言

Hadoop相关项目

Avro用于数据序列化的系统;

HDFS是一种分布式文件系统，运行于大型商用机集群，HDFS为HBase提供了高可靠性的底层存储支持;

HBase位于结构化存储层，是一个分布式的列存储数据库;

MapReduce是一种分布式数据处理模式和执行环境，为HBase提供了高性能的计算能力;

Zookeeper是一个分布式的、高可用性的协调服务，提供分布式锁之类的基本服务，用于构建分布式应用，为HBase提供了稳定服务和failover机制;

Hive是一个建立在Hadoop 基础之上的数据仓库，它提供了一些用于数据整理、特殊查询和分析存储在Hadoop 文件中的数据集的工具;

Pig是一种数据流语言和运行环境，用以检索非常大的数据集，大大简化了 Hadoop常见的工作任务;

Sqoop为HBase提供了方便的RDBMS数据导入功能，使得传统数据库数据向HBase中迁移变的非常方便。

HDFS和MapReduce是Hadoop的两大核心。而整个Hadoop的体系结构 主要是通过HDFS 来实现对分布式存储的底层支持的，并且它会通过 MapReduce 来实现对分布式并行任务处理的程序支持

HDFS上的数据操作，适合大量的大文件（平均500M以上，一次写入，多次读出，不能被修改，只能被末尾添加），创建新文件，删除文件，修改文件名和文件属性。

Hadoop-分布式开发，每一个map任务和每一个reduce任务都可以同时运行于一个单独的计算节点上。

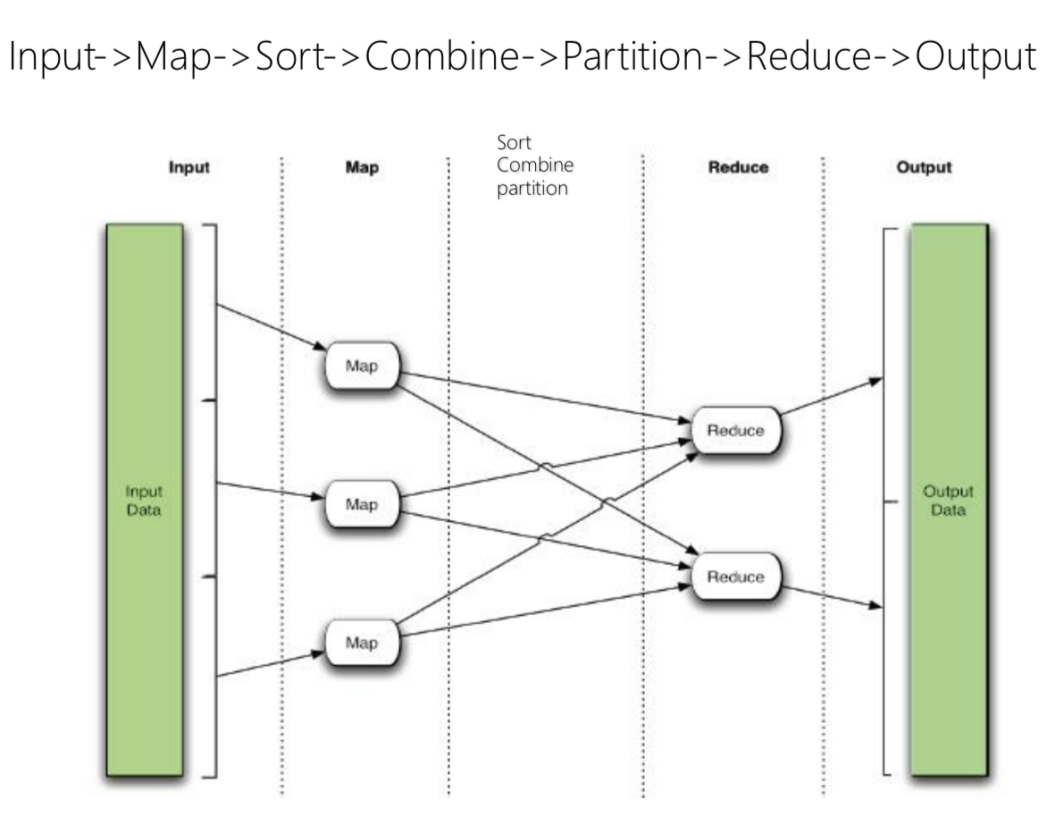
1. 数据分布存储  
   2. 分布式并行计算  
   3. 本地计算  
   4. 任务粒度  
   5. 数据分割(Partition) 6. 数据合并(Combine) 7. Reduce  
   8. 任务管道

3

Hadoop：支持在由普通计算机组成的集群中运行海量数据的分布式计算，它可以让应用程序轻松扩展到上千个节点和PB级别的运算

两大核心内容：

* MapReduce任务的分解，结果的汇总
* HDFS NameNode DataNode Client



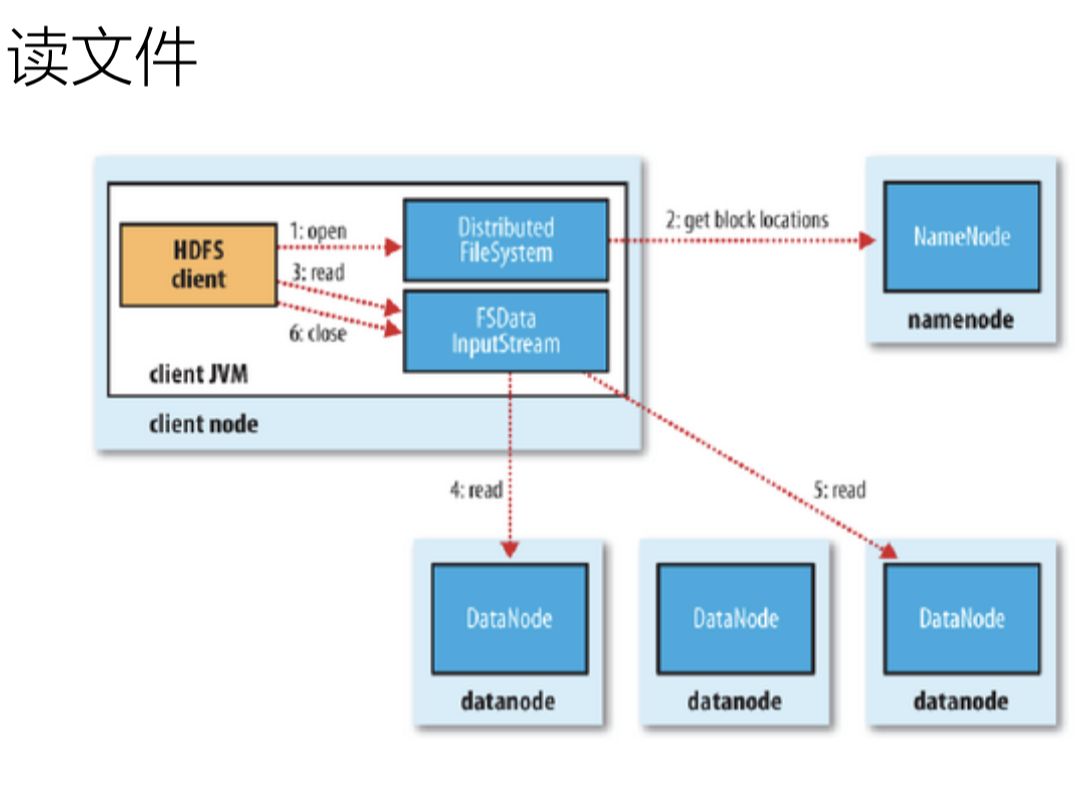
***HDFS***

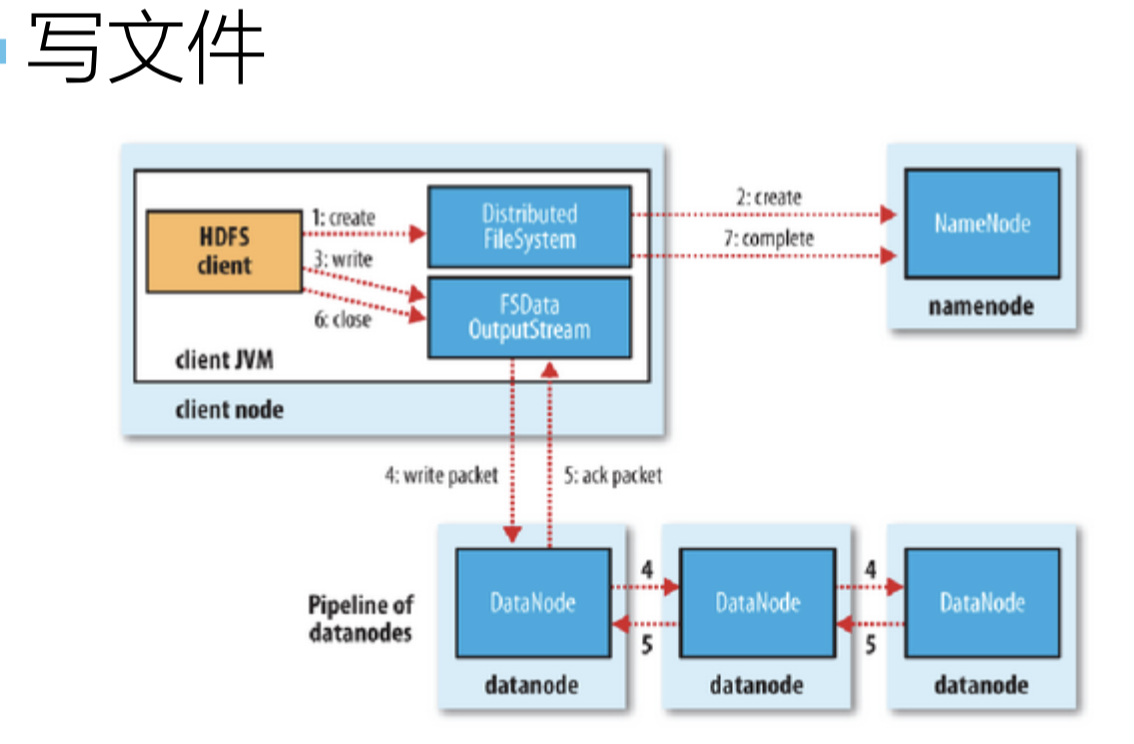
1. Block
2. Namenode & Secondary Namenode
3. Datanode

***Namenode***：分布式文件系统中的管理者，存储文件系统的metadata，主要负责管理文件系统的命名空间，集群配置信息，存储块的复制。两个文件EditLog, FsImage. 两个映射Filename -> BlockSequence , Block -> DatanodeList。单点风险

***Secondary Namenode：***不是备用Namenode，是秘书。合并和保存EditLog, FsImage, checkPoint.period , chechPoint.size

***DataNode :*** 存储文件的基本单元，它存储文件快在本地文件系统中，保存了文件快的meta-data，并周期性的发送所有存在的文件块的报告给namenode.





***HBase***

HBase是在HDFS上开发的，它提供了一种容错的方式存储大量的稀疏数据。HBase系统包含一组表，每个表包含行和列，每个表定义一个主键的元素，所有对HBase访问尝试都必须使用这个主键。

* 一个数据库
* 分布式的，非结构化的，稀疏的，面向列的
* 基于HDFS，山寨版的Bigtable，集成了可靠性，高性能，可伸缩性

***HBase角色***

Zookeeper 一个高效的、可扩展的协调系统，是Hadoop的子项目，不属于HBase

HMaster管理用户对Table的增删改查操作，管理HRegionServer，负载均衡

HRegionServer管理一系列HRegion Region为读写的场所。

HRegion 对应Table的一个Region， HStore, HLog

HStore对应了Table中的一个Column Family 的存储， MemStore, StoreFile(HFile)

StoreFile Compact多个64MB的文件区域存储256MB的文件 Split将文件分离存储

***HBase数据模型***

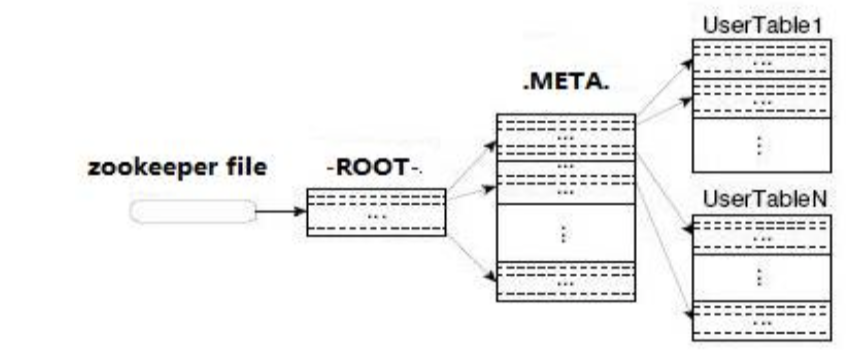
Table & Column Family: 记录数不断增多后，Table会逐渐分裂成多份splits，成为regions，一个region由[startkey, endkey）表示。不同的region会被Master分配给相应的RegionServer进行管理

三层结构

.META. : 记录用户表的Region信息，可以有多个region

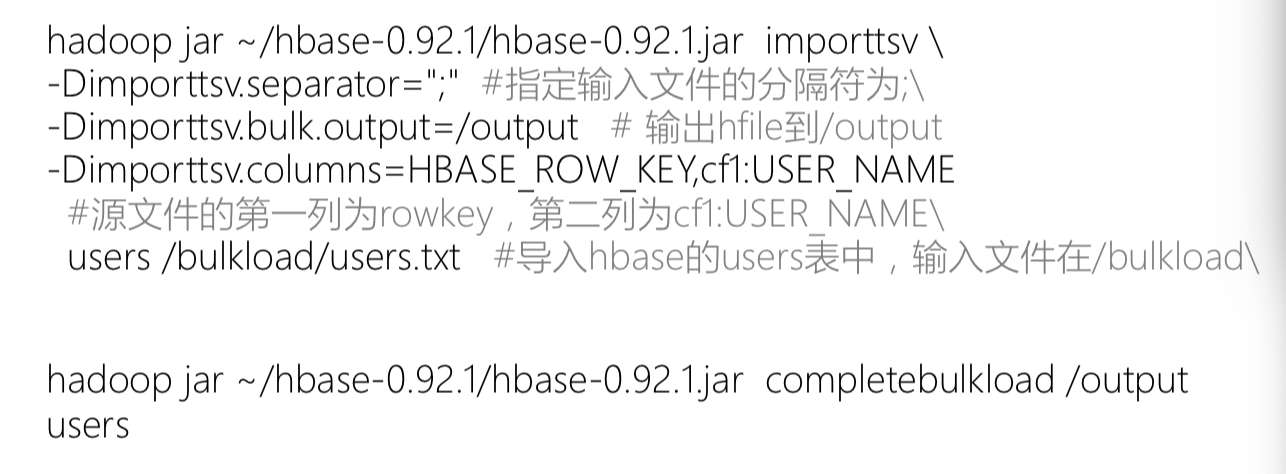
-ROOT- : 记录.META.表的Region信息，只有一个region

Zookeeper : 记录了-ROOT-表的location



数据导入Sybase -> HBase

* Bcp out
* Importtsv



访问接口

Hadoop & HBase提供JAVA接口，Thrift支持多语言访问HBase

***Hive***

Hive是基于Hadoop的数据仓库工具

学习成本低，提供完整的sql查询功能，将结构化的数据结构映射为一张数据库表。将sql语句转为MapReduce任务进行运行。通过类似SQL语言，快速实现简单的mapreduce统计，不必开发专门的mapreduce应用

Hive体系结构

|  |  |
| --- | --- |
| 服务协同与管理 | Hive web interface |
| 服务器应用 |  |
| Hive |  |
|  | mapreduce |
|  | HDFS |

Hive相当于一套Hadoop访问结构

* 使用单独的数据库存储元数据
* 定义合理的表分区和键，设置合理的bucket数量，表压缩
* 定义外部表使用规范
* 合理控制mapper reducer数量

***Pig***

严格用于使用Pig分析Hadoop中的数据的语言叫做Pig Latin. 一种高级数据处理语言。用Pig Latin编写Pig脚本，并使用任何执行机制执行，应用Pig框架的转换来生成所需输出。在内部 Pig将这些脚本转换为一系列Map Reduce作业。

角色

解析器Parser 检查语法

优化器Optimizer 逻辑计划传递后，执行逻辑优化

编译器 将优化的逻辑计划转化为Map Reduce作业

执行引擎 排序提交，并在Hadoop上执行

Pig使用延迟评价，支持ETL，能够在管道中任意时刻存储数据，支持管道分裂。Pig Latin是程序性的，SQL是声明性的，可以制定连接的两个表，但不能指定要使用的连接实现。

4 链接分析

图数据

将Web表示为有向图， 节点表示网页，边表示超链接

网络是巨大的，充满了许多不可信，过时和随机的东西

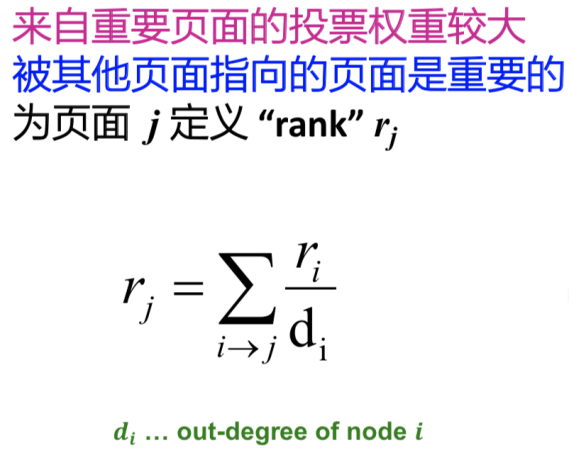
挑战：存在多个数据的来源，谁能被trust / 查询数据的最佳回答是什么？

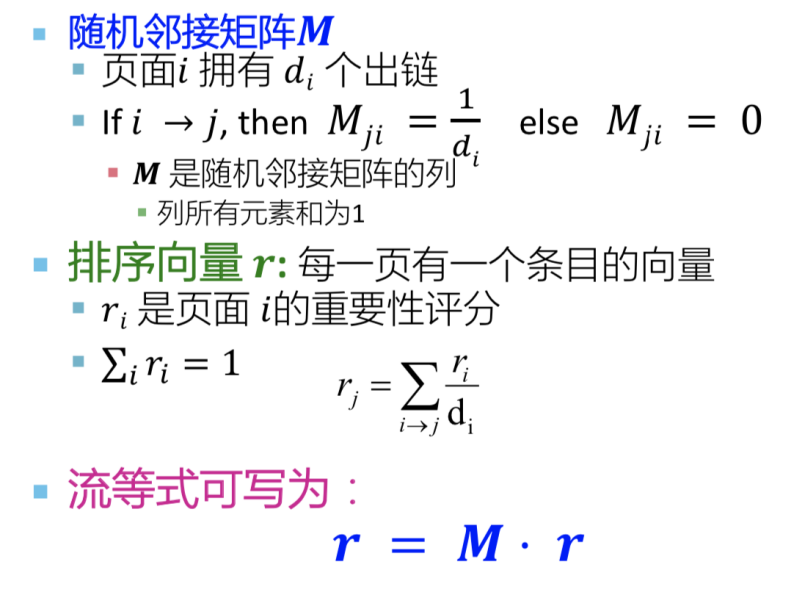
重要性都是不平等的，权重！

PageRank “流”模型

出链和入链，入链组成自己的权重，出链平分自己的权重

矩阵等式？





r表示所有节点组成的列，而M的构造左下入链，右上出链

Page rank完整算法！！

R= (E-dM)^-1 · ((1-d)/n)I

5 数据降维

矩阵的秩

对一个3\*3的矩阵A，如果A有秩为2，将其特征向量作为basis vectors，然后使用新的coordinates，[1 0] [0 1] [1 -1]

降维方法汇总

* 特征值与特征向量
* PCA主元素分析
* LDA线性判别分析
* 因子分析
* SVD奇异值分析

SVD

贝叶斯定理

6 分类A

分类问题中，某些领域的每个实体都可以放在一个离散的类别集中：

* 二元分类
* 多媒体分类
* 非排他性分类
* 排名ranking

评估规则及其预测的许多标准

* 过量错误
* 不同类型的错误
* 操作成本

每个被分类的对象，都以一对(x,y)表示。前者为对象的描述，后者为标签

成功或失败的机器学习分类，往往取决于选择良好的描述对象描述。描述的选择也可以被视为一个学习的问题。

数据类型

* 物理属性
* 行为属性
* 上下文
* 历史记录

文本与超文本

email电子邮件

protein序列，Unix system calls序列

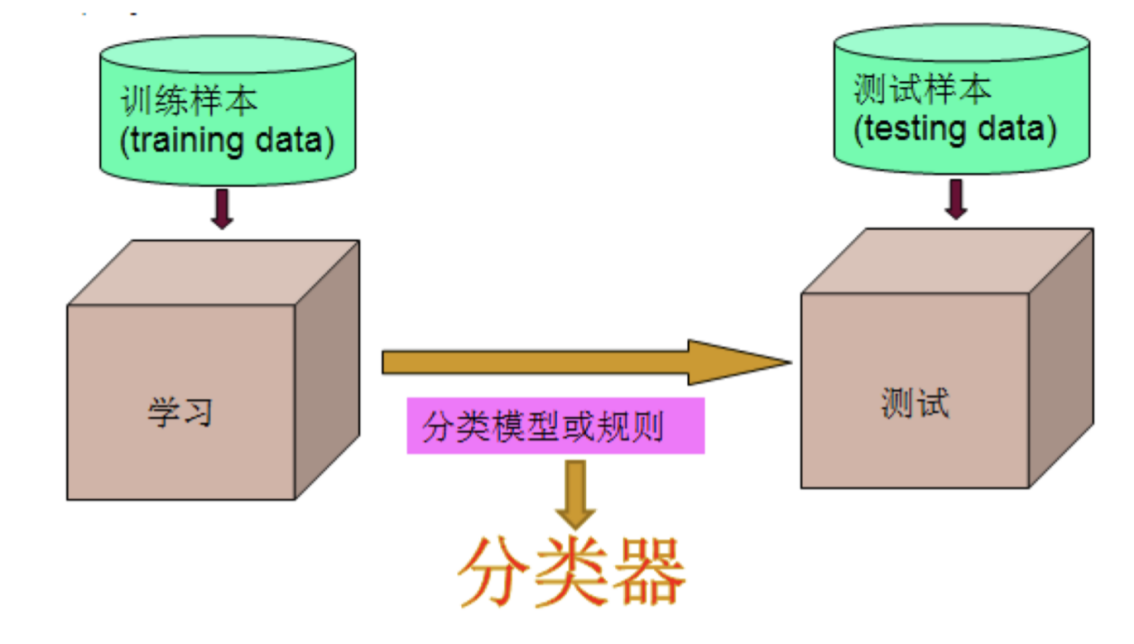
图graph

Training and Validation

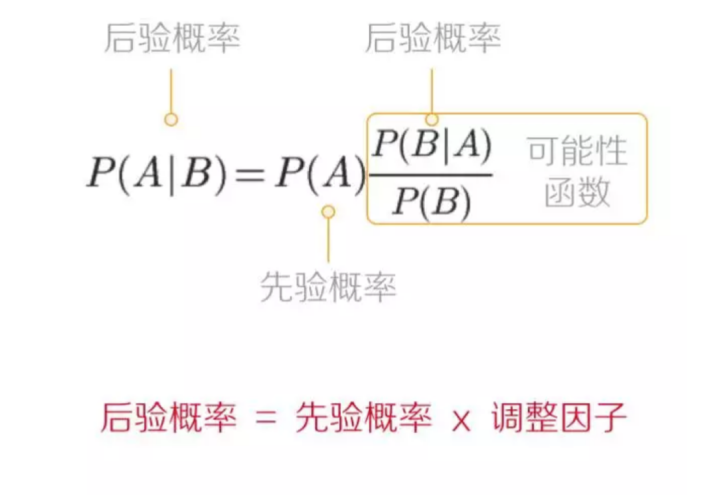
问题： 过拟合， 保守的精度估计， 测试数据上良好的分类器

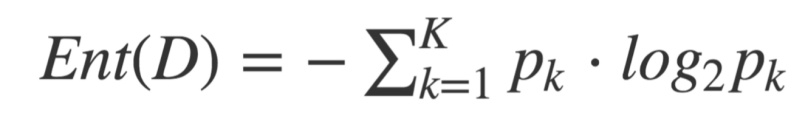
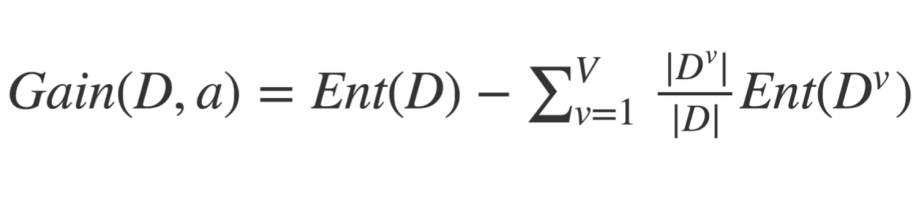
分类问题： 1 两类问题 2 多类问题

处理方法：1 直接分为多类 2 拆分为多个两类问题 3 排除法，排他分类



常用分类方法：

* 贝叶斯
* 决策树 - 构建

***ID3算法***使用所有没有使用的属性并计算与之相关的样本熵值，选取其中熵值最小的属性，生成包含该属性的节点。pk表示第k类样本所占的比例。（如色泽的乌黑，白亮，青紫，都是第K类的）计算信息增益的公式。总的信息增益，减去集合中每个属性的信息增益（Dv/D 是指吴青色的样本的占比）

***C4.5算法***

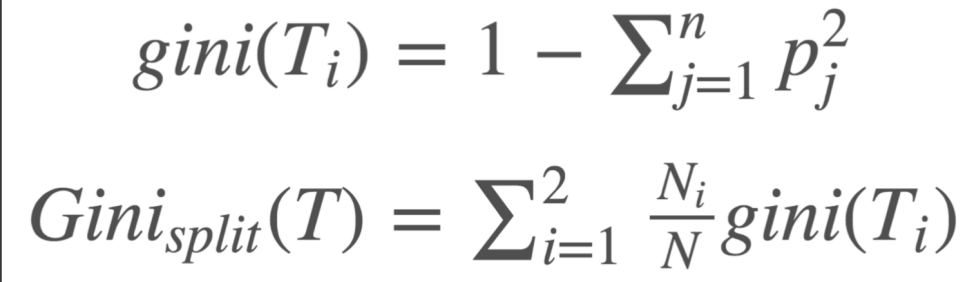
信息增益偏向于选择分歧比较多的属性值。不能处理连续的属性

CART算法

7 分类B

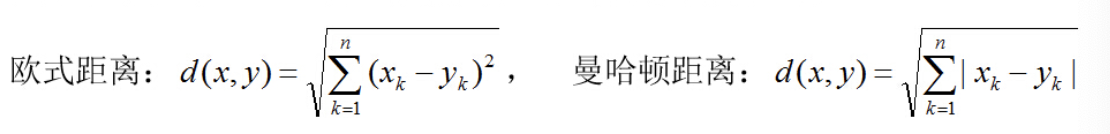
CART 分类回归树算法: 将样本递归划分进行建树过程，用验证数据进行剪枝。

GINI指数：总体内包含的类别越杂乱，该指数越大



***KNN算法***

距离函数，组合函数（民主投票，加权投票），邻居的数目K. 邻居的种类决定自身被分入哪一个类。哪个邻居多进入哪个类。



K的取值会影响类别的判定，距离的计算使用哪种方法？

归一化，Xs = (X - Min)/(Max - Min)

7 聚类

给定一组点，用点与点之间的距离的概念，将点分成若干个簇。集群成员彼此相近，不同集群的成员则不同。

Usually点在高维空间中，相似度是通过距离度量来定义的。

难点与挑战

许多应用程序涉及10个或1000个维度（难），高维空间中，几乎所有的点都在相同的距离上

硬聚类 1:1 软聚类1:n

soft用于创建可浏览层次结构之类的应用程序根据有意义

划分方法

在给定数据库D中的n个对象的分区构造一个k个簇的集合。

全局最优：穷举所有分区

启发式方法：

* K-means
* K-medoids

启发式方法被希望接近最优的可能答案。是一种经验法则，有根据的猜测。假设文件是实值向量

K-means 算法

随机选取k个中心点，计算其他点的距离，站队；然后对每个站队，通过平均值的方法，选举虚拟出新的中心点，然后再次站队。直至两次迭代结果不变

* 优势：相对有效；通常在局部最优处终止。收敛速度块，聚类效果好。面对大数据，可以保持可伸缩性和高效性。
* 劣势：只有定义了平均值时才适用。需要预先指定集群的数量；噪声数据和异常值会影响，偏移严重；不适合发现非凸形状的簇。

***层次方法***

计算类别的相似度类创建一个有层次的嵌套的树

自下而上凝聚，最初每个点都是一个集群，反复将两者结合起来，最近的集群合成一个。分裂自顶向下，从一个集群开始，递归地分割他。

对点而言，点维护一组群组，点属于最近的集群。

步骤

* 初始化，将每个样本都视为一类
* 计算各个聚类之间的相似度
* 寻找最近的两个聚类，将他们归为一类
* 重复直至所有样本都归为一类

距离衡量（决定最近的群组

* Single linkage：两个类中距离最近的样本距离作为两个集合的距离
* Complete linkage：取两个集合中距离最远的两个点的距离作为两个集合的距离
* Average linkage 把两个集合中的点两两的距离全部放在一起求均值，相对也能得到合适一点的结果。但是异常点的存在会影响均值。
* Centroid：取两个集合的重心的距离

欧几里得情况：Centroid= 数据的平均值，用于表示多点的聚类，并以centroid的距离表示集群的距离。

非欧几里得情况？Clustroid=数据点与其他店最接近。Closest Point 与其他点的距离（？）最近的点表示聚类。

簇间的距离？任意两点间距离的最小值。

选择集群的***内聚性***概念：cohesion的度量选择

* 1使用合并后集群的直径 = 集群中点之间的最大距离
* 点之间的平均距离
* 基于密度的方法，取直径或平均距离 然后 除以 聚类中的点的数量

层次聚类的实现

在每一步，计算所有集群对之间的两两距离，然后合并 时间复杂度 ^3

使用优先队列小心实现，将时间减少到 ^2 log

其他层次聚类方法

聚类方法的主要缺点：扩展性不好，时间复杂度至少为n2,无法撤销之前所做的操作。

使用基于距离的聚类进行层次划分：

BIRCH：使用CF树逐步调整自聚类的质量。CURE：从集群中选择分散良好的点，然后将他们缩小到集群中心的指定的分数。

8 推荐系统

From scarcity to Abundance 稀缺-丰富

模型：

u : X \* S -> R

X为顾客集，S为商品集，R为set of ratings，完全有序

Problem：

* 如何收集已知的rating？
* 从已知的推测未知的评级？要的是感兴趣的是什么，而不是喜欢什么
* 评价外推法如何衡量推荐方法的成功？

效用矩阵U是稀疏的？

推荐系统的三种实现：基于内容，协同，潜在因素为主。

Content-based 推荐系统

主要思想：向客户推荐与之前评价较高的商品相似的产品

对每个项目创建一个profile项目概要

简介是一组特征的集合

如何选择重要特征？（term频率 \* Inverse Doc频率）

优点：不需要其他用户的信息，无冷启动或稀疏问题；能够向用户推荐根据独特口味；能够推荐新的和不受欢迎的项目，没有第一评级的问题；能够提供解释。能够通过列出导致某个项目被推荐的内容特性来提供对推荐项目的解释。

缺点：找到合适的特性很难；对新用户的建议，如何建立用户档案？；过度专门化，不会推荐用户意外的用品。人们可能有多种兴趣，无法利用其他用户来判断。进行质量判断。

协同过滤

考虑用户x，找到其他与之相似的用户，基于其他用户评级估计x的评级（用户-用户协同）

项目协同过滤

利弊：

* 适用于任何类型的物品
* 冷启动，需要足够的用户在系统中找到匹配稀疏性
* 稀疏性：难以找到相同的用户
* 第一个评级：不能推荐没有评分的内容
* 流行偏见：倾向于推荐流行商品，品味独特的完蛋

评估预测？