数据挖掘学习笔记

这里写目录标题

一、NoSQL

1、NoSQL兴起原因

关系型数据库已经无法满足大数据的需求：

无法满足海量数据的管理需求
无法满足数据高并发的需求
无法满足高可扩展性和高可用性的需求

2、NoSQL四大类型

	键值数据库	列族数据库	文档数据库	图形数据库
数据模型	键值对 键是一个字符串对象 值可以是任意数据类型	以列簇式存储 将同一列数据存在一起	键值对 值是版本化的文档	图结构
优点	查找快、大量写操作时性能高	查找速度快，可扩展性强 容易进行分布式扩展，复杂性低	高并发 可以通过键或内容来构建索引	支持复杂度图形算法，可构造复杂的关系图谱
缺点	数据无结构，无法存储结构化信息 条件查询效率低 不能通过两个及以上的键来关联数据	大多不支持事务一致性	没有统一的查询语法 不适合有关联的事务处理	复杂性高，只支持一定的数据规模
用途	涉及频繁读写、拥有简单数据模型的应用 如：内存缓存、购物车	分布式数据存储与管理 可以容忍副本短期内不一致的应用	存储、索引并管理面向文档的数据或类似的半结构化数据	专门用于处理具有高度相关关系的数据
代表性数据库	redis	HBase	mongoDB	Neo4j

二、数据处理

1、数据预处理

目的

数据分析算法的设计与选择要考虑被处理数据的特征数据质量过低或数据的心态不符合算法需求时，需要进行数据预处理工作

概念

对数据进行正式计算前，根据后续数据计算的需求，对元 数据集　进行审计、清洗、变换、集成、脱敏、规约和标注等一系列活动，提升数据质量，使数据形态更符合某一算法的要求

2、数据审计

数据审计是指按照数据质量的一般规律与评价方法对数据内容及其元数据进行审计，发现其中存在的问题（缺失、噪声、矛盾、被篡改）。

3、数据清洗

在数据审计基础上，将“脏数据”清洗成“干净数据”的过程

缺失数据：识别、分析、处理
冗余数据：识别、分析、过滤
噪声数据【分箱】：分箱、计算箱特征值、替换箱内成员、生成新数据集
噪声数据【聚类 k-means　】：随机选K个对象，每个对象初始代表一个类的平均值或中心，对剩余每个对象，根据其到类中心的距离，被划分到最近的类；然后重新计算每个类的平均值。不断重复这个过程，直到所有的样本都不能再分配为止。
噪声数据【回归】：用函数拟合数据

4、数据变换

平滑处理

（分箱、聚类、回归）去除噪声数据

特征构造

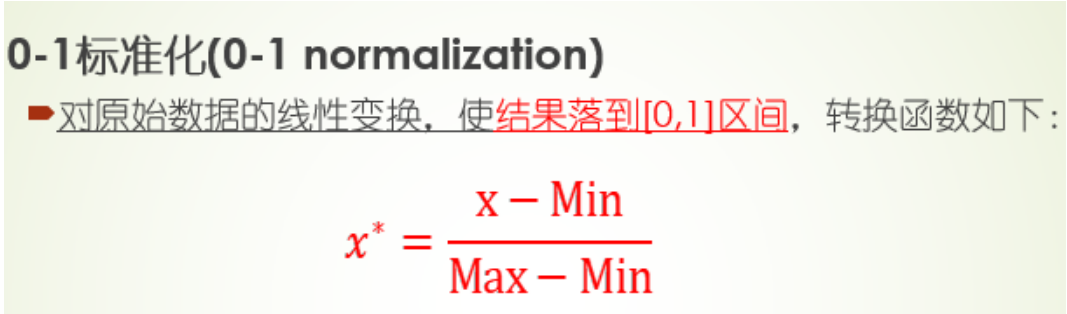
采用一致的特征构造出新的属性（如，质量、体积→密度）

聚集

对数据汇总（如，根据日销额统计月销额）

标准化

将特征值按比例缩放，落入特定区间



数据泛化

使用概念标签/概念分层

5、数据集成

对不同的数据源进行集成，并在集成后的数据集上进行数据处理

内容集成：目标数据集与来源数据集结构相同，进行合并
结构集成：目标数据集与来源数据集结构不同，进行自然连接

6、数据脱敏

保护主体的信息安全隐患和个人隐私风险

7、数据规约

在不影响数据的完整性和数据分析结果的正确性的前提下，通过减少数据规模的方法达到提升分析的效果和目的

8、 数据结构 模式

结构化数据模式

【先把模式定好，如数据库的表，需要预先设计好】

即行数据, 存储在数据库里,可以用二维表结构来逻辑表达实现的数据

半结构化数据模式

【先把数据放好，再加标记】

HTML文档就属于半结构化数据。它一般是自描述的，数据的结构和内容混在一起，没有明显的区分

无结构化数据模式

【单个Media本身是半结构化的数据】

包括所有格式的办公文档、文本、图片等

三、 聚类算法

1、KNN

基本思路

寻找最近的k个邻居

算法描述

当一个样本在特征空间中的k个最相邻的样本中的大多数属于某一类别时，该样本也属于这个类别，并具有这个类别上样本的特性

适用范围

对于类域的交叉或重叠较多的待分样本集来说，KNN较其他方法更为合适

2、K均值

步骤

第1步：从n个数据对象中随机选k个作为初始聚类中心
第2步：计算除了k个初始中心外的对象与中心对象的距离，并根据最小距离进行划分
第3步：重新计算每个有变化的聚类的均值，确定新的聚类中心
第4步：重复2-3步，知道每个聚类不再发生变化或小于指定阈值，结束算法

四、 Apriori算法

1、Apriori定律1

如果一个集合是频繁项集，则它的所有子集都是频繁项集。

举例：假设一个集合{A,B}是频繁项集，即A、B同时出现在一条记录的次数大于等于最小支持度min_support，则它的子集{A},{B}出现次数必定大于等于min_support，即它的子集都是频繁项集。

2、Apriori定律2

如果一个集合不是频繁项集，则它的所有超集都不是频繁项集

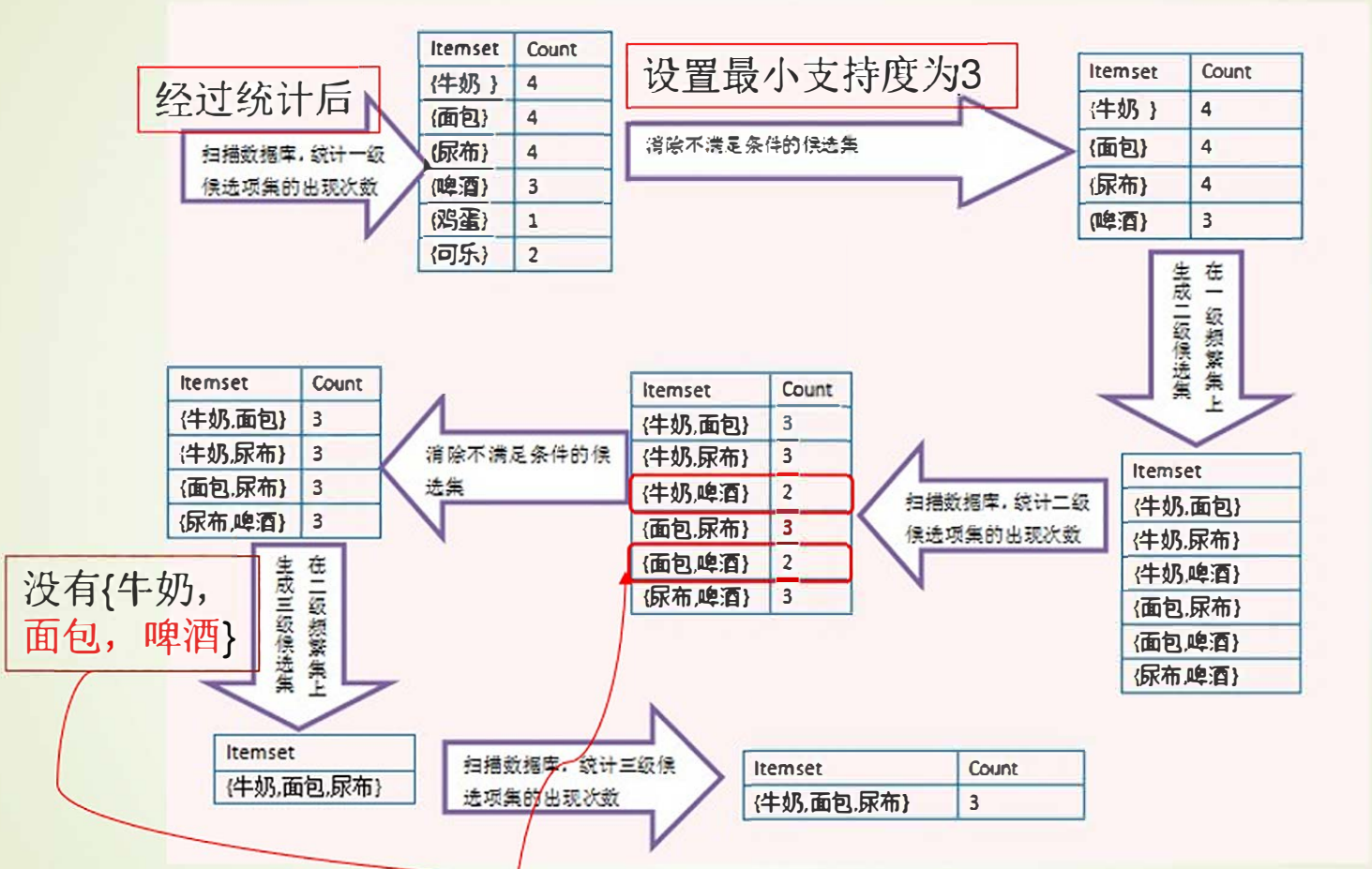
举例：假设集合{A}不是频繁项集，即A出现的次数小于min_support，则它的任何超集如{A,B}出现的次数必定小于min_support，因此其超集必定也不是频繁项集

3、Apriori算法

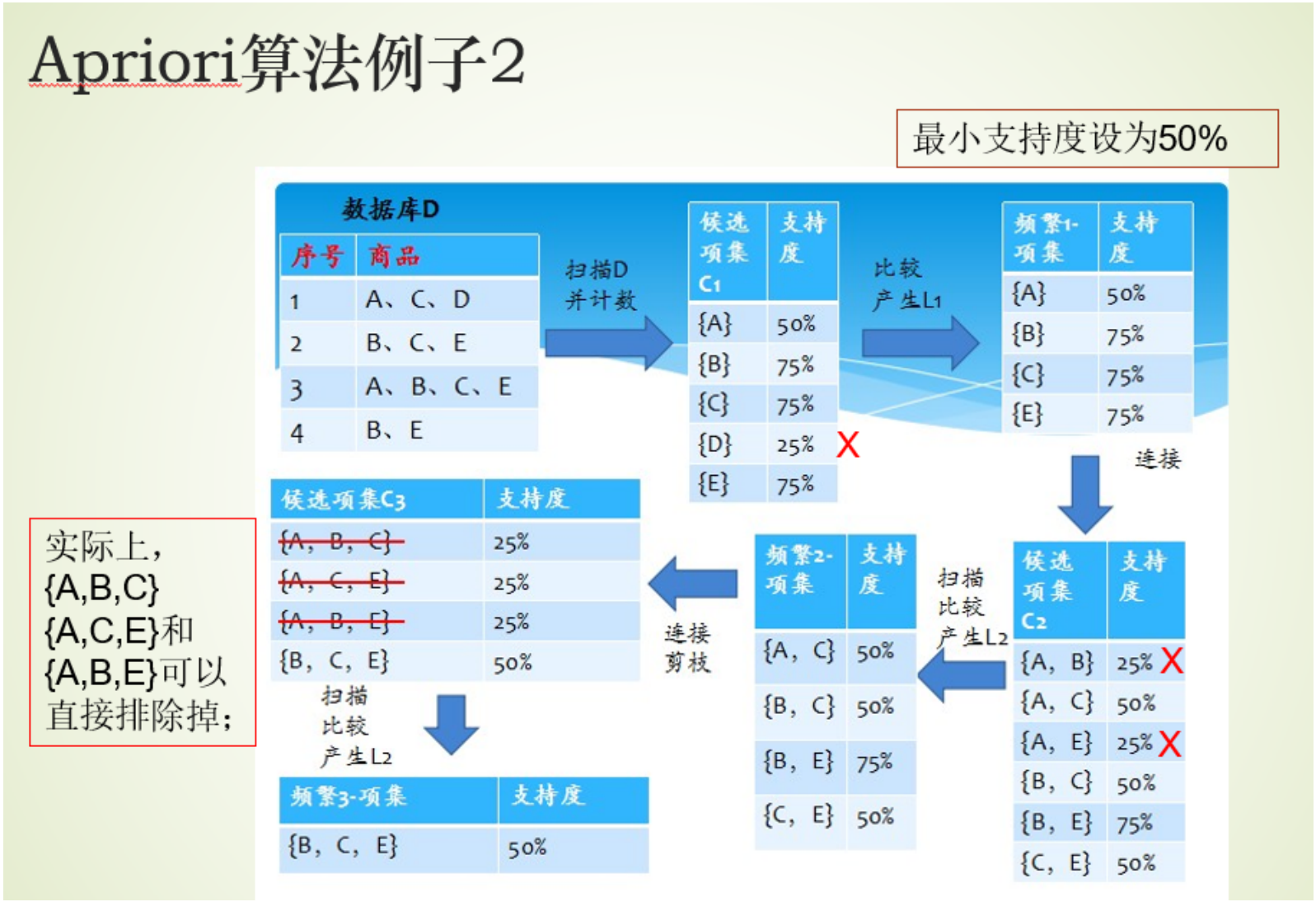
算法描述

首先，找出频繁“1项集”的集合，该集合记为 L 1 。用 L 1 找出频繁“2项集”的集合 L 2，继而用 L 2找 L 3如此下去，直到不能找到“K项集”【不能发现更大的频集】。（因此找每个 L i都要一次数据库扫描）
举例

Apriori算法例子1



注意看由二级频繁项集生成三级候选项集时，没有{牛奶,面包,啤酒}，那是因为{面包,啤酒}不是二级频繁项集，这里利用了Apriori定理



五、 机器学习

1、训练经验的选择

注意点

训练经验能否为系统的决策提供直接或间接的反馈
直接反馈：告诉你某种做法好还是不好
间接反馈：告诉你过程和结果，自行分析好还是不好
训练经验能否被学习系统控制
训练集的分布是否与实际数据集具有相似的分布

2、目标函数的选择

在许多实际问题的结果过程中，学习目标函数(T)是一个十分困难的任务，无法找到准确的目标函数(T)。 因此，一般采用函数逼近的方法，仅希望学习到一个近似的目标函数V。所以，学习目标函数的算法通常称为函数近似算法。

近似函数的设计应避免采用“不可操作的方法”：也就是理论上是可行的，但实现起来特别困难或不符合实际任务需要的方法。（比如穷举法）

目标函数的表示需要综合考虑目标函数的表达能力（选择什么样的目标函数适合具体的应用场景，如是线性、还是多项式函数等等）和训练数据的规模，进而确定近似函数的各个参数，达到表示目标函数的目的。

六、强化学习

概念

强化学习，也叫奖惩式学习。只对实际输出给出评价，而不给出正确的输出值。

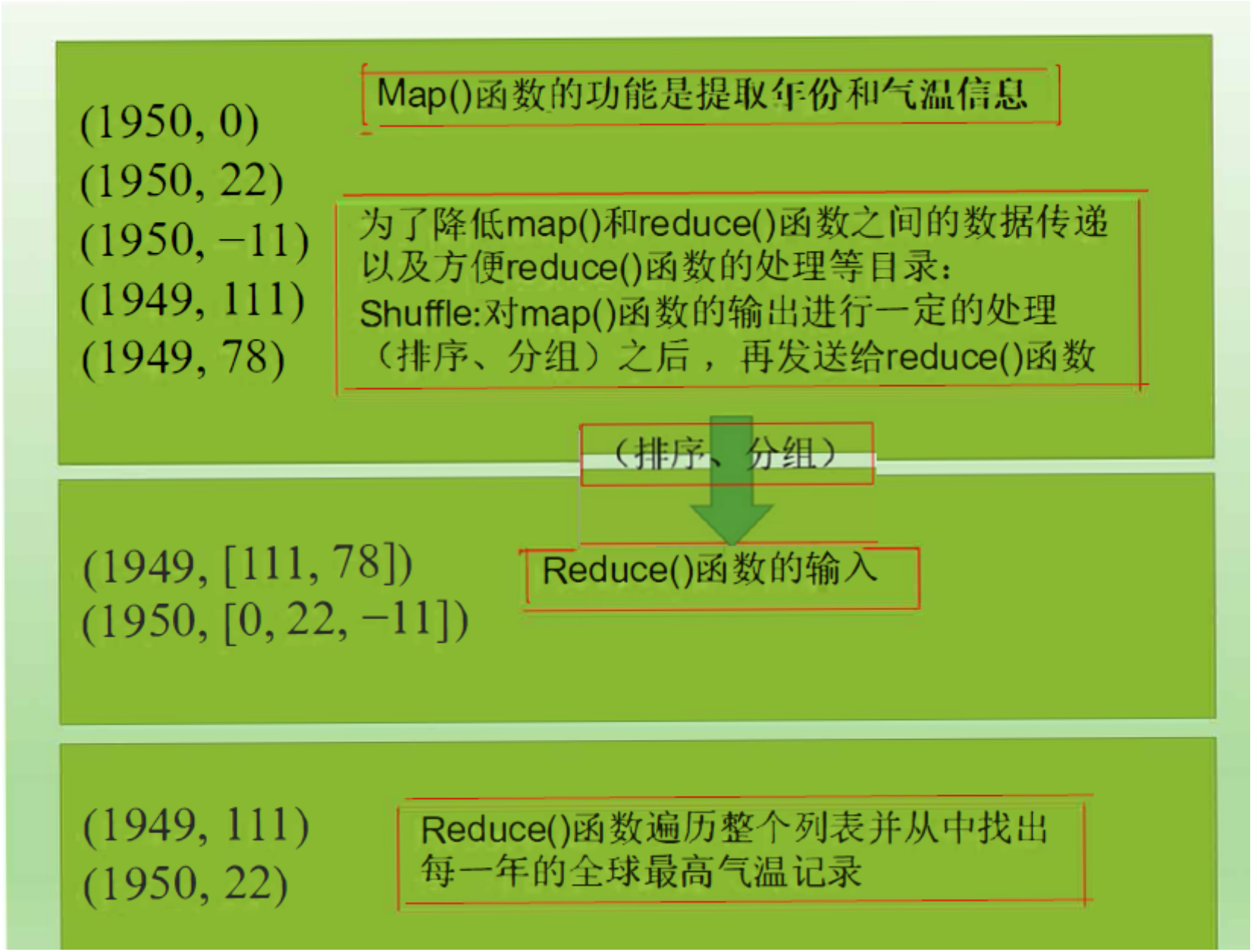
大数据的特征

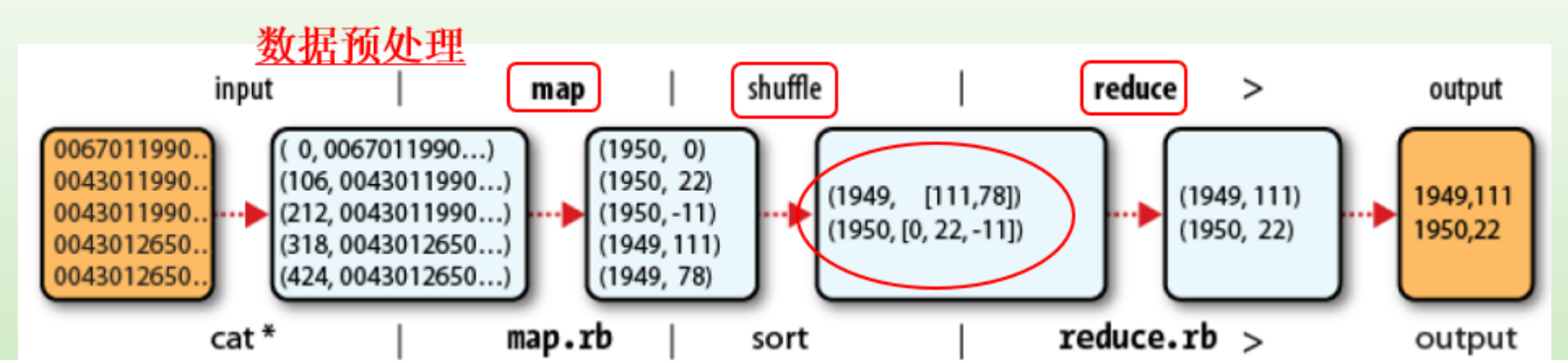
大数据的5V特点：Volume（大量）、Velocity（高速）、Variety（多样）、Value（价值）、Veracity（真实性）。

七、 MapReduce

介绍

MapReduce计算框架源自一种分布式计算模型，其输入和输出值均为<key, value>键/值对，其计算过程分为两个阶段——map阶段和reduce阶段，并分别以两个函数map（）和reduce（）进行抽象。
举例

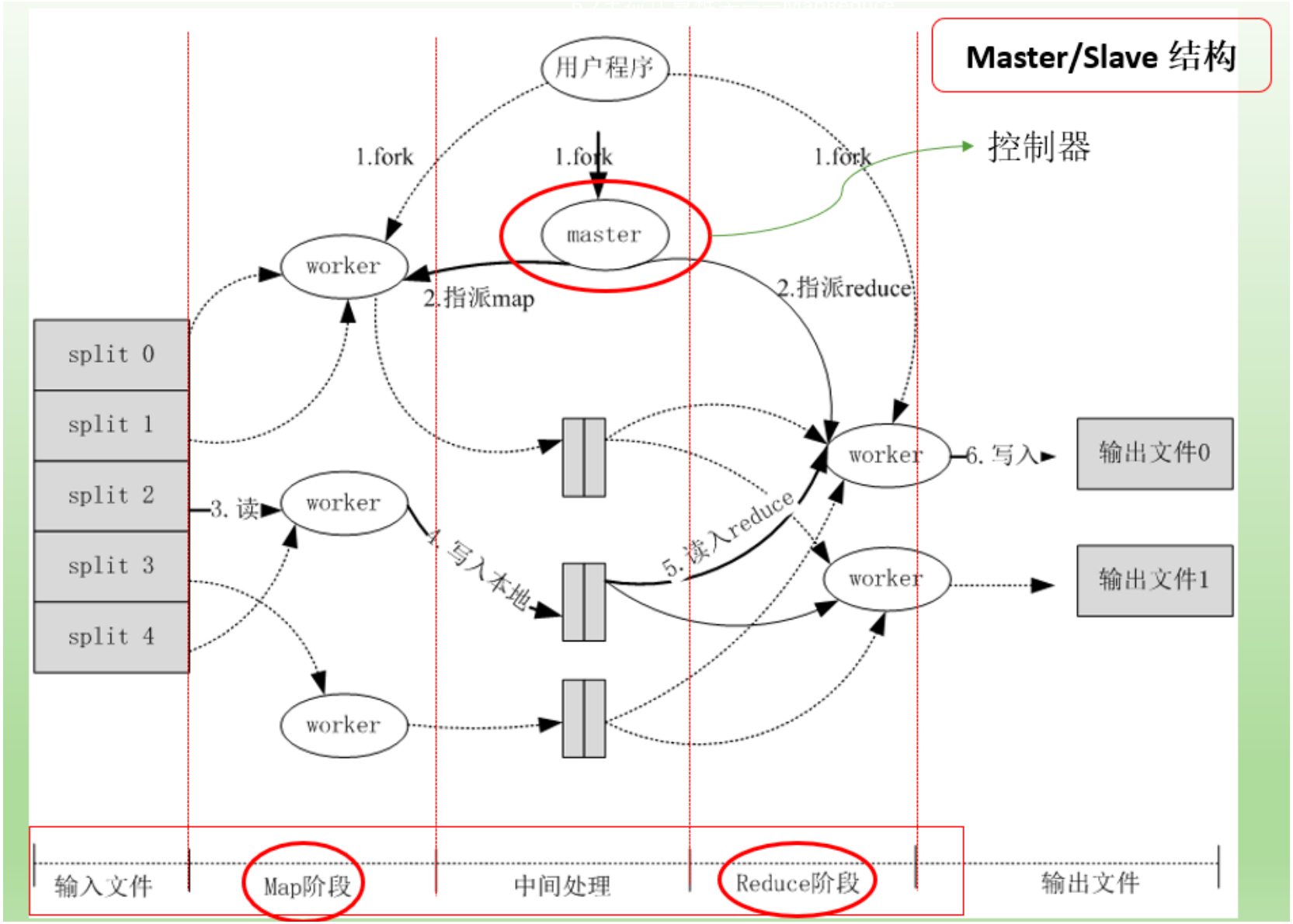

(1950, 0)
(1950, 22)
(1950, −11)
(1949, 111)
(1949, 78)
Map()函数的功能是提取年份和气温信息
为了降低map()和reduce()函数之间的数据传递以及方便reduce()函数的处理等目录：
Shuffle:对map()函数的输出进行一定的处理（排序、分组）之后，再发送给reduce()函数
（排序、分组）
(1949, [111, 78])
(1950, [0, 22, −11])
Reduce()函数的输入
(1949, 111)
(1950, 22)
Reduce()函数遍历整个列表并从中找出每一年的全球最高气温记录



计算框架原理

首先将输入文件分为M个数据块，每个数据块的大小一般为16MB~64MB.

然后在计算机集群中启动大量复制程序



应用

MapReduce这套分布式处理框架常用于：

分布式Grep
URL访问频率统计
逆向Web-Link图
主机关键向量指标
逆序索引
分布式排序
总的来说，就是适合大规模键值对数据的简单逻辑的分析

特征

主从结构
map、reduce两个函数的数据处理
键值对输入输出
容错机制复杂
数据存储位置多样
任务粒度大小重要
任务备份机制必要

八、 Spark

介绍

Spark是一个快速且通用的集群计算平台

特点

运行速度快：DAG执行引擎支持循环数据流和内存计算
容易使用：支持多种编程语言编程
通用性
运行模式多样：可以运行于独立的集群模式中，可以运行于Hadoop中，……

九、推荐系统

1、冷启动 问题

介绍

冷启动问题：如何在没有大量用户数据的情况下设计个性化推荐系统并让用户对推荐结果满意而愿意使用推荐系统

几种推荐系统是否存在冷启动问题

基于人口统计学的推荐：无冷启动问题
基于内容的推荐：有冷启动问题
基于 协同过滤 的推荐：对新物品和新用户都有冷启动问题

十、协同过滤

1、UserCF算法和ItemCF算法对比

UserCF算法推荐的是那些和目标用户有着共同兴趣爱好的其他用户所喜欢的物品

ItemCF算法推荐的是那些和目标用户之前喜欢的物品类似的其他物品

UserCF算法的推荐更偏向于社会化，而ItemCF算法的推荐更偏向于个性化

	UserCF	ItemCF
性能	适用于用户较少的场合，如果用户很多，计算用户相似度矩阵代价很大	适用于物品数明显小于用户数的场合，如果物品很多（网页），计算物品相似度矩阵代价很大
领域	时效性较强，用户个性化兴趣不太明显的领域	长尾物品丰富，用户个性化需求强烈的领域
实时性	用户有新行为，不一定造成推荐结果的立即变化	用户有新行为，一定会导致推荐结果的实时变化
冷启动	在新用户对很少的物品产生行为后，不能立即对他进行个性化推荐，因为用户相似度表是每隔一段时间离线计算的	新用户只要对一个物品产生行为，就可以给他推荐和该物品相关的其他物品
	新物品上线后一段时间，一旦有用户对物品产生行为，就可以将新物品推荐给和对它产生行为的用户兴趣相似的其他用户	但没有办法在不离线更新物品相似度表的情况下将新物品推荐给用户
推荐理由	很难提供令用户信服的推荐解释	利用用户的历史行为给用户做推荐解释，可以令用户比较信服

2、协同过滤的推荐机制总结

基于协同过滤的推荐机制是现今应用最为广泛的推荐机制，它有以下几个显著的优点：

它不需要对物品或者用户进行严格的建模，不要求物品的描述是机器可理解的，这种方法也是领域无关的
可以共用他人的经验，很好的支持用户发现潜在的兴趣偏好
基于协同过滤的推荐机制也存在以下几个问题

方法的核心是基于历史数据，所以对新物品和新用户都有“冷启动”的问题
推荐的效果依赖于用户历史偏好数据的多少和准确性