<<大数据>>

图书基本信息

书名：<<大数据>>

13位ISBN编号：9787115291318

10位ISBN编号：7115291314

出版时间：2012-9

出版时间：人民邮电出版社

作者：Anand Rajaraman,Jeffrey David Ullman

译者：王斌

前言

本书是在Anand Rajaraman和Jeff Ullman于斯坦福大学教授多年的一门季度课程的材料基础上总结而成的。
该课程名为“Web挖掘”（编号CS345A），尽管它已经成为高年级本科生能接受并感兴趣的课程之一，但其原本是一门为高年级研究生设计的课程。
本书内容简单来说，本书是关于数据挖掘的。
但是，本书主要关注极大规模数据的挖掘，也就是说这些数据大到无法在内存中存放。
由于重点强调数据的规模，所以本书的例子大都来自Web本身或者Web上导出的数据。
另外，本书从算法的角度来看待数据挖掘，即数据挖掘是将算法应用于数据，而不是使用数据来“训练”某种类型的机器学习引擎。
本书的主要内容包括：（1） 分布式文件系统以及已成功应用于大规模数据集并行算法构建的Map-Reduce工具；（2） 相似性搜索，包括最小哈希和局部敏感哈希的关键技术；（3） 数据流处理以及面对快速到达、须立即处理、易丢失的数据的专用处理算法；（4） 搜索引擎技术，包括谷歌的PageRank、链接作弊检测及计算网页导航度（hub）和权威度（authority）的HITS方法；（5） 频繁项集挖掘，包括关联规则挖掘、购物篮分析、A-Priori及其改进算法；（6） 大规模高维数据集的聚类算法；（7） Web应用中的两个关键问题：广告管理及推荐系统。
先修课程尽管从编号CS345A看，本课程属于高年级研究生课程，但是我们发现高年级本科生和低年级硕士生也能接受该课程。
该课程将来可能会分配一个介于高年级研究生和低年级硕士生水平之间的编号。
CS345A的先修课程包括：（1） 数据库系统的首期课程，包括基于SQL及其他数据库相关语言（如XQuery）的应用编程；（2） 大二的数据结构、算法及离散数学课程；（3） 大二的软件系统、软件工程及编程语言课程。
习题本书包含大量的习题，基本每节都有对应习题。
较难的习题或其中较难的部分都用惊叹号“！”来标记，而最难的习题则标有双惊叹号“！！”。
致谢本书封面由Scott Ullman设计。
感谢Foto Afrati和Arun Marathe精心阅读本书初稿并提出建设性的意见。
感谢Leland Chen、Shrey Gupta、Xie Ke、Haewoon Kwak、Brad Penoff、Philips Kokoh Prasetyo、Mark Storus、Tim Triche Jr.及Roshan Sumbaly指出了本书中的部分错误。
当然，剩余错误均由我们负责。
A. R.J. D. U.加利福尼亚州帕洛阿尔托2011年6月

<<大数据>>

内容概要

《大数据：互联网大规模数据挖掘与分布式处理》源自作者在斯坦福大学教授多年的“Web挖掘”课程材料，主要关注大数据环境下数据挖掘的实际算法。
书中分析了海量数据集数据挖掘常用的算法，介绍了目前Web应用的许多重要话题。
主要内容包括：分布式文件系统以及Map-Reduce工具；相似性搜索；数据流处理以及针对易丢失数据等特殊情况的专用处理算法；搜索引擎技术，如谷歌的PageRank；频繁项集挖掘；大规模高维数据集的聚类算法；Web应用中的关键问题：广告管理和推荐系统。

作者简介

Jeffrey David Ullman 斯坦福大学计算机科学系Stanford W. Ascherman教授，数据库技术专家。
他独立或与人合作出版了15本著作，发表了170多篇技术论文。
他的研究兴趣包括数据库理论、数据库集成、数据挖掘和利用信息基础设施进行教育。
他是美国国家工程院成员，曾获得Knuth奖、SIGMOD贡献奖、Karlstrom杰出教育家奖和Edgar F. Codd发明奖。
Anand Rajaraman　数据库和Web技术领域权威,1创业投资基金Cambrian联合创始人,斯坦福大学计算机科学系助理教授。
Rajaraman职业生涯非常成功：1996年创办Junglee公司,两年后该公司被亚马逊以2。
5亿美元收购,Rajaraman被聘为亚马逊技术总监,推动亚马逊从一个零售商转型为零售平台；2000年与人合创Cambrian,孵化出几个后来被谷歌收购的公司；2005年创办Kosmix公司并任CEO,该公司2011年被沃尔玛集团收购。
Rajaraman生于印度,在斯坦福大学获得计算机科学硕士和博士学位。
求学期间与人合著的一篇论文荣列近20年来被引用次数最多的论文之一。

书籍目录

章节摘录

版权页： 插图： 然而，当项对的数目太多而无法在内存中对所有的项对计数时，上述简单的方法就不再可行。
A-Priori算法被设计成能够减少必须计数的项对数目，当然其代价是要对数据做两遍而不是一遍扫描。
1.A-Priori算法的第一遍扫描 第一遍扫描中，我们要建立两张表。
如有必要，第一张表要将项的名称转换为1到n之间的整数（参考6.2.2节中的描述）。
另一张表则是一个计数数组，第i个数组元素是上述第i个项的出现次 数。
这些所有项的计数值的初始值都是0。
在读取购物篮时，我们检查购物篮中的每个项并将其名称转换为一个整数。
然后，将该整数作为计数数组的下标找到对应的数组元素，最后，对该数组元素加1。
2.A-Priori算法两遍扫描之间的处理 第一遍扫描之后，我们检查所有项的计数值，以确定哪些项构成单元素频繁项集。
我们可能会看到，大部分单元素项集都是不频繁的。
这一点可能会有点出人意料。
但是，前面提到，我们常常将阈值s设置得足够高以保证频繁集不会太多。
一个典型的s值为所有购物篮数目的1%。
想象一下自己到超市购物的情况，我们购买某些商品的次数肯定会超过总次数的1%，这些商品可能是牛奶、面包、可口可乐或百事可乐什么的。
我们甚至相信，虽然我们不购买尿布，但是会有1%的顾客会购买尿布。
然而，货架上的大部分商品的顾客购买比例肯定都不会超过1%，比如奶油凯撒沙拉汁。
对于A-Priori算法的第二遍扫描，我们会只给频繁项重新编号，编号范围是1到m。
此时的表格是一个下标为1到n的数组，如果第i项不频繁，则对应的第IAI数组元素为0，否则为1到m之间的一个唯一整数。
我们应将此表格称为频繁项表格。
3.A-Priori算法的第二遍扫描 在第二遍扫描中，我们对两个频繁项组成的所有项对计数。
从6.2.3节的讨论可知，除非一个项对中的两个项都频繁，否则这个项对也不可能是频繁的。
因此，在扫描过程中我们不可能会丢掉任何频繁项对。
如果采用前面提到的三角矩阵方法来计数的话，则第二遍扫描所需的空间是2n2而不是2n2。
需要注意的是，如果要使用一个大小正确的三角矩阵，那么就一定要只对频繁项进行重新编号处理。
第一遍和第二遍扫描中所使用的完整内存结构集合如图6-3所示。
需要注意的另外一点是，上述非频繁项去除的好处会被放大：如果只有一半的项是频繁项，那么在计数过程中仅需要原来空间的1/4。
类似地，如果使用三元组方式，我们只需要对至少出现在一个购物篮中的两个频繁项组成的项对进行计数。
第二遍扫描的技术细节如下：（1）对每个购物篮，在频繁项集表中检查哪些项是频繁的；（2）通过一个双重循环生成所有的频繁项对；（3）对每个频繁项对，在存储计数值的数据结构中相应的计数值上加1；最后，在第二遍扫描结束时，检查计数值结构以确定哪些项对是频繁项对。

编辑推荐

版权说明