第2章

大数据分析方法

本章目标

- 掌握大数据的处理流程：数据采集、预处理、存储、挖掘和解释
- 掌握大数据的3种来源：核心数据、外围数据、常规渠道数据
- 掌握大数据的主要架构
- 掌握数据挖掘常用方法

本章简介

本章将从大数据处理流程、数据来源、大数据生态圈及主要架构、数据挖掘的主要方法这几个方面来介绍大数据技术的分析方法。

2.1 大数据处理流程

大数据的处理流程为：首先，利用多种轻型数据库收集海量数据，对不同来源的数据进行预处理，并整合存储到大型数据库中；其次，根据企业或个人目的和需求，运用合适的数据挖掘技术提取有益的知识；最后，利用恰当的方式将结果展现给终端用户。具体流程包括数据采集、数据预处理、数据存储、数据挖掘及数据解释5个步骤，如图2.1所示。

图2.1 大数据的处理流程

2.1.1 数据采集

大数据的采集是大数据处理的第一步，它是数据分析和挖掘的基础。大数据的采集是指在确定用户目标的基础上，对该范围内的所有结构化、半结构化、非结构化数据进行采集的过程。采集的数据大部分是瞬时值，也包括某时段内的特征值。大数据的主要来源有商业数据、互联网数据和传感器数据。针对不同来源的数据，运用不同的采集方法。主要的大数据采集方法有系统日志采集方法、网络数据采集方法和其他数据采集方法。

1. 系统日志采集方法

大多数互联网企业都有自己的海量数据采集工具，常用于系统日志采集，如Scribe、Flume、Chukwa、Kafka等。Scribe是Facebook的开源日志收集系统，能够从各种日志源收集日志，存储到一个中央存储系统中，以便进行集中统计分析和处理；Chukwa属于Hadoop系列产品，是一个大型的分布式系统监测数据的收集系统，提供了很多模块以支持Hadoop集群分析；Flume是cloudera的开源日志系统，能够有效地收集汇总和移动大量的实时日志数据。这些工具均采用分布式架构，能满足每秒数百MB的日志数据采集和传输需求。

2. 网络数据采集方法

网络数据采集是指利用互联网搜索引擎技术从网站抓取数据信息。目前，网络数据的采集基本上是利用垂直搜索引擎技术的网络爬虫或数据采集机器人、分词系统、任务与索引系统等技术综合运用而完成。该方法可以将非结构化数据从网页中抽取出来，将其存储为统一的本地数据文件，并以结构化的方式存储。它支持图片、音频、视频等文件或附件的采集，附件与正文可以自动关联。除了网络包含的内容之外，对于网络流量的采集可以

使用 DPI 或 DFI 等带宽管理技术进行处理。

3. 其他数据采集方法

对于企业生产经营数据或学科研究数据等保密性要求较高的数据，可以通过与企业或研究机构合作，使用特定系统接口等相关方式采集数据。

在大数据的采集过程中，同一网站同一时间可能会有很多用户访问和操作。例如，火车票售票网站和淘宝，它们并发的访问量在峰值时超过了上百万，并发数极高。因此，需要在采集端部署大量数据库支撑。

2.1.2 数据预处理

第一步收集到的数据是原始数据，存在不完整、不一致的问题，无法直接存储到数据库进行数据挖掘。因此，在将来自前端的数据导入一个集中的大型数据库或者分布式存储集群前，需要对大数据进行预处理，这样不但能够节省大量的空间和时间，还能得到更好的数据挖掘结果。大数据预处理包括对数据进行清理、集成、变换和归约 4 个过程。

1. 数据清理

数据清理是数据准备过程中最乏味，也是最关键的一步。其目的是补充缺失的数据、平滑噪声数据、删除冗余数据、纠正错误数据、清除异常数据，将原始的数据格式进行标准化。

2. 数据集成

数据集成是将多个数据源中的数据结合起来并统一存储，建立数据仓库，以更好地解决数据的分布性和异构性问题。数据集成技术的关键设备是数据高速缓存器。拥有一个包含目标计划、源—目标映射、数据获取、分级抽取、错误恢复和安全性转换的数据高速缓存器，可以大大降低直接访问后端系统和进行复杂实时集成的需求。

3. 数据变换

数据变换是采用线性或非线性的数学变换方法将多维数据压缩成较少维数的数据，消除它们在时间、空间、属性、精度等特征表现方面的差异。数据变换可用相当少的变量捕获原始数据的最大变化，具体变换方法的选择可根据实际数据的属性特点而定，常见的数据变换方法有数据平滑、数据聚焦、数据规范化等。

4. 数据归约

数据归约是指在对数据挖掘任务和数据本身内容理解的基础上寻找依赖于发现目标的数据的有用特征，以缩减数据规模，从而在尽可能保持数据原貌的前提下，最大限度地精减数据量。数据归约主要有两个途径：属性选择和数据采样，分别针对原始数据集中的属性和记录。数据归约技术可以用来得到数据集的归约表示，它虽然小，但仍然大致保持原始数据的完整性。这样，在归约后的数据集基础上挖掘将更有效，并能产生相同(或几乎相同)的分析结果。数据归约的类型主要有特征归约、样本归约和特征值归约。

2.1.3 数据存储

大数据种类繁多，数据结构化程度不同，传统的结构化数据库无法满足大数据的存储要求。下面介绍3种典型的大数据存储方案，分布式文件系统、分布式数据库和云存储。

1. 分布式文件系统

分布式文件系统是指文件系统管理的物理存储资源不一定直接连接在本地节点上，而是通过计算机网络与节点相连，众多的节点组成一个文件系统网络；每个节点可以分布在不同的地点，通过网络进行节点的通信和数据传输。常见的分布式文件系统有 GFS、HDFS、Lustre、Ceph 等，它们各自适用于不同的领域，其中 GFS 和 HDFS 最具有代表性。GFS 是 Google 公司设计的专用文件系统，主要用于存储海量搜索数据，处理大文件。HDFS 是 Hadoop 分布式文件系统，它是一种被设计成适合运行在通用硬件上的分布式文件系统，具有高容错性。

2. 分布式数据库

分布式数据库是利用网络将物理上分布的多个数据存储单元连接起来组成的逻辑数据库，其目的是将集中式数据库中的数据分散存储到多个数据存储节点上，并通过网络节点连接起来，以获取更大的存储容量和更高的并发访问量。与传统的集中式数据库相比，分布式数据库具有高扩展性、高并发性、高可用性以及更高的数据访问速度。近年来，随着数据量的高速增长，传统的关系型数据库开始从集中式模型向分布式架构发展，从集中式存储走向分布式存储，从集中式计算走向分布式计算。

3. 云存储

云存储是一种以数据存储和管理为核心的云计算系统，它是指利用集群应用、分布式文件和网络技术系统等功能，通过应用软件协同网络中大量的各种不同类型的存储设备，共同建设一个具有数据存储和业务访问功能的系统，以保障数据的安全性，节约存储空间。互联网技术的发展是实现云存储的基本条件。通过互联网技术，云存储实现数据、文档、图片、音频、视频等内容的存储和共享。云存储系统结构主要由存储层、基础管理层、应用接口层、访问层4个部分组成。

2.1.4 数据挖掘

数据挖掘是指根据业务的需求和目的，运用合适的工具软件和数据挖掘方法对数据仓库中的数据信息进行处理，寻找特定的数据规律或数据模式，得出有价值的信息和知识。根据信息存储格式，可以把数据挖掘的对象分为关系数据库、面向对象数据库、数据仓库、文本数据源、多媒体数据库、空间数据库、时态数据库、异质数据库以及 Internet 等。数据挖掘常用的工具软件有 Intelligent Miner、SPSS、SAS、WEKA、Matlab、R 语言、Python 等。数据挖掘的任务是从数据中发现模式，按照数据挖掘的实际作用数据挖掘任务可分为关联分析、聚类分析、分类、回归、预测、序列和偏差分析。

2.1.5 数据解释

数据解释是一个面向用户的过程，它是指将大数据挖掘及分析结果在显示终端以友好、形象、易于理解的形式呈现给用户。传统的数据解释方法是以文本形式输出结果或者直接在电脑终端上显示结果。大数据分析的结果一般数据量巨大且关系复杂，传统的分析结果展示方法已基本不适用。现阶段，主要是利用可视化技术、人机交互、数据起源等新的方法将结果展示给用户，帮助用户更加清晰地了解数据处理后的结果，为用户提供决策信息的支持。目前，大部分企业已引进数据可视化技术和人机交互技术。

1. 数据可视化技术

数据可视化技术主要是通过图形化方法进行清晰、有效的数据传递。其基本思想是使用单个图元元素表示数据库的每一个数据项，大量的数据集组成数据图像，并以多维数据的形式表示数据的各个属性值。运用可视化技术就可以将数据结果转化为静态或者动态的图形展示给用户，通过交互手段抽取或者集成数据能在画面中动态地显示改变的结果。这样，用户可以从不同的维度观察数据，对数据进行更深入的观察和分析。可视化技术可以分为 5 类：几何技术、图标技术、图形技术、分层技术和混合技术。基于不同的需求既可以采取不同的可视化技术，也可以通过多种技术手段来展示数据处理结果。例如，电力网络中电力的传输，为直观地反映各个城市的电力需求状况，可以利用基于图标技术，用不同的颜色标明图中各个城市的电力负载情况。

2. 人机交互技术

人机交互技术是指通过系统输入、输出设备，以有效的方式实现人与系统之间信息交换的技术。其中，系统可以是各类机器、计算机和软件。用户界面或人机界面是人机交互所依托的介质和对话接口，通常包括硬件和软件系统。人机交互技术是一种双向的信息传递过程，既可以由用户向系统输入信息，也可以由系统向用户反馈信息。通过人机交互技术，用户只需要通过输入设备给系统输入有关信息、提示、请示等，系统就会输出或通过显示设备提供相关信息、回答问题等。人机交互技术能够使大数据分析的数据结果被更好地解释给用户。这种交互式的数据分析过程可以引导用户逐步地进行分析，使用户在得到结果的同时能够更好地理解分析结果的由来。有同种作用的还有数据起源技术，通过该技术可以帮助用户追溯整个数据分析的过程，从而有助于用户理解结果。

@ 2.2 数据来源

要做大数据，首先，要了解自己的企业，或者自己所在行业的核心是什么。也就是说，最关键的企业需要找到自己的核心数据(价值)。只有在这个前提下，建立自己的大数据才能做一些延伸。其次，要找到内部的一些外围相关数据，慢慢地成长它。第一层是核心；第二层是外围相关的数据；第三层是外部机构的一些结构化数据；第四层是社会化的以及各种现在所谓的非结构化的数据。要做好找外围相关数据工作，需要做好以下四步。

第一步，找到核心数据，核心数据现在对很多企业来说就是CRM——自己的用户系统，这是最重要的。第二步，找到外围数据，通过营销活动等获取大量数据。第三步，找到常规渠道的数据，这就需要企业去找常规渠道打下的数据，跟自己的CRM结合起来，为下一步做市场营销、做推广、产品创新等打下基础。第四步，找到外部的社会化或者非结构化的数据，即现在所谓的社会化媒体数据。这方面信息的主要特征是非结构化，而且数量非常庞大。

下面以金融企业为例，重点讨论金融企业的数据来源、数据现状，企业存在哪些问题以及应该怎么应对。

2.2.1 核心数据

1. 现状

金融企业的核心数据主要有以下5个来源，如图2.2所示。

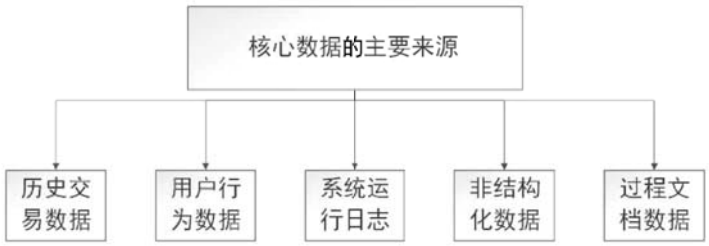

图2.2 数据来源

1) 历史交易数据

按照主数据的普遍规划来划分，金融企业一般拥有客户数据、交易数据、账户数据等，这些数据有一些已沉淀了多年，伴随着当年的一些金融产品进入数据库，正处于生命周期的某一阶段。这些数据极具潜力价值，通常可以用来促进精确营销、优化产品设计等分析项目。

2) 用户行为数据

企业每天处理海量的交易，有相当一部分交易是网络上的终端客户直接发起的，特别是在一些业务促销活动过程中。因此，柜员服务系统、网上服务系统产生了大量的业务行为轨迹，这些数据通常用来分析如何提高运营效率，促进精准营销。

3) 系统运行日志

金融企业的应用系统数量较多，分别负责完成各个子领域的业务处理与管理决策。这些应用系统会产生大量的数据库日志和应用程序日志。在日常维护中，这些日志的数量很大、价值密度低，并不受重视。实际上，通过日志分析应用系统效率，是提高应用系统服务水平和客户满意度的重要方法。

4) 非结构化数据

金融企业普遍经济实力雄厚，在众多基础设施建设中投入了巨资。因此，通过大规模的语音呼叫中心、邮件中心、短信中心等客户接触渠道，金融企业拥有发布和采集数据的主动权。另外，不少金融单位有着遍布全国的客户服务大厅，安装了先进的视频监控系

统，视频数据既能起到安全防范作用，也能用于分析客户时长等服务类指标。

5) 过程文档数据

金融企业通常成立了大规模的研发中心和数据中心，按照标准的流程开发和部署应用系统。在这个过程中，产生大量的需求分析、设计文档、测试报告、上线部署、问题记录等过程和技术文档。这些文档是分析和提升服务水平的重要支撑。

2. 问题

核心数据最大的问题在于来源多样、流动性差、共享性差。

1) 数据质量问题

某些应用系统开发历史较久，随着架构规划和科学技术的不断进步，出现接口数量多、数据不一致、数据质量差等问题，难以进行大数据分析。

2) 内部管理“壁垒”

金融行业在开展大数据项目获取数据时，最严重的问题是内部管理“壁垒”。对许多企业来说，信息流被各部门彼此分割，数据难以互通，在这种情况下，大数据的共享和汇集就变得非常困难，更难以实现大数据的深度应用。

3. 解决方法

数据作为一项资产，部门之间存在数据“壁垒”的问题，不是各部门造成的，而是公司在数据职责、权利的定位方面出现了偏差。

因此，解决此问题需要通过以下3个途径。

(1) 明确数据相关的职责与归属。金融企业要明确各个渠道和部门拥有的是数据采集职责，为公司增加数据资产；数据资产的所有权与使用权，只能归公司所有。

(2) 提升对数据资产质量的认识。数据资产至关重要，不少金融企业依靠销售渠道或者第三方平台开展销售，若客户资料质量很差或者根本无法获取，就相当于向公司提供了伪劣的数据资产。

(3) 打通数据流转。金融行业有独立的研发中心和数据中心。其中，研发中心负责程序的开发，不得接触生产数据以及未脱敏的测试数据；数据中心负责程序的部署，不得接触程序源码。应用系统研发与生产的剥离可能会加大大数据实施的难度。在大数据这项需要创新与试错的任务面前，数据中心作为数据的实际保有者，往往不愿意向具有创新能力的研发中心提供数据。因此，对大数据应用来说，要确定真正具有创新实践能力的组织架构，并从决策管理层明确所需的各类支持必须到位，确立一定的考核与激励措施，做到利益均衡、成果共享。

2.2.2 外围数据

1. 外围数据的基本准则

(1) 符合法律规定，遵循道德规范。这是一项基本要求。

(2) 在使用外围数据前，要清楚提供者的商业模式，如果提供者的商业模式会给本企业的未来带来竞争关系，那么合作时就需要仔细商榷。

(3) 要在购买与交换之间权衡利弊。在数据所有权不清晰的情况下，交换数据是一种合作举措，可以看作两家单位在以客户为中心的目标下开展的联合行为。

(4) 外部数据的目的是补充内部数据，转化为企业数据资产。如果企业已存在类似的内部数据，但因部门利益割裂的原因无法作为数据资产共享，而采用外购形式弥补，那么这些外部数据往往会变成一个新的分割独占的数据，同样不能变成企业级资产。

2. 外围数据来源

随着数据资产地位的确立，外围数据和固定资产、知识产权一样，会形成新的产业链条。不过数据资产极为特殊，它的价值会随着交换与使用而扩大，这与固定资产、货币资产存在着显著不同。另外，所有权和使用权难以界定，也大大增加了数据交易的难度与风险。

金融企业外围数据的来源如下。

1) 数据共享联盟

对大数据来说，整合和共享的价值更大。例如在医疗行业中，每家医院对于自己的数据进行分析，需要共享跨医院、跨地域的医疗信息。未来数据将呈现共享的趋势，数据联盟成为数据集散地之一。

2) 互联网数据

网络爬虫仍然是外部数据的有效获取途径，因为互联网有着最大的数据库。在进行舆情监控时，这类数据是必不可少的。另外，也可以直接和大型互联网平台进行数据交易。

3) 运营商数据

例如，在统计房屋空置率时，利用大数据根据电力局的智能电表数据、水利局的水表走数、邮局和快递公司的针对该地址的投递率、通信公司的固定电话使用率，基本能清楚哪些房屋无人居住。因此，金融企业在寻找优质企业时可以反其道而行之，挖掘客户。未来各行业更好发展的一条捷径就是客户资源共享。

3. 常见问题

1) 数据获得成本

金融企业数据是非常有价值的一类数据。数据提供商最清楚数据的价值，因此选择通过“购买加交换”的形式提供数据。金融企业需要评估可能付出的成本与代价。

2) 数据价值发挥

很多购买数据的金融企业，是由于内部数据的所有权和使用权不清晰而被迫的行为。在这种情况下，虽然购买数据可以解决某个部门的一时之需，但是这些购入的数据在使用上往往不兼容，无法最大限度地发挥数据的价值。

2.2.3 常规渠道数据

在大数据时代下，数据将发挥生产资料的作用，数据储备和数据分析能力将成为未来新兴国家最重要的核心战略能力之一。各地政府正在尝试由信息公开转向数据公开。政府开放数据侧重开放大量的、实时的、结构化的数据和信息，将其在相关业务上所收集、整

理、产生或者保有的数据与信息，主动开放给其他对象(包括社会组织与公众)进行数据创新增值应用。

尽管受格局、意识、管理水平的限制，各地各级政府的数据公开呈现发展迅速却明显不均的态势，但是金融企业应该做好准备，将公开数据资产转化为企业内部的核心竞争力。

1. 政府数据开放存在内驱动力

在所有数据来源中，政府通常掌握着最大量的、关键性的数据和公共信息资源，加大开发力度，极大地推动政府办事效率的提升和国家信息服务业的发展。

从政府对内有效管理和对外民生服务两个层面而言，降低行政成本、提高决策的科学化水平需要高效、实时的信息系统，而大数据的支持是此类信息系统有效发挥作用的支柱之一。政府提供公共服务及促进经济社会发展的职能发挥同样需要大数据支持。政府掌握了大量关于人口、法人和城市空间地理等数据，如果要提供满足群众需求、有针对性的公共服务，则需要对所掌握的数据进行精细化分析。

2. 政府公开数据的步骤

公开数据需要各级政府出台更多具有可操作性的细则和措施。首先，相应部门应制定由政府或者行业协会牵头的整合数据标准。定义政府开放数据的最小数据集，从最小数据集来控制收集、扩大开放。其次，要制定开放数据的相关法规，界定哪些数据可以开放，因为开放数据有成本，要开放那些最有用、需求量最大的数据。最后，还要加大数据开放所带来的价值分析和评估力度，研究持续开放的政策。

3. 金融行业积极参与政府数据开放的过程

首先，政府数据公开需要一整套的完整规划、顶层设计和系统建设，贯穿信息收集、整理、存储、发布、服务等全过程，内容包括信息网络、应用系统、信息的采集和发布及相关的管理体制、程序、实施模式和项目管理等。其次，政府公开数据在不同部门、不同层级、不同领域、不同行业之间的分享、交换、整合还存在很多问题，想要建成统一的数据平台，还需要做很多工作。最后，对大数据产业而言，政府公开数据的管理、整合及挖掘，也是具有广阔前景的业务发展方向。

金融行业应秉承社会和政治责任，发挥资金、网点、技术优势，积极参与到政府的数据开放的过程中，以政府为导向，帮助建立起公共数据服务平台，为自身和行业的健康有序发展发挥重要的基础作用。

@ 2.3 大数据架构

基于上述大数据的特征，通过传统 IT 技术存储和处理大数据成本高昂。一个企业要大力发展大数据应用需要解决两个问题：一是低成本、快速地对海量、多类别的数据进行抽取和存储；二是使用新的技术对数据进行分析和挖掘，为企业创造价值。因此，大数据的存储和处理与云计算技术密不可分，在当前的技术条件下，基于分布式系统的 Hadoop，被

认为是最适合处理大数据的技术平台。Hadoop 的功能是利用服务器集群，根据用户的自定义业务逻辑，对海量数据进行分布式处理。从广义上说，Hadoop 通常是指一个更广泛的概念——Hadoop 生态圈，如图 2.3 所示。

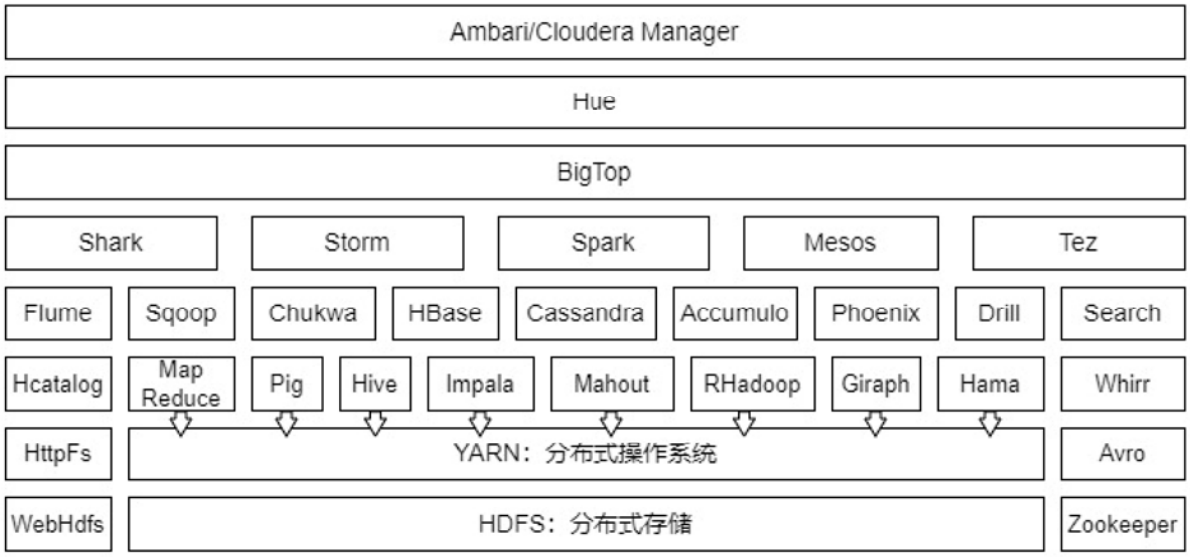

图 2.3　Hadoop 生态圈

Hadoop 生态圈各组件简介如下。

1. 主要模块

(1) HDFS：分布式文件系统。

(2) MAPREDUCE：分布式运算程序开发框架，用于大规模数据集的并行计算。

(3) HBASE：基于 Hadoop 的分布式海量数据库，可以将结构化数据文件映射为数据库表，并提供常用的 SQL 支持。Hive 查询引擎将 SQL 语句转化为 Hadoop 平台的 MapReduce 任务运行。

2. 数据管道

(1) Sqoop：主要用于跟关系数据库进行数据交互，通过 JDBC 方式实现数据迁移。

(2) Flume：Cloudera 提供的日志收集框架，用于将海量日志数据并行导入 HDFS 或者 Hive 中。

(3) DistCp：一般用于在两个 HDFS 集群中传输数据，但目前此命令只支持同版本下的集群数据迁移，主要用于冷热数据迁移、测试等场景。

(4) Scribe：Facebook 开源的日志收集系统，它能够从各种日志源上收集日志，然后存储到一个中央存储系统(可以是 NFS、分布式文件系统等)上，以便于进行集中统计分析处理。它为日志的“分布式收集，统一处理”提供了一个可扩展的、高容错的方案。

3. 数据分析

(1) Hive：提供了一套类数据库的数据存储和处理机制，并采用 HQL (类 SQL)语言

对这些数据进行自动化管理和处理。Hive 中的海量结构化数据被看成一个个的表，而实际上，这些数据是分布式存储在 HDFS 中的。注意，Hive 是离线查询工具，由于其内部机制需要把 SQL 转换成 MapReduce 后进行分布式查询，所以查询时间最少也需要十几秒，仅适用于海量数据场景，不适合即时查询需求。

(2) Impala：Impala 是 Google Dremel 的 Java 实现版本之一。Dermel 由 Google 设计开发，最显著的特性就是支持 SQL 方式在秒级别分析 TB 级别数据(1TB 数据 3 秒完成分析计算)。Impala1.0 版本完全兼容 SQL92 规范，不同于 Hive 将 SQL 转换为 MapReduce 方式，Impala 通过与商用并行关系数据库类似的分布式查询引擎框架(由 Query Planner、Query Coordinator 和 Query Exec Engine 三部分组成，与 MR 相似的技术架构，但即时性更好)，可以直接从 HDFS 或者 HBase 中用 SELECT、JOIN 和统计函数查询数据，性能是 Hive(0.81)的 3~90 倍。虽然，目前刚发布的 Hive1.0 在原有性能上有很大提升，且都属于数据仓库工具，但 Impala 架构更先进。

(3) Pig：Apache Pig 是一个分析大规模数据集的平台，其使用场景和 Hive 相似，但 Hive 更简单，使用类 SQL 进行数据分析，Pig 使用脚本语言，编程性更强，具体选择主要由程序员的熟悉程度及场景复杂度决定。

(4) Mahout：主要用于并行数据挖掘，该框架对主流数据挖掘算法已经基于 MapReduce 进行了实现，节省了很多开发时间。如推荐引擎、用户关系引擎、GiS 热点聚类等都可以基于此框架算法来实现。

(5) Scalding：使用 Scala 编程语言封装 MapReduce 编程模型，支持 DSL(domain-specific language)语法编程，易用性大大提升。主要用于高并发简单 ETL 处理场景。

4. 任务调度

(1) Oozie：其作用就是将多个 MapReduce 作业连接到一起，作为一个工作流程执行。一般情况下，一个大型任务由多个 MapReduce 组成。如果不用 Oozie，需要手动编写大量连接和转换代码，以此串联起多个 MR 任务流程，比较耗时，出错率和维护率也比较高。Oozie 通过 xml 方式配置连接起整个任务流程。与传统工作流引擎作用相似。

(2) Azkaban：美国知名互联公司 Linkedin 发布的开源产品，属于 Oozie 的同类产品，在细节上有区别。

5. 管理

Hue 是运营和开发 Hadoop 应用的图形化用户界面。对单独的用户来说，不需要额外的安装。另外，Hue 具备简单的权限和用户管理功能，这是其他开源 UI 不具备的。

Hadoop 是一个分布式的基础架构，能够让用户高效地利用运算资源和处理海量数据，目前已在很多大型互联网企业得到了广泛应用，如亚马逊、Facebook、Yahoo 等。它是一个开放式的架构，架构成员也在不断扩充、完善。

Hadoop 是一个开发和运行处理大规模数据的软件平台，属于 Apache 开源组织，用 Java 语言开发，用于在大量计算机组成的集群中对海量数据进行分布式存储和计算。Hadoop 最核心的设计包含两个模块：HDFS 和 MapReduce。其中 HDFS 提供海量数据的存储，MapReduce 提供海量数据的分布式计算能力。

2.3.1 HDFS 系统

1. HDFS 系统的概念和特性

首先，HDFS 系统是一个文件系统，用于存储文件，通过统一的命名空间——目录树来定位文件。其次，HDFS 系统是分布式的，通过很多服务器共同作用实现其功能，集群中的服务器均有各自的角色。

HDFS 系统在大数据中的应用是为各类分布式运算框架提供数据存储服务，将大文件、大批量文件，分布式存放在大量的服务器上，以便于对海量数据分别进行运算分析。

HDFS 系统的特性如下。

(1) 具有高容错性。

(2) 整个系统部署在低廉的硬件上。

(3) 提供高传输率来访问应用程序的数据。

(4) 适合超大数据集的应用程序。

(5) 流式数据访问。

HDFS 本身是软件系统，不同于传统硬盘和共享存储介质，在文件操作上有其不同之处。

(1) 不支持文件随机写入。支持随机读，但没有随机写入机制，这与 HDFS 文件写入机制有关，所以也不支持断点续传等功能。

(2) 需要客户端与 HDFS 交互。目前已有开源支持 HDFS mount 到 Linux 服务器上，但性能非常不好。

(3) 适合大文件读取场景。因为其分块冗余存储机制，其存储架构在处理小于其分块文件大小的文件时，会浪费管理节点资源，导致效率低。

(4) 吞吐和并发具备横向扩展能力。单节点系统比传统硬盘效率低很多，但在大量机器集群环境下，其吞吐和并发能力可以线性提升，远远高于单一硬件设备。

(5) 不适合高响应系统。由于 HDFS 是为高数据吞吐量应用而设计的，而这以高延迟为代价。

2. HDFS 系统的结构

HDFS 中有 3 个重要角色：NameNode、DataNode 和 Client，如图 2.4 所示。

对外部客户机而言，HDFS 就像一个传统的分级文件系统，可以删除、移动或重命名文件等。HDFS 架构是基于一组特定的节点构建的，这是由它自身的特点所决定的。这些节点包括 NameNode(仅 1 个)，它在 HDFS 内部提供元数据服务；DataNode 为 HDFS 提供存储块。

存储在 HDFS 系统中的文件被分成块，然后将这些块复制到多台计算机(DataNode)。这与传统的 RAID 架构大不相同。块的大小(通常为 64MB)和复制的块数量在创建文件时由客户机决定。NameNode 可以控制所有文件操作。HDFS 内部的所有通信都基于标准的 TCP/IP 协议。

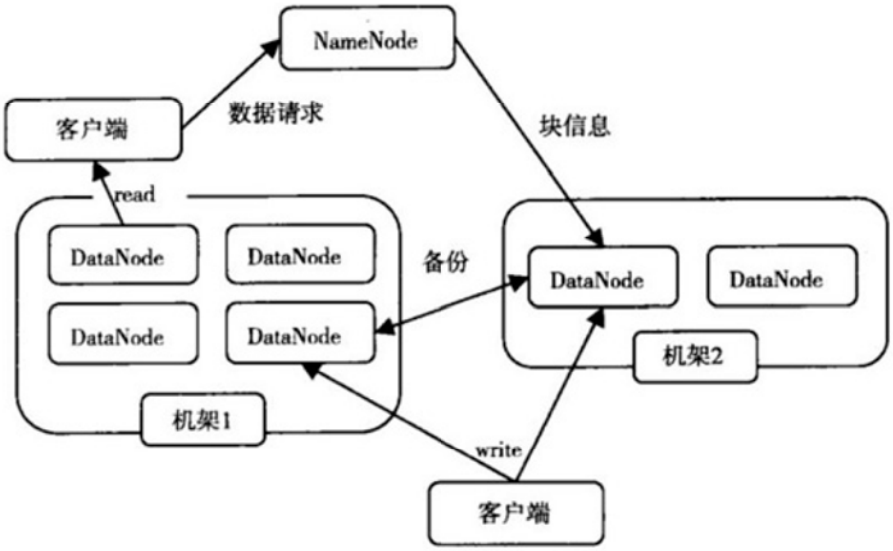

图 2.4 HDFS 系统的结构

1) NameNode

NameNode 是一款通常在 HDFS 实例中的单独机器上运行的软件。它负责管理文件系统名称空间和控制外部客户机的访问。NameNode 决定是否将文件映射到 DataNode 上的复制块上。对于最常见的 3 个复制块，第一个复制块存储在同一机架的不同节点上，最后一个复制块存储在不同机架的某个节点上。Metadata 所有的相关服务都由 NameNode 提供，包括 filename->block (namespace)，以及 block->DataNode 的对应表。其中，前者通过 FsImage 写入本地文件系统，而后者是通过每次 HDFS 启动时，DataNode 进行 blockreport 后在内存中重构的数据结构。

实际的 I/O 实务并没有经过 NameNode，只有表示 DataNode 和块的文件映射的元数据经过 NameNode。当外部客户机发送请求要求创建文件时，NameNode 会以块标识和该块的第一个副本的 DataNode 的 IP 地址作为响应。NameNode 还会通知其他将要接收该块的副本的 DataNode。

NameNode 在名为 FsImage 的文件中存储所有关于文件系统名称空间的信息。这个文件和一个包含所有事务的记录文件(EditLog)将存储在 NameNode 的本地文件系统上。FsImage 和 EditLog 文件也需要复制副本，以防文件损坏或 NameNode 系统丢失。

2) DataNode

DataNode 也是一款通常在 HDFS 实例中的单独机器上运行的软件。Hadoop 集群中包含一个 NameNode 和大量 DataNode。DataNode 通常以机架的形式组织，机架通过一个交换机将所有系统连接起来。

DataNode 响应来自 HDFS 客户机的读写请求，并且还响应来自 NameNode 的创建、删除和复制块的命令。NameNode 依赖来自每个 DataNode 的定期心跳(Heartbeat)消息。每条消息都包含一个块报告，NameNode 可以根据这个报告验证块映射和其他文件系统元数据。

分布式文件存储的数据节点，存储着文件块(Block)，而文件是由文件块组成的，每个

块存储在多个(可配，默认为 3)不同的 DataNode，可以提高数据的可靠性。

如果客户机想将文件写到 HDFS 上，首先，需要将文件缓存到本地的临时存储区。如果缓存的数据大于所需的 HDFS 块大小，创建文件的请求将发送给 NameNode。NameNode 将以 DataNode 标识和目标块响应客户机，同时也通知将要保存文件块副本的 DataNode。其次，当客户机开始将临时文件发送给第一个 DataNode 时，将立即通过管道方式将块方式内容转发副本 DataNode。客户机也负责创建保存在相同 HDFS 名称空间中的校验文件。在最后的文件块发送后，NameNode 将文件创建提交到它的持久化数据存储(EditLog 和 FsImage 文件)中。

3) Client

用于实现客户端文件存储的所有操作，包括文件的增删及查询等。

3. HDFS 文件写入与读取

HDFS 文件的写入流程如图 2.5 所示。

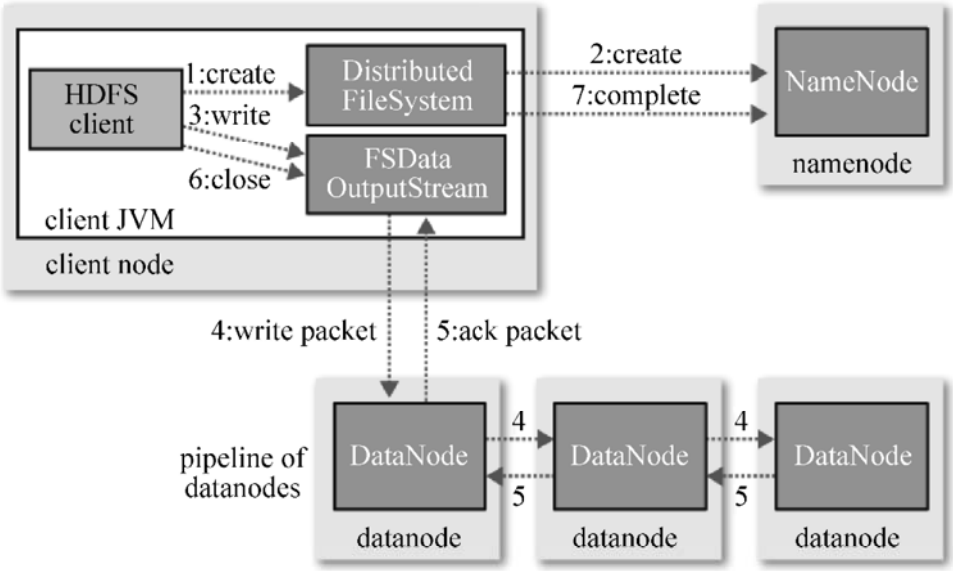

图 2.5 HDFS 文件的写入流程

(1) 客户端通过 Distributed FileSystem 上的 create()方法指明一个欲创建文件的文件名，然后 client 通过 RPC 方式与 NameNode 通信创建一个新文件映射关系。

(2) 客户端写数据：FSData OutputStream 把写入的数据分成包(packet)，放入一个中间队列——数据队列(data queue)。OutputStream 从数据队列中取数据，同时向 NameNode 申请一个新的 block 来存放它已经取得的数据。NameNode 选择一系列合适的 DataNode(个数由文件的 replication 数决定，默认为 3)，构成一个管道线(pipeline)，所以管道线中就有 3 个 DataNode。OutputStream 把数据流式地写入管道线的第一个 DataNode 中，第一个 DataNode 再把接收到的数据转到第二个 DataNode 中，以此类推。

(3) FSData OutputStream 同时也维护着另一个中间队列——确认队列(ack queue)，确认队列中的包只有在得到管道线所有的 DataNode 的确认后才会被移出确认队列。

(4) 所有文件写入完成后，关闭文件写入流。

从以上文件写入流程，可以总结出 HDFS 文件写入具备如下特性。

- 响应时间比较长。
- 文件写入效率与 block 块数和集群数量相关。

HDFS 文件的读取流程如图 2.6 所示。

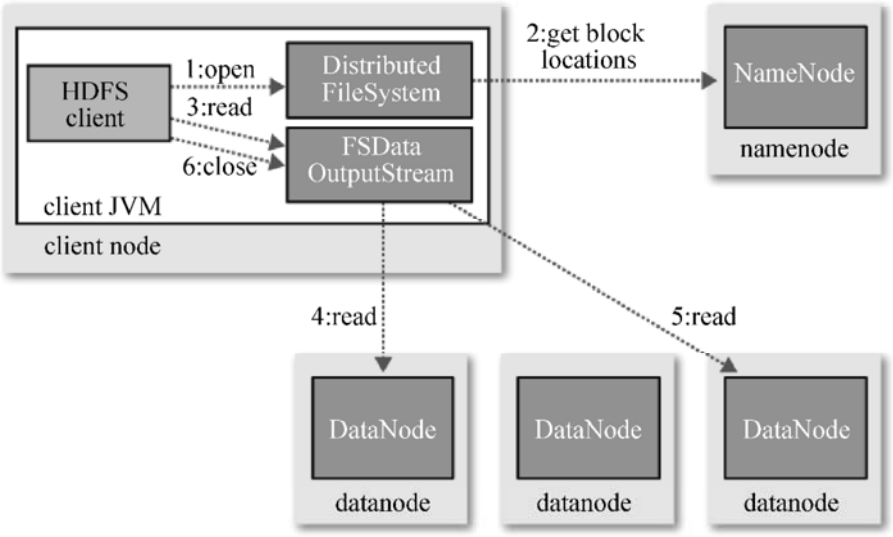

图 2.6 HDFS 文件的读取流程

(1) 打开文件流[open()]。

(2) 从 NameNode 读取文件块位置列表。

(3) FSDataInputSteam 打开 read() 方法。

(4) 根据文件块与 DataNode 的映射关系。

(5) 从不同的 DataNode 中并发读取文件块。

(6) 文件读取完毕，关闭 input 流。

因为冗余机制，当 HDFS 文件读取压力比较大时，可以通过提高冗余数的方式，NameNode 可以通过轮询方式，分配不同 client 访问不同 DataNode 上的相同文件块，提升整体吞吐率。

Hadoop 在创建新文件时是如何选择 block 的位置的？综合来说，要考虑带宽(包括写带宽和读带宽)和数据安全性，如图 2.7 所示。

如果把 3 个备份全部放在一个 DataNode 上，虽然可以避免写带宽的消耗，但几乎没有数据冗余带来的安全性，如果 DataNode 宕机，那么这个文件的所有数据就全部丢失了。另一个极端情况是，如果把 3 个冗余备份全部放在不同的机架上，甚至数据中心里面，虽然这样做数据很安全，但写数据会消耗很多的带宽。HDFS 提供了一个默认备份分配策略：把第一个备份放在与客户端相同的 DataNode 上，第二个放在与第一个不同机架的一个随机 DataNode 上，第三个放在与第二个相同机架的随机 DataNode 上。如果备份数

多于 3，则随后的备份在集群中随机存放，Hadoop 会尽量避免将过多的备份存放在同一个机架上。

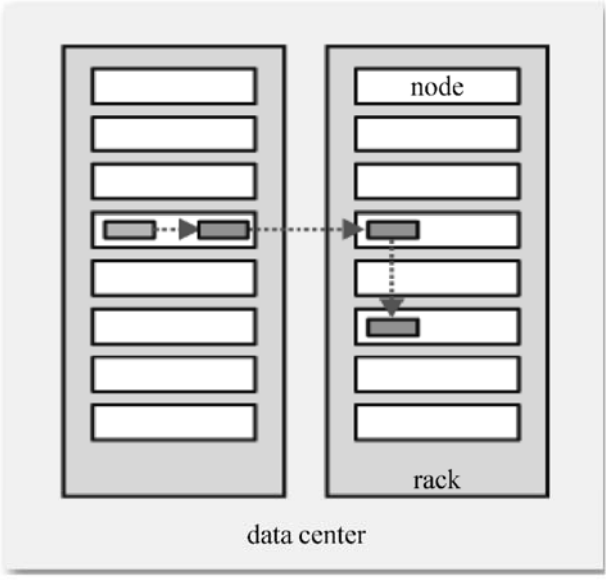

图 2.7　选择 block 的位置

2.3.2　MapReduce

MapReduce 是 Google 提出的并行计算架构，用于大规模数据集(TB 级以上)的并行运算。 此算法的计算能力，随着计算节点的数量增加而呈线性上升。

图 2.8 表示一个 MapReduce 计算处理思路，可以简要分解为两部分，数据分块映射处理(Map)和数据结果聚合(Reduce)两个步骤，源数据可以存储在 HDFS 或者第三方数据源上，计算过程临时数据存储在 HDFS 和内存中，最终获得我们需要的计算结果，其具体处理流程如图 2.9 所示。

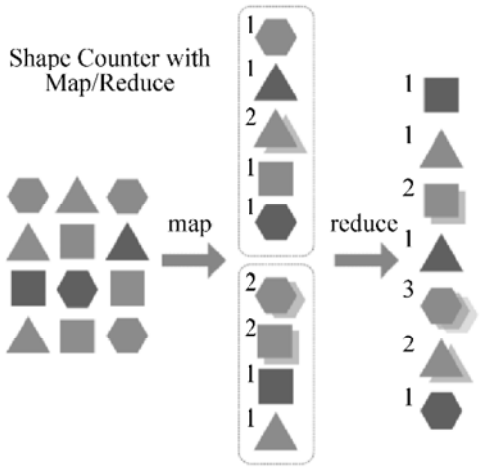

图 2.8　MapReduce 计算处理思路

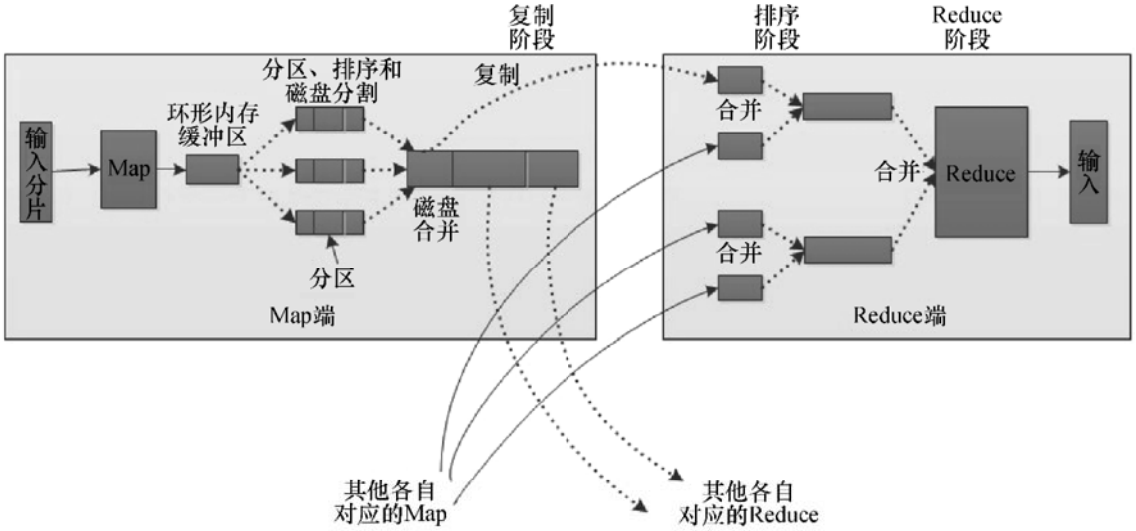

图 2.9 MapReduce 计算处理流程

1. Map 端

(1) 每个输入分片会让一个 Map 任务来处理，默认情况下，以 HDFS 的一个块的大小(默认为 64MB)为一个分片，当然，我们也可以自行设置块的大小。Map 输出的结果会暂时放在一个环形内存缓冲区中(该缓冲区的大小默认为 100MB)，当该缓冲区快要溢出时(默认为缓冲区大小的 80%)，会在本地文件系统中创建一个溢出文件，将该缓冲区中的数据写入这个文件。

(2) 在写入磁盘之前，线程首先根据 Reduce 任务的数量将数据划分为相同数量的分区，也就是一个 Reduce 任务对应一个分区的数据。这样做是为了避免有些 Reduce 任务分到大量数据，而有些 Reduce 任务却分到很少数据，甚至没有分到数据。其实分区就是对数据进行 Hash 的过程。然后对每个分区中的数据进行排序，如果此时设置了 Combiner，将排序后的结果进行 Map 合并操作，这样做的目的是让尽可能少的数据写入磁盘。

(3) 当 Map 任务输出最后一个记录时，可能会有很多溢出文件，这时需要将这些文件合并。合并的过程中会不断地进行排序和合并操作，目的有两个：一是尽量减少每次写入磁盘的数据量；二是尽量减少下一复制阶段网络传输的数据量。最后合并成一个已分区且已排序的文件。为了减少网络传输的数据量，可以将数据压缩。

(4) 将分区中的数据复制给相对应的 Reduce 任务。Map 任务一直和其父 TaskTracker 保持联系，而 TaskTracker 又一直和 JobTracker 保持心跳。所以 JobTracker 中保存了整个集群中的宏观信息。Reduce 任务只需向 JobTracker 获取对应的 Map 输出位置。

2. Reduce 端

(1) Reduce 会接收到不同的 Map 任务传来的数据，并且每个 Map 传来的数据都是有序的。如果 Reduce 端接收的数据量相当小，则直接存储在内存中，如果数据量超过了该缓冲区大小的一定比例，则对数据合并后溢写到磁盘中。

(2) 随着溢写文件的增多，后台线程会将它们合并成一个更大的有序的文件，这样做是为了给后面的合并节省时间。其实不管在 Map 端还是 Reduce 端，MapReduce 都是反复

地执行排序、合并操作，这就是为什么有些人会说“排序是 Hadoop 的灵魂”。

(3) 合并的过程中会产生许多中间文件(写入磁盘了)，但 MapReduce 会让写入磁盘的数据尽可能地少，并且最后一次合并的结果并没有写入磁盘，而是直接输入 Reduce 函数。

3. Shuffle

在 Hadoop 的集群环境中，大部分 Map 任务和 Reduce 任务是在不同的 Node 上执行，主要的开销是网络开销和磁盘 I/O 开销，因此 Shuffle 的主要作用如下。

(1) 完整地从 Map 端传输到 Reduce 端。

(2) 跨节点传输数据时，尽可能减少对带宽的消耗(注意是 Reduce 执行时去拉取 Map 端的结果)。

(3) 减少磁盘 I/O 开销对任务的影响。

2.3.3 HBase

HBase 是 Google Bigtable 的开源实现版本。数据存储在 HDFS 中，继承了 HDFS 的高可靠性、可伸缩架构，同时自己实现了高性能、列存储、实时读写的特性。

不同于 HDFS 的高吞吐低响应，HBase 设计用于高并发读写场景。

(1) HIBase 基于 Hadoop HIDFS append 方式进行数据追加操作，非常适合列族文件存储架构。

(2) HBase 写请求，都会先写 redo log，然后更新内存中的缓存。缓存会定期刷入 HDFS。文件基于列创建，因此任何一个文件(MapFile)都只包含一个特定列的数据。

(3) 当某一列的 MapFile 数量超过配置的阈值时，一个后台线程就开始将现有的 MapFile 合并为一个文件，这一操作叫 Compaction。在合并的过程中，读写不会被阻塞。

(4) 读操作会先检查缓存，若未命中，则从最新的 MapFile 开始，依次往最早的 MapFile 找数据。可以想象，一次随机读操作可能需要扫描多个文件。

HBase 的文件和日志确实都存储在 HDFS 中，但通过精致设计的算法实现了对高并发数据随机读写的完美支持，这依赖于 HBase 数据排序后存储的特性。与其他的基于 Hash 寻址的 NoSQL 数据库有很大不同。

在使用特性上，原生 HBase 既不支持 JDBC 驱动，也不支持 SQL 方式进行数据查询，只有简单的 PUT 和 GET 操作。数据查询通过主键(row key)索引和 Scan 查询方式实现，在事务上，HBase 支持单行事务(可通过上层应用和模块如 hive 或者 coprocessor 来实现多表 join 等复杂操作)。HBase 主要用来存储非结构化和半结构化的松散数据，如图 2.10 所示。

HBase 中的表一般有以下特点。

(1) 大：一个表可以有上亿行，上百万列。

(2) 面向列：面向列(族)的存储和权限控制，列(族)独立检索。

(3) 稀疏：对于为空(null)的列，并不占用存储空间(每个列族是一个文件，没内容的情况下不会占用空间)，因此，表可以设计得非常稀疏。

(4) HBase 适用于海量高并发文本数据写入、存储、查询需求场景，这些数据量是传统数据库难以满足的。

(5) 详单管理、查询。

(6) GiS 数据存储、统计。

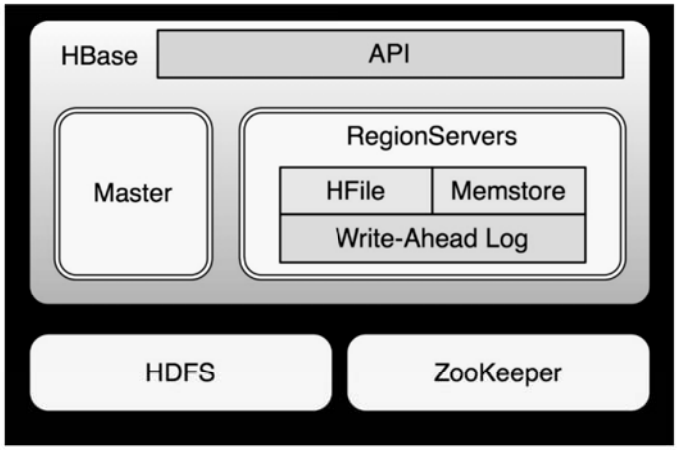

图 2.10 HBase 的架构

2.4 数据挖掘方法

大数据时代，数据挖掘是最关键的工作。大数据的挖掘是从海量的、不完全的、有噪声的、模糊的、随机的大型数据库中发现隐含其中有价值的、潜在有用的信息和知识的过程，也是一种决策支持过程。其主要基于人工智能、机器学习、模式学习、统计学等，通过对大数据高度自动化的分析，做出归纳性的推理，从中挖掘出潜在的模式，可以帮助企业、商家、用户调整市场政策，减少风险，理性面对市场，并做出正确的决策。目前，在很多领域尤其是在商业领域(如银行、电信、电商等)，数据挖掘可以解决很多问题，包括市场营销策略制定、背景分析、企业管理危机等。大数据的挖掘常用的方法有分类分析法、回归分析法、基于树的方法、聚类分析法、关联规则法、因子分析法、主成分分析法、Web 数据挖掘等。这些方法从不同的角度对数据进行挖掘。

2.4.1 分类分析方法

分类是数据挖掘技术中运用最为广泛且比较重要的分析手段，它是指运用训练数据集，通过分析数据的特征和运用一定的算法求得分类规则，该分类规则就是数据分类的模型，然后运用该模型对任何位置的数据对象进行分类。分类分为两个阶段：第一阶段为构建分类模型，通过一定的算法对已知类标记的数据集建立分类模型；第二阶段为用第一阶段构造的模型来预测给定的数据对象的类别。比较典型的分类方法有 Logistic 回归、判别分析法、K-近邻分类法、神经网络分类法、决策树方法以及随机森林方法。判别分析法又可进一步分为贝叶斯判别分析法、线性分析法以及二次判别分析法。分类分析可以被用于分析客户的属性和特征，进行精准营销。

1. Logistic 回归

Logistic 回归是一种经典分类算法，适合用于病例对照研究、经济预测随访研究和横断面研究等，且结果变量的取值必须是二分的或多项分类。可以利用影响结果变量的因素作为自变量，将数据拟合到一个 logistic 函数中，建立回归方程，从而预测某一结果发生的概率，将数据分成不同类别。

根据定性因变量取值的特点，可将其分为二元变量和多分类变量。二元变量的取值一般为 1 和 0，取值为 1 表示某件事情发生，取值为 0 则表示不发生。例如，银行可以基于贷款客户的收入情况、受教育程度、还款记录等特征预测客户的违约概率，判断客户违约这件事是否会发生，从而评估自己的坏账水平。此时，可将客户是否违约看作一个二元变量，取值为 1 表示客户违约，取值为 0 表示客户不会违约。

考虑多元自变量下的因变量二元选择模型：

$$Y = X\beta + \varepsilon$$

也可以表示为：

$$Y_i = X_i\beta + \varepsilon_i,\ i = 1,2,\cdots,n$$

其中，$\beta = (\beta_0, \beta_1, \cdots, \beta_{k+1})'$，$X_i$ 是包含常数项的 k 元设计矩阵，Y_i 是二元取值的因变量：

$$Y_i = \begin{cases} 1, & \text{某一件事发生} \\ 0, & \text{某一件事不发生} \end{cases}$$

若假定 $E(\varepsilon_i \mid X_i) = 0$，则总体回归方程为：

$$E(Y_i \mid X_i) = X_i\beta$$

进一步地假设在给定 X_i 时，某一事件发生的概率为 $p(X_i)$，不发生的概率为 $(1 - p(X_i))$，即 $\text{Prob}(Y_i = 1 \mid X_i) = p(X_i)$，$\text{Prob}(Y_i = 0 \mid X_i) = 1 - p(X_i)$。由于 Y_i 只取两个值，所以其条件概率为：

$$E(Y_i \mid X_i) = 1 \times \text{Prob}(Y_i = 1 \mid X_i) + 0 \times \text{Prob}(Y_i = 0 \mid X_i) = p(X_i)$$

综合上述两式可得到：

$$E(Y_i \mid X_i) = X_i\beta = \text{Prob}(Y_i = 1 \mid X_i) = p(X_i)$$

因此，方程拟合的是当给定自变量 X_i 的值时，某事件发生的平均概率。这一概率体现为线性的形式 $X_i\beta$，也称此为线性概率模型(Linear Probability Model，LPM)。实际上是用普通线性回归方法对二元取值的因变量进行建模。由于用最小二乘法对待估参数进行估计时，无法保证 $X_i\beta$ 的值在 0～1，完全有可能出现大于 1 或小于 0 的情形。

由于 LPM 不能保证概率的取值在 0～1，一般不直接使用 LPM。为避免这类问题，必须找到一个函数建立针对 $p(X)$ 的模型，使对任意 X 值该函数的输出结果都在 0 和 1 之间。在 Logistic 回归中，使用 Logistic 函数进行处理：

$$p(X) = \frac{e^{X\beta}}{1 - e^{X\beta}}$$

通过整理，可得：

$$\frac{p(X)}{1-p(X)}=e^{X\beta}$$

其中，$\frac{p(X)}{1-p(X)}$的值称为发生比，取值范围为 0 到∞，其值接近于 0 表示事件发生概率非常低，接近于∞则表示事件发生概率非常高。

对于模型中的待估参数$\beta=(\beta_0,\beta_1,\cdots,\beta_{k+1})'$，在 Logistic 回归中常采用极大似然估计方法来估计。极大似然法拟合 Logistic 回归模型的基本思想是寻找$\beta_0,\beta_1,\cdots,\beta_{k+1}$的一个估计，使由 Logistic 函数得到的事件发生预测概率最大可能地与观测情况接近。这个思想可以表达为数学方程的似然函数，形式如下：

$$L(\beta_0,\beta_1,\cdots,\beta_{k+1})=\prod_{i:y_i=1}p(x_i)\prod_{i':y_{i'}=0}(1-p(x_{i'}))$$

所估计的系数$\hat{\beta}_0,\hat{\beta}_1,\cdots,\hat{\beta}_{k+1}$应使似然函数值最大。将得到的参数估计值代入 Logistic 回归函数中，得到模型表达式，并可由新的X得到事件发生的概率，从而判定X的从属类别。一般来说，概率大于 0.5 时，属于 1 类；概率小于 0.5 时，属于 0 类。

2. 判别分析法

对于二元因变量，Logistic 模型采取的方法是在给定自变量X情况下，建立因变量Y的条件分布模型。还有一种间接估计这些概率的方法——判别分析。判别分析采取的方法是先对每一个给定的Y建立自变量X的分布，然后使用贝叶斯定理反过来再去估计$\text{Prob}(Y=k\mid X=x)$，即估计在给定x的条件下，判断Y属于某一类的概率。

为什么有了 Logistic 回归模型，还要考虑判别分析呢？一方面，当类别的区分度较高，或当样本量n较小且自变量X近似服从正态分布时，Logistic 模型的参数估计相对不稳定。另一方面，响应分类多于两类时，判别分析应用更普遍。

1) 贝叶斯分类法

贝叶斯(Bayes)分类算法是利用统计学贝叶斯定理，来预测类成员的概率，即给定一个样本，计算该样本属于一个特定的类的属性。这些算法主要利用 Bayes 定理来预测一个未知类别的样本属于各个类别的可能性，选择其中可能性最大的一个类别作为该样本的最终类别。由于贝叶斯定理的成立本身需要一个很强的条件独立性假设前提，而此假设在实际情况中经常是不成立的，因而其分类准确性就会下降。为此就出现了许多降低独立性假设的贝叶斯分类算法，如 TAN 算法，它是在贝叶斯网络结构的基础上增加了属性对之间的关联来实现的。

贝叶斯分类的主要算法包括朴素贝叶斯分类算法、贝叶斯网络分类算法等。

朴素贝叶斯分类(Naïve Bayes Analysis，NBC)，假设每个属性之间都是相互独立的，并且每个属性对非类问题产生的影响都是一样的，即一个属性值对给定类的影响独立于其他属性的值。

贝叶斯定理是概率论中的一个结果，它与随机变量的条件概率以及边缘概率分布有关。通常来讲，事件 A 在事件 B 发生的条件下的概率，与事件 B 在事件 A 发生的条件下的概率是不一样的，这两者有确定的关系，贝叶斯定理就是这种关系的陈述。

贝叶斯分类方法可简述如下：

假设观测分成 K 类，$K \geqslant 2$，即定性的响应变量 Y 可以取 K 个不同的无序值。

设 π_k 为一个随机选择的观测来自第 k 类的先验概率，即给定观测属于响应变量 Y 的第 k 类的概率。

设 $f_k(X) \equiv \Pr(X = x \mid Y = k)$ 表示第 k 类观测的 X 的密度函数，如果第 k 类的观测在 $X \approx x$ 附近有很大的可能性，那么 $f_k(X)$ 值很大；反之，则很小。贝叶斯定理可以表述为：

$$\Pr(Y = k|X = x) = \frac{\pi_k f_k(x)}{\sum_{i=1}^{K} \pi_k f_k(x)}$$

记 $p_k(X) = \text{Prob}(Y = k \mid X)$，只需要估计 π_k 和 $f_k(x)$ 就可以计算出 $p_k(x)$。

通常对 π_k 的估计可以通过取一些变量 Y 的随机样本，分别计算属于第 k 类的样本占总样本的比例，即 $\hat{\pi}_k = \frac{n_k}{n}$。$f_k(x)$ 的估计相对复杂，除非假设它们的密度函数形式简单。称 $p_k(x)$ 为 $X = x$ 的观测属于第 k 类的后验概率，即给定观测的预测变量值时，观测属于第 k 类的概率。贝叶斯分类将一个观测分到 $p_k(x)$ 最大的一类中，它在所有分类中的错误率最小。

2) 线性判别分析法(Linear Discriminant Analysis，LDA)

线性判别分析在进行分类时，若找到合适的方法估计 $f_k(x)$，便可构造一个与贝叶斯分类类似的分类方法，进而估计出 $p_k(x)$，然后根据 $p_k(x)$ 的值，将观测分入值最大的一类中。为获取 $f_k(x)$ 的估计，首先需要对其做一些假设。

通常，假设 $f_k(x)$ 的分布是正态的，当 $p = 1$ 时，密度函数为一维正态密度函数：

$$f_k(x) = \frac{1}{\sqrt{2\pi}\sigma_k} \exp\left[-\frac{1}{2\sigma_k^2}(x - \mu_k)^2\right]$$

其中，μ_k 和 σ_k^2 分别为第 k 类的均值和方差。再假设 $\sigma_1^2 = \cdots = \sigma_K^2 = \sigma^2$，即所有 K 个类别方差相同，均为 σ^2。

结合前式，有：

$$p_k(x) = \frac{\pi_k \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2\sigma^2}(x - \mu_k)^2\right]}{\sum_{i=1}^{K} \pi_i \frac{1}{\sqrt{2\pi}\sigma} \exp\left[-\frac{1}{2\sigma^2}(x - \mu_i)^2\right]}$$

对其取对数，并将对 $p_k(x)$ 大小无影响的项删除，整理等式，得到将观测分入

$$\delta_k(x) = x\frac{\mu_k}{\sigma^2} - \frac{\mu_k^2}{2\sigma^2} + \ln\pi_k$$

最大的一类中。

线性判别分析方法常用的参数估计如下：

$$\hat{\mu}_k = \frac{1}{n_k} \sum_{i:y_i = k} x_i$$

$$\hat{\sigma}^2 = \frac{1}{n - K} \sum_{k=1}^{K} \sum_{i:y_i = k} (x_i - \mu_k)^2 = \sum_{k=1}^{K} \frac{n_k - 1}{n - K} \hat{\sigma}_k^2$$

其中，n 为随机抽取的样本数，n_k 为第 k 类的样本数，μ_k 的估计值为第 k 类观测的均值；$\hat{\sigma}_k^2=\frac{1}{n_k-1}\sum_{i:y_i=k}(x_i-\mu_k)^2$ 为第 k 类观测的样本方差，σ^2 的估计值可以看作 k 类样本方差的加权平均。

结合上式可得线性判别分析的判别函数：

$$\hat{\delta}_k=x\frac{\hat{\mu}_k}{\hat{\sigma}^2}-\frac{\hat{\mu}_k^2}{2\hat{\sigma}^2}+\ln\hat{\pi}_k$$

LDA 分类将观测 $X=x$ 分入 $\hat{\delta}_k$ 值最大的一类中。由于判别函数中 $\hat{\delta}_k$ 是关于 x 的线性函数，所以该方法称为线性判别分析。

与贝叶斯分类比较，LDA 分类是建立在观测都来自均值不同、方差相同的正态分布假设上的，将均值、方差和先验概率的参数代入贝叶斯分类便可得到 LDA 分类。

若自变量维度 $p>1$，假设 $X=(X_1,\cdots,X_p)$ 服从一个均值不同、协方差矩阵相同的多元正态分布，即假设第 k 类观测服从一个多元正态分布 $N(\mu_k,\Sigma)$，其中，μ_k 是一个均值向量，Σ 为所有 k 类共同的协方差矩阵，其密度函数形式为：

$$f_k(x)=\frac{1}{(2\pi)^{\frac{p}{2}}|\Sigma|^{\frac{1}{2}}}\exp\left[-\frac{1}{2}(x-\mu_k)^T\Sigma^{-1}(x-\mu_k)\right]$$

通过类似一维自变量的方法，可以知道将 $X=x$ 分入

$$\delta_k(x)=x^T\Sigma^{-1}\mu_k-\frac{1}{2}\mu_k^T\Sigma^{-1}\mu_k+\ln\pi_k$$

最大的一类中。

同样地，需要估计未知参数 $\mu_1,\cdots,\mu_K$，$\pi_1,\cdots,\pi_K$ 和 Σ，估计方法与一维类似。同样地，LDA 分类将各个参数估计值代入判别函数中，并将观测值 $X=x$ 分入 $\hat{\delta}_k(x)$ 值最大的一类。

3) 二次判别分析法(Quadratic Discriminant Analysis，QDA)

如前所讨论的，LDA 假设每一类观测服从协方差矩阵相同的多元正态分布，但现实中可能很难满足这样的假设。二次判别分析放松了这一假设，虽然 QDA 分类也假设每类观测都服从一个正态分布，并把参数估计代入贝叶斯定理进行预测，但 QDA 假设每类观测都有自己的协方差矩阵，即假设第 k 类观测服从的分布为 $X\sim N(\mu_k,\Sigma_k)$，其中，Σ_k 是第 k 类观测的协方差矩阵。此时，二次判别函数：

$$\delta_k(x)=-\frac{1}{2}(x-\mu_k)^T\Sigma_k^{-1}(x-\mu_k)+\ln\pi_k$$

QDA 分类把 μ_k、Σ_k、π_k 的估计值代入函数，然后将观测分入使 $\hat{\delta}_k(x)$ 值最大的一类。

通常情况下，如果训练数据相对较少，那么降低模型的方差就有必要，这时 LDA 是一个比 QDA 更好的选择。反之，如果训练集非常大，则会更倾向于使用 QDA，因为此时 K 类的协方差矩阵往往不相同。

3. *k*-近邻分类法

k-近邻分类法不是事先通过数据来选好分类模型，再对未知样本分类，而是存储带有

标记的样本集，给一个没有标记的样本，用样本集中 k 个与之相近的样本对其进行即时分类。k-近邻就是找出 k 个相似的样本来建立目标函数逼近。

k-近邻分类法的基本思路是首先，存储一些标记好的样本集；其次，要有一个未知类的样本用来对其分类；再次，逐一取出样本集中的样本，与未知类样本相比较，找到 k 个与之相近的样本，用这 k 个样本的多数的类为未知样本定类；最后，在样本集为连续值时，用 k 个样本的平均值为未知样本定值。

4. 神经网络分类法

神经网络作为一种先进的人工智能技术，因其自身自行处理、分布存储和高度容错等特性，非常适合处理非线性的以及那些以模糊、不完整、不严密的知识或数据为特征的问题，它的这一特点十分适合解决数据挖掘的问题。

神经网络由大量的人工神经元联结进行计算。大多数情况下人工神经网络能在外界信息的基础上改变内部结构，是一种自适应系统。现代神经网络是一种非线性统计性数据建模工具，常用来对输入和输出间复杂的关系进行建模，或用来探索数据的模式。

图 2.11 是大脑神经元的学习训练模型：

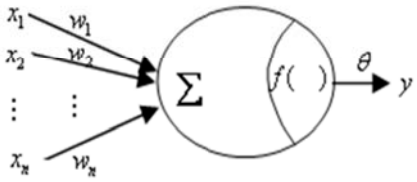

图 2.11　大脑神经元学习训练模型

在图 2.11 中，$x_1, x_2, \cdots, x_n$ 可以理解为 n 个输入信号(信息)，$w_1, w_2, \cdots, w_n$ 可以理解为对 n 个信号的加权，从而得到一个综合信号 $\Sigma = \sum_{i=1}^{n} w_i x_i$ (对输入信号进行加权求和)。神经元需要对这个综合信号做出反应，即引入一个阈值 θ 并与综合信号比较，根据比较的不同做出不同的反应，即输出。这里用激发函数 $f(\Sigma - \theta)$ 来模拟其反应，从而获得反应值，进而进行判别。

借鉴人脑的工作机制和活动规律，简化其网络结构，并用数学模型来模拟，从而提出了神经网络模型。典型的神经网络模型主要分为三大类：第一类是用于分类预测和模式识别的前馈式神经网络模型，其主要代表为函数型网络、感知机、BP 神经网络。第二类是用于联想记忆和优化算法的反馈式神经网络模型，以 Hopfield 的离散模型和连续模型为代表。第三类是用于聚类的自组织映射方法，以 ART 模型为代表。神经网络的主要任务是根据生物神经网络的原理和实际应用的需要建造实用的人工神经网络模型，设计相应的学习算法，模拟人脑的某种智能活动，然后在技术上实现出来用以解决实际问题，在控制与优化、预测与管理、模式识别与图像处理、通信等方面得到了十分广泛的应用。

下面以常用的 BP 神经网络模型为例介绍神经网络方法的分析过程。BP 神经网络是 1986 年以 Rumelhart 和 McClelland 为首的科学家提出的概念，是一种按照误差逆向传播算法训练的多层前馈神经网络，是应用最广泛的神经网络模型之一。BP 神经网络模型的网

络结构及数学模型如图 2.12 所示。

x 为 m 维向量，y 为 n 维向量，隐含层有 q 个神经元。假设有 N 个样本数据，$\{y(t),x(t),t=1,2,\cdots,N\}$。从输入层到隐含层的权重记为：$W_{ki}(k=1,2,\cdots,q,i=1,2,\cdots,n)$，从隐含层到输出层的权重记为：$W_{ki}(k=1,2,\cdots,q,i=1,2,\cdots,n)'$。

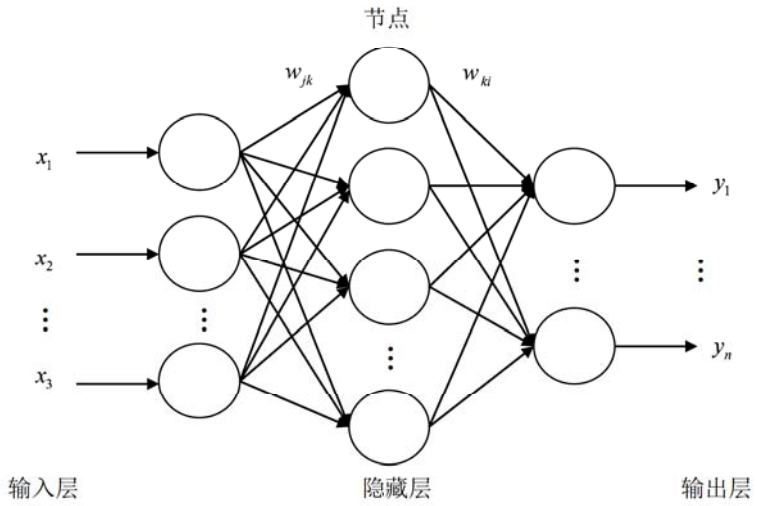

图 2.12 BP 神经网络模型结构

记第 t 个样本 $x(t)=\{x_1(t),x_2(t),\cdots,x_m(t)\}$ 输入网络时，隐含层单元的输出为 $H_k(t)(k=1,2,\cdots,q)$，输出层单元的输出为 $\hat{f}(t)=\{\hat{f}_1(t),\hat{f}_2(t),\cdots,\hat{f}_m(t)\}$，即

$$H_k(t)=g[\sum_{j=0}^{m}V_{jk}x_j(t)],k=1,2,\cdots,q$$

$$\hat{f}_i(t)=f[\sum_{k=0}^{q}V_{ki}H_k(t)],i=1,2,\cdots,n$$

这里 V_{ok} 为对应输入神经元的阈值，而 $x_0(t)$ 通常为 1，W_{oi} 为对应隐含层神经元的阈值，而 $H_0(t)$ 通常为 1，$g(x)$、$f(x)$ 分别为隐含层、输出层神经元的激发函数。常用的激发函数有：$f(x)=\dfrac{1}{1+e^{-\alpha x}}$ 或 $f(x)=\tanh(x)$。

由网络图可以看出，我们选定隐含层及输出层神经元的个数和激发函数后，这个网络就只有输入层至隐含层、隐含层至输出层的参数未知了。一旦确定了这些参数，神经网络就可以工作了。如何确定这些参数呢？其基本思路如下：通过输入的 N 个样本数据，使得真实的 y 值与网络的预测值的误差最小即可，它变成了一个优化问题如下：

$$\min E(w)=\frac{1}{2}\sum_{it}[y_t(t)-\hat{y}_i(t)]^2=\frac{1}{2}\sum_{it}[y_i(t)-f(\sum_{k=0}^{q}W_{ki}H_k(t))]^2$$

如何求解这个优化问题获得最优的 w^*？常用的有 BP 算法，这里不再介绍该算法的具体细节。

5. 决策树方法

决策树是用于分类和预测的主要技术之一，决策树学习是以实例为基础的归纳学习算法，它着眼于从一组无次序、无规则的实例中推理出以决策树表示的分类规则。构造决策

树的目的是找出属性和类别间的关系，用它来预测将来未知类别记录的类别。它采用自顶向下的递归方式，在决策树的内部节点进行属性的比较，并根据不同的属性值判断从该节点向下的分支，在决策树的叶节点得到结论。决策树的表现形式类似于流程图的树结构，在决策树的内部节点进行属性值测试，并根据属性值判断由该节点引出的分支，在决策树的叶节点得到结论。内部节点是属性或者属性组合，而叶节点代表样本所属的类或类分布。经由训练样本集产生一棵决策树后，为了对未知样本集进行分类，需要在决策树上测试未知样本的属性值。测试路径是由根节点到某个叶节点，叶节点代表的类就是该样本所属的类。

决策树将决策过程各个阶段的结构绘制成一张箭线图，如图2.13所示。

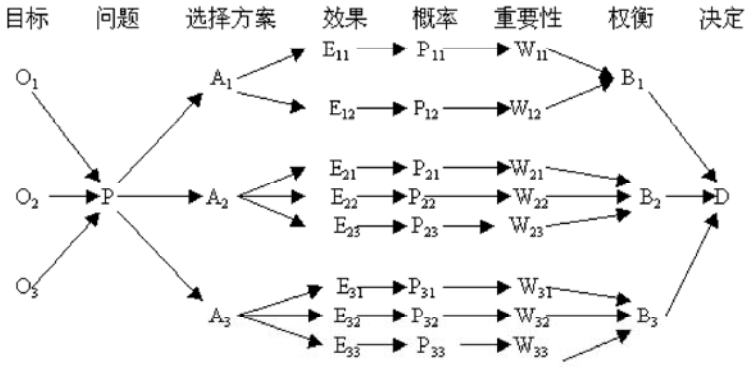

图2.13　决策树决策过程箭线图

决策树的构成有四个要素：(1)决策节点；(2)方案枝；(3)状态节点；(4)概率枝，如图2.14所示。

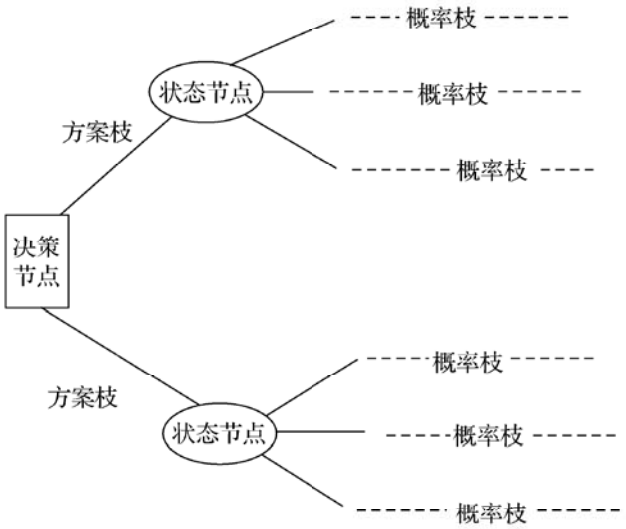

图2.14　决策树的构成要素

决策树一般由方块节点、圆形节点、方案枝、概率枝等组成，方块节点称为决策节点，由节点引出若干条细支，每条细支代表一个方案，称为方案枝；圆形节点称为状态节点，由状态节点引出若干条细支，表示不同的自然状态，称为概率枝。每条概率枝代表一

种自然状态，在每条细枝上标明客观状态的内容和其出现概率。在概率枝的最末梢标明该方案在该自然状态下所达到的结果(收益值或损失值)。这样树形图由左向右，由简到繁展开，组成一个树状网络图。

科学的决策是现代管理者的一项重要职责。在企业管理实践中，常遇到的情境是，若干个可行性方案制订出来了，分析一下企业内、外部环境，大部分条件是已知的，但还存在一定的不确定因素。每个方案的执行都可能出现几种结果，各种结果的出现有一定的概率，企业决策既存在着一定的胜算，也存在着一定的风险。这时，决策的标准只能是期望值，即各种状态下的加权平均值。针对上述问题，用决策树法来解决不失为一种好的选择。

决策树法作为一种决策技术，已被广泛地应用于企业的投资决策之中，它是随机决策模型中最常见、最普及的一种规策模式和方法。此方法有效地控制了决策带来的风险。所谓决策树法，就是运用树状图表示各决策的期望值，通过计算，最终优选出效益最大、成本最小的决策方法。决策树法属于风险型决策方法，不同于确定型决策方法，二者适用的条件也不同。

应用决策树决策方法必须具备以下条件。

(1) 具有决策者期望达到的明确目标；

(2) 存在决策者可以选择的两个以上的可行备选方案；

(3) 存在着决策者无法控制的两种以上的自然状态(如气候变化、市场行情、经济发展动向等)；

(4) 不同行动方案在不同自然状态下的收益值或损失值(简称损益值)可以计算出来；

(5) 决策者能估计出不同的自然状态发生概率。

决策树法的决策程序如下：

(1) 绘制树状图，根据已知条件排列出各个方案和每一方案的各种自然状态；

(2) 将各状态概率及损益值标于概率枝上；

(3) 计算各个方案期望值并将其标于该方案对应的状态节点上；

(4) 进行剪枝，比较各个方案的期望值，并标于方案枝上，期望值小的(即劣等方案剪掉)所剩的最后方案为最佳方案。

决策树法在企业决策中有着广泛的应用。

6. 随机森林方法

随机森林，指利用多棵树对样本进行训练并预测的一种分类器。该分类器最早由 Leo Breiman 和 Adele Cutler 提出，并被注册成了商标。简单来说，随机森林就是由多棵 CART(Classification And Regression Tree)构成的。对于每棵树，它们使用的训练集是从总的训练集中有放回采样出来的，这意味着，总的训练集中的有些样本可能多次出现在一棵树的训练集中，也可能从未出现在一棵树的训练集中。在训练每棵树的节点时，使用的特征是从所有特征中按照一定比例随机地无放回地抽取的，根据 Leo Breiman 的建议，假设总的特征数量为 M，这个比例可以是 $\sqrt{M}$ 、$\frac{1}{2}\sqrt{M}$ 或 $2\sqrt{M}$ 。

因此，随机森林的训练过程可以总结如下。

(1) 给定训练集 S，测试集 T，特征维数 F。确定参数：使用到的 CART 的数量 t，每棵树的深度 d，每个节点使用到的特征数量 f。终止条件：节点上最少样本数 s，节点上最少的信息增益 m。

对于第 i 棵树，i=1,…, t：

(2) 从 S 中有放回的抽取大小和 S 一样的训练集 $S(i)$，作为根节点的样本，从根节点开始训练。

(3) 如果当前节点上达到终止条件，则设置当前节点为叶子节点，如果是分类问题，该叶子节点的预测输出为当前节点样本集合中数量最多的那一类 $c(j)$，概率 p 为 $c(j)$占当前样本集的比例；如果是回归问题，预测输出为当前节点样本集各个样本值的平均值。然后继续训练其他节点。如果当前节点没有达到终止条件，则从 F 维特征中无放回地随机选取 f 维特征。利用这 f 维特征，寻找分类效果最好的一维特征 k 及其阈值 th，当前节点上样本第 k 维特征小于 th 的样本被划分到左节点，其余的被划分到右节点。继续训练其他节点。有关分类效果的评判标准在后面会讲。

(4) 重复(2)、(3)直到所有节点都训练过了或者被标记为叶子节点。

(5) 重复(2)、(3)、(4)直到所有 CART 都被训练过。

利用随机森林的预测过程如下：

对于第 i 棵树，i=1,…, t：

(1) 从当前树的根节点开始，根据当前节点的阈值 th，判断是进入左节点($<th$)还是进入右节点($\geqslant th$)，直到到达，某个叶子节点，并输出预测值。

(2) 重复执行(1)直到所有 t 棵树都输出了预测值。如果是分类问题，则输出为所有树中预测概率总和最大的那一个类，即对每个 $c(j)$的 p 进行累计；如果是回归问题，则输出为所有树的输出的平均值。

2.4.2 回归分析方法

回归分析是指对具有相关关系的两个变量或多个变量建立合适的数学模型，以近似地表示变量之间平均变化关系的一种统计方法。回归分析与分类分析类似，但回归分析的目的不是寻找描述类的模式，而是寻找变量间的关系模式以确定数值。例如，简单的线性回归技术，它的结果是一个函数，可以根据输入变量的值来计算输出变量的值。此外，还有非线性回归模型，有的可以转化为线性模型。回归分析方法被广泛地用于解释市场占有率、销售额、品牌偏好及市场营销效果。在此重点介绍线性回归中的简单线性回归与多元线性回归模型的主要思路以及最常用于拟合模型的最小二乘法。

1. 简单线性回归

简单线性回归是一种非常简单的根据单一预测变量 X 预测定量响应变量 Y 的方法。它假定 X 和 Y 存在线性关系，模型的一般形式是：

$$Y = \beta_0 + \beta_1 X + \mu$$

其中，Y 为被解释变量，X 为解释变量，β_0 与 β_1 为待估参数，μ 为随机扰动项。在有 n 个样本观测点 $\{(X_i, Y_i): i = 1,2,\cdots,n\}$ 的情况下，模型也可以表示为：

$$Y_i = \beta_0 + \beta_1 X_i + \mu_i, i = 1,2,\cdots,n$$

样本回归模型可表示为：

$$\hat{Y} = \hat{\beta}_0 + \hat{\beta}_1 X$$

其随机表达式为：

$$Y = \hat{\beta}_0 + \hat{\beta}_1 X + e$$

其中 e 称残差或剩余项，可看成总体回归模型中随机干扰项 μ 的近似替代。

回归分析的主要目的是要通过样本回归模型尽可能准确地估计总体回归模型。给定解释变量 X 与被解释变量 Y，在对模型设定、解释变量与随机干扰项进行一系列假定后，利用普通最小二乘法对待估参数 β_0 与 β_1 进行估计。普通最小二乘法要求样本回归函数尽可能好地拟合这组值，即样本回归线上的点 $\hat{Y}_i$ 与真实观测点 Y_i 的“总体误差”尽可能小。普通最小二乘法给出的判断标准是，被解释变量的估计值与实际值之差的平方和最小，

$$Q = \sum_{i=1}^{n} e_i^2 = \sum_{i=1}^{n} \left(Y_i - \hat{Y}_i\right)^2 = \sum_{i=1}^{n} [Y_i - (\hat{\beta}_0 + \hat{\beta}_1 X_i)]^2$$

即在给定样本观测值下，选择 $\hat{\beta}_0$，$\hat{\beta}_1$ 使 Y_i 与 $\hat{Y}_i$ 之差的平方和最小。对得到的表达式进行微积分学的运算，可得参数估计量：

$$\begin{cases} \hat{\beta}_1 = \dfrac{\sum x_i y_i}{\sum x_i^2} \\ \hat{\beta}_0 = \overline{Y} - \hat{\beta}_1 \overline{X} \end{cases}$$

即可进一步得到模型的估计表达式，利用该模型可以对新增的 X 进行 Y 值的预测。

2. 多元线性回归

多元线性回归模型的一般形式为：

$$Y = \beta_0 + \beta_1 X_1 + \beta_2 X_2 + \cdots + \beta_k X_k + \mu$$

其中，k 为解释变量的数目，$\beta_j (j = 1,2,\cdots,k)$ 被称为回归系数。人们习惯上把常数项 β_0 看作一个虚变量的参数，在参数估计过程中该虚变量的样本观测值始终取 1，则模型中解释变量的数目为 $k+1$。

如果给出一组观测值 $\{(X_{i1}, X_{i2}, \cdots, X_{ik}, Y_i) : i = 1,2,\cdots,n\}$，则总体回归模型还可以写成：

$$Y_i = \beta_0 + \beta_1 X_{i1} + \beta_2 X_{i2} + \cdots + \beta_k X_{ik} + \mu_i \quad i = 1,2,\cdots,n$$

或

$$Y_i = X_i \beta + \mu_i \quad i = 1,2,\cdots,n$$

其中，$X_i = (1, X_{i1}, X_{i2}, \cdots, X_{ik})$，$\beta = (\beta_0, \beta_1, \cdots, \beta_k)'$。

由前式表示的 n 个随机方程的矩阵表达式为：

$$Y = X\beta + \mu$$

其中

$$Y = (Y_1, Y_2, \cdots, Y_n)'$$

$$X=\begin{pmatrix}1 & X_{11} & X_{12} & \cdots & X_{1k}\\ 1 & X_{21} & X_{22} & & X_{2k}\\ & \vdots & \vdots & & \vdots\\ 1 & X_{n1} & X_{n2} & \cdots & X_{nk}\end{pmatrix}$$

$$\beta=(\beta_0,\beta_1,\beta_2,\cdots,\beta_k)'$$

$$\mu=(\mu_1,\mu_2,\cdots,\mu_n)'$$

与一元回归分析相仿，在给出总体的一个样本时，估计样本回归模型，并让它近似代表未知的总体回归模型。

样本回归模型可表示为：

$$\hat{Y}=\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\cdots+\hat{\beta}_kX_k$$

其随机表达式为：

$$Y=\hat{\beta}_0+\hat{\beta}_1X_1+\hat{\beta}_2X_2+\cdots+\hat{\beta}_kX_k+e$$

其矩阵表达式为：

$$\hat{Y}=X\hat{\beta}$$

$$Y=X\hat{\beta}+e$$

其中

$$\hat{\beta}=(\hat{\beta}_0,\hat{\beta}_1,\hat{\beta}_2,\cdots,\hat{\beta}_k)'$$

$$e=(e_1,e_2,\cdots,e_n)'$$

与一元线性回归类似，多元线性回归分析同样是要通过样本回归模型尽可能准确地估计总体回归模型。给定解释变量 X 与被解释变量 Y，在对模型设定、解释变量与随机干扰项进行一系列假定后，利用普通最小二乘法对待估参数 $\{\beta_j, j=0,1,2,\cdots,k\}$ 进行估计。根据最小二乘原理，参数估计值应使达到最小。

$$Q=\sum_{i=1}^{n}e_i^2=\sum_{i=1}^{n}(Y_i-\hat{Y}_i)^2=\sum_{i=1}^{n}[Y_i-(\hat{\beta}_0+\hat{\beta}_1X_{i1}+\hat{\beta}_2X_{i2}+\cdots+\hat{\beta}_kX_{ik})]^2$$

由微积分知识，只需求 Q 关于待估参数的偏导数，并令其值为 0，就可得到(k+1)个待估参数估计值的正规方程，解线性代数方程组，即可得到(k+1)个待估参数的估计值 $\{\hat{\beta}_j, j=0,1,2,\cdots,k\}$。

对上述过程用矩阵表示如下：

根据最小二乘原理，需寻找一组参数估计值 $\hat{\beta}$，使残差平方和最小，

$$Q=\sum_{i=1}^{n}e_i^2=e'e=(Y-X\hat{\beta})'(Y-X\hat{\beta})$$

即参数估计值应该是方程组的解。

$$\frac{\partial}{\partial\hat{\beta}}(Y-X\hat{\beta})'(Y-X\hat{\beta})=0$$

求解可得到参数的最小二乘估计值为：

$$\hat{\beta}=(X'X)^{-1}X'Y$$

即可进一步得到模型的估计表达式，利用该模型可以对新增的 X 进行 Y 值的预测。

2.4.3 其他方法

1. 聚类分析

聚类分析源于许多研究领域，包括数据挖掘、统计学、机器学习、模式识别等。聚类分析是指将物理或抽象对象的集合分组为由类似的对象组成的多个类的分析过程。聚类是将数据分类到不同的类或者簇这样的一个过程，所以同一个簇中的对象有很大的相似性，而不同簇间的对象有很大的相异性。聚类分析是一种探索性的分析，在分类的过程中，人们不必事先给出一个分类的标准，聚类分析能够从样本数据出发，自动进行分类。聚类分析所使用的方法不同，也会得到不同的结论。不同研究者对于同一组数据进行聚类分析，所得到的聚类数未必一致。作为数据挖掘中的一个功能，聚类分析能作为一个独立的工具来获得数据分布的情况，并且概括出每个簇的特点，或者集中注意力对特定的某些簇做进一步分析。数据挖掘技术的一个突出特点是能处理巨大的、复杂的数据集，这对聚类分析技术提出了较高的要求，要求算法具有可伸缩性，可处理不同类型的属性，可发现任意形状的类及处理高维数据等。根据潜在的各项应用，数据挖掘对聚类分析方法提出了不同要求。

聚类类似于分类，但与分类的目的不同，是针对数据的相似性和差异性将一组数据分为几个类别。属于同一类别的数据间的相似性很大，但不同类别之间数据的相似性很小，跨类的数据关联性很低。

聚类在数据挖掘中的典型应用有以下 3 个方面。①聚类分析可以作为其他算法的预处理步骤：利用聚类进行数据预处理，可以获得数据的基本情况，在此基础上进行特征抽取或分类可以提高精确度和挖掘效率，也可将聚类结果用于进一步关联分析，以获得有用信息。②可以作为一个独立的工具来获得数据的分布情况：聚类分析是获得数据分布情况的有效方法。通过观察聚类得到每个簇的特点，可以集中对特定的某些簇做进一步的分析。③聚类分析可以完成孤立点挖掘。许多数据挖掘算法试图使孤立点影响最小化，或者排除它们。然而孤立点本身可能是非常有用的，如在金融欺诈探测中，孤立点可能预示着金融欺诈行为的存在。

聚类分析法有快速聚类和系统聚类。

1) 快速聚类

要求事先确定分类。它不仅要求确定分类的类数，而且还需要事先确定点，也就是聚类种子然后，将其他点根据离这些种子的远近进行分类。之后就是将这几类的中心(均值)作为新的基石，再分类，如此迭代。

2) 系统聚类

系统聚类是将样品分成若干类的方法，其基本思想是，先将每个样品各看成一类，然后规定类与类之间的距离，选择距离最小的一对合并成新的一类，计算新类与其他类之间的距离，再将距离最近的两类合并，这样每次减少一类，直至所有的样品合为一类为止。

2. 关联规则

关联规则挖掘是数据挖掘中研究较早而且至今仍活跃的研究方法之一。关联规则是隐藏在数据项之间的关联或相互关系，即可以根据一个数据项的出现推导出其他数据项的出现。关联规则的挖掘过程主要包括两个阶段：第一阶段为从海量原始数据中找出所有的高频项目组；第二阶段为从这些高频项目组产生关联规则。关联规则挖掘技术已被广泛应用于金融行业企业中预测客户的需求，通过捆绑客户可能感兴趣的信息供用户了解并获取相应信息来改善自身的营销。

关联规则是描述数据库中数据项关系的规则，即根据一个事务中某些项的出现可导出另一些项在同一事务中也出现，即隐藏在数据间的关联或相互关系。

在客户关系管理中，通过对企业的客户数据库里的大量数据进行挖掘，可以从大量的记录中发现有趣的关联关系，找出影响市场营销效果的关键因素，为产品定位、定价与定制客户群，客户寻求、细分与保持，市场营销与推销，营销风险评估和诈骗预测等决策支持提供参考依据。

1) Apriori 算法

Apriori 算法是一种最有影响的挖掘布尔关联规则频繁项集的算法。其核心是基于两阶段频集思想的递推算法。该关联规则在分类上属于单维、单层、布尔关联规则。在这里，所有支持度大于最小支持度的项集称为频繁项集，简称频集。

该算法的基本思想是，首先找出所有的频集，这些项集出现的频繁性至少和预定义的最小支持度一样。由频集产生强关联规则，这些规则必须满足最小支持度和最小可信度。然后使用第一步找到的频集产生期望的规则，产生只包含集合的项的所有规则，其中每一条规则的右部只有一项，这里采用的是中规则的定义。一旦这些规则被生成，那么只有那些大于用户给定的最小可信度的规则才能被留下来。为了生成所有频集，使用了递推的方法。

可能产生大量的候选集，以及可能需要重复扫描数据库，是 Apriori 算法的两大缺点。

2) 基于划分的算法

Savasere 等设计了一个基于划分的算法。这个算法先把数据库从逻辑上分成几个互不相交的块，每次单独考虑一个分块并对它生成所有的频集，然后把产生的频集合并，用来生成所有可能的频集，最后计算这些项集的支持度。这里分块的大小选择要使每个分块都可以被放入主存，每个阶段只需被扫描一次。而算法的正确性是由每一个可能的频集至少在某一个分块中是频集保证的。该算法是可以高度并行的，可以把每一分块分别分配给某一个处理器生成频集。产生频集的每一个循环结束后，处理器之间进行通信来产生全局的候选 k-项集。一方面，这里的通信过程是算法执行时间的主要“瓶颈”；另一方面，每个独立的处理器生成频集的时间也是一个“瓶颈”。

3) FP-树频集算法

针对 Apriori 算法的固有缺陷，J. Han 等提出了不产生候选挖掘频繁项集的方法——FP-树频集算法。采用分而治之的策略，在第一遍扫描之后，把数据库中的频集压缩进一棵频繁模式树(FP-tree)，同时依然保留其中的关联信息，随后再将 FP-tree 分化成一些条件

库，每个库和一个长度为 1 的频集相关，然后再对这些条件库分别进行挖掘。当原始数据量很大的时候，也可以结合划分的方法，使一个 FP-tree 可以放入主存中。

3. 因子分析法

因子分析的基本目的就是用少数几个因子描述许多指标或因素之间的联系，即将相关比较密切的几个变量归在同一类中，每一类变量就成为一个因子，以较少的几个因子反映原资料的大部分信息。

运用这种研究技术，我们可以方便地找出影响消费者购买、消费和满意度的主要因素，以及这些因素的影响力如何。运用这种研究技术，我们还可以为市场细分做前期分析。

4. 主成分分析法

设法将原来的变量重新组合成一组新的互相无关的几个综合变量，同时根据实际需要从中可以取出几个较少的综合变量尽可能多地反映原变量的信息的统计方法叫作主成分分析或主分量分析，这也是数学上用来降维的一种方法。

主成分分析是设法将原来众多的具有一定相关性(比如 P 个指标)，重新组合成一组新的互相无关的综合指标来代替原来的指标。

最经典的做法就是用 F1(选取的第一个线性组合，即第一个综合指标)的方差来表达，即 Var(F1)越大，表示 F1 包含的信息越多。因此，在所有的线性组合中选取的 F1 应该是方差最大的，故称 F1 为第一主成分。如果第一主成分不足以代表原来 P 个指标的信息，再考虑选取 F2，即选第二个线性组合，为了有效地反映原来的信息，F1 已有的信息就不需要再出现在 F2 中，用数学语言表达就是要求 Cov(F1，F2)=0，则称 F2 为第二主成分，以此类推可以构造出第三、第四……第 P 个主成分。

主成分分析作为基础的数学分析方法，其实际应用十分广泛，比如人口统计学、数量地理学、分子动力学模拟、数学建模、数理分析等学科中均有应用，是一种常用的多变量分析方法。

5. Web 数据挖掘

Web 数据挖掘是一项综合性技术，指 Web 从文档结构和使用的集合 C 中发现隐含的模式 P，如果将 C 看作输入，将 P 看作输出，那么 Web 挖掘过程就可以看作从输入到输出的一个映射过程。

当前越来越多的 Web 数据都是以数据流的形式出现的，因此对 Web 数据流挖掘就具有很重要的意义。目前常用的 Web 数据挖掘算法有：PageRank 算法、HITS 算法以及 LOGSOM 算法。这三种算法提到的用户都是笼统的用户，并没有区分用户的个体。目前 Web 数据挖掘面临着一些问题，主要包括：用户的分类问题、网站内容时效性问题、用户在页面停留时间问题、页面的链入与链出数问题等。在 Web 技术快速发展的今天，这些问题仍旧值得研究并加以解决。

6. 序列分析

序列分析是对序列数据进行分析以发现蕴藏其中的模式和规律。序列数据和时间序列

数据都是连续的观测值，观测值之间相互依赖。它们之间的差别在于序列数据包含离散的状态，而时间序列是连续的数值。序列数据和关联数据比较相似，它们都是一个项集或一组状态，区别在于序列分析是状态的转移，将数据间的关联性和时间联系起来，而关联分析不需要考虑时间问题。Markov 链是进行序列分析的主要技术之一。

7. 偏差分析

数据库中一般存在很多异常数据，找出这些异常数据非常重要，偏差分析可以解决此类问题。偏差分析用于检测数据现状、历史记录与标准之间的显著变化和偏离，例如，观测结果与期望的偏离、分类中的反常实例、模式的例外等。偏差分析的基本方法就是寻找观察结果与参照之间的差别。例如，信用卡欺诈案行为检测、网络入侵检测、劣质产品分析等。

8. 预测

预测是大数据最核心的功能。大数据预测是指运用历史数据和预测模型预测未来某件事情的概率。精度和不确定性是预测的关注点，通常用预测方差进行衡量。预测技术是以表示一系列时间值的数列作为输入，再运用计算机学习和统计技术对数据进行周期性分析、趋势分析和噪声分析，进而估算这些序列未来的值。例如，可以挖掘企业的历史销售数据预测该企业未来一年的销售额。

本章总结

- 大数据的处理流程归纳为：首先，利用多种轻型数据库收集海量数据，对不同来源的数据进行预处理，并整合存储到大型数据库中；其次，根据企业或个人目的和需求，运用合适的数据挖掘技术提取有益的知识；最后，利用恰当的方式将结果展现给终端用户。具体包括数据采集、数据预处理、数据存储、数据挖掘以及数据解释五个步骤。
- 要做大数据，首先要了解自己的企业。第一步，找到核心数据。第二步，获取外围数据，通过营销活动等获取大量数据。第三步，常规渠道的数据，这就需要企业去找常规渠道里面的数据，跟自己的 CRM 结合起来。第四步，获取外部的社会化的或者非结构化的数据，即现在所谓的社会化媒体数据。这方面信息的主要特征是非结构化，而且数量庞大。
- 金融企业的核心数据主要来源于历史交易数据、用户行为数据、系统运行日志、非结构化数据、过程文档数据。核心数据最大的问题在于来源多样、流动性差、共享性差。要解决这些问题，必须明确数据相关的职责与归属，提升对数据资产质量的认识和打通数据流转。
- 金融企业外围数据主要来源于数据共享联盟、互联网数据、运营商数据。外围数据存在的问题主要是，存在数据获得成本以及无法最大限度地发挥数据的价值。
- 分类是数据挖掘技术中运用最为广泛，也是比较重要的分析手段，它是指运用训

练数据集，通过分析数据的特征和运用一定的算法求得分类规则，该分类规则就是数据分类的模型，然后运用该模型对任何位置的数据对象进行分类。分类分为两个阶段：第一阶段为构建分类模型，通过一定的算法对已知类标记的数据集建立分类模型；第二阶段用第一阶段构造的模型来预测给定的数据对象的类别。比较典型的分类方法有 Logistic 回归、判别分析法、K-近邻分类法、神经网络分类法、贝叶斯分类方法以及决策树分类方法。

本章作业

1. 列举大数据处理流程包括的步骤。
2. 简要介绍数据预处理的 3 种主要方法。
3. 简要说明 HDFS 的结构及 3 种主要角色。
4. 列举说明数据挖掘中分类分析的主要方法。