第5章 数据采集与预处理

（PPT版本号：2022--V1）

CONTENTS

5.1 数据采集

5.1.1数据采集概念

5.1.2数据采集的三大要点

5.1.3数据采集的数据源

5.1.4数据采集方法

5.1.5 网络爬虫

5.1.1数据采集概念

数据采集，又称“数据获取”，是数据分析的入口，也是数据分析过程中相当重要的一个环节，它通过各种技术手段把外部各种数据源产生的数据实时或非实时地采集并加以利用。

表 传统的数据采集与大数据采集区别

	传统的数据采集	大数据采集
数据源	来源单一，数据量相对较少	来源广泛，数据量巨大
数据类型	结构单一	数据类型丰富，包括结构化、半结构化和非结构化
数据存储	关系数据库和并行数据仓库	分布式数据库，分布式文件系统

5.1.2数据采集的三大要点

5.1.3数据采集的数据源

5.1.3 数据采集的数据源

1.传感器数据

传感器是一种检测装置，能感受到被测量的信息，并能将感受到的信息，按一定规律变换成为电信号或其他所需形式的信息输出，以满足信息的传输、处理、存储、显示、记录和控制等要求。

在工作现场，我们会安装很多的各种类型的传感器，如压力传感器、温度传感器、流量传感器、声音传感器、电参数传感器等等。

传感器对环境的适应能力很强，可以应对各种恶劣的工作环境。在日常生活中，如温度计、麦克风、DV录像、手机拍照功能等都属于传感器数据采集的一部分，支持图片、音频、视频等文件或附件的采集工作。

5.1.3 数据采集的数据源

2.互联网数据

互联网数据的采集通常是借助于网络爬虫来完成的。所谓“网络爬虫”，就是一个在网上到处或定向抓取网页数据的程序。抓取网页的一般方法是，定义一个入口页面，然后一般一个页面中会包含指向其他页面的URL，于是从当前页面获取到这些网址加入到爬虫的抓取队列中，然后进入到新页面后再递归地进行上述的操作。爬虫数据采集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。

5.1.3 数据采集的数据源

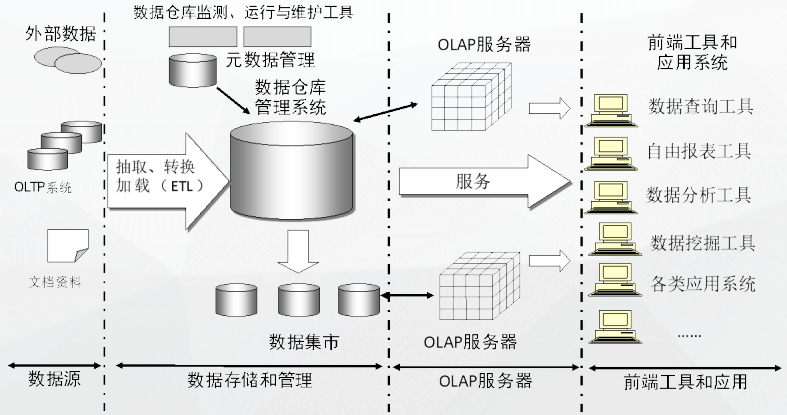
3.日志文件

许多公司的业务平台每天都会产生大量的日志文件。日志文件数据一般由数据源系统产生，用于记录数据源的执行的各种操作活动，比如网络监控的流量管理、金融应用的股票记账和Web服务器记录的用户访问行为。对于这些日志信息，我们可以得到出很多有价值的数据。通过对这些日志信息进行采集，然后进行数据分析，就可以从公司业务平台日志数据中挖掘得到具有潜在价值的信息，为公司决策和公司后台服务器平台性能评估提供可靠的数据保证。系统日志采集系统做的事情就是收集日志数据提供离线和在线的实时分析使用。很多互联网企业都有自己的海量数据采集工具，多用于系统日志采集，如Hadoop的Chukwa，Cloudera的Flume，Facebook的Scribe等，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。

5.1.3 数据采集的数据源

4.企业业务系统数据

一些企业会使用传统的关系型数据库MySQL和Oracle等来存储业务系统数据，除此之外，Redis和MongoDB这样的NoSQL数据库也常用于数据的存储。企业每时每刻产生的业务数据，以数据库一行记录形式被直接写入到数据库中。企业可以借助于ETL（Extract-Transform-Load）工具，把分散在企业不同位置的业务系统的数据，抽取、转换、加载到企业数据仓库中，以供后续的商务智能分析使用（如图所示）。通过采集不同业务系统的数据并统一保存到一个数据仓库中，就可以为分散在企业不同地方的商务数据提供一个统一的视图，满足企业的各种商务决策分析需求。



5.1.4数据采集方法

数据采集是数据系统必不可少的关键部分，也是数据平台的根基。根据不同的应用环境及采集对象，有多种不同的数据采集方法，包括：

系统日志采集

分布式消息订阅分发

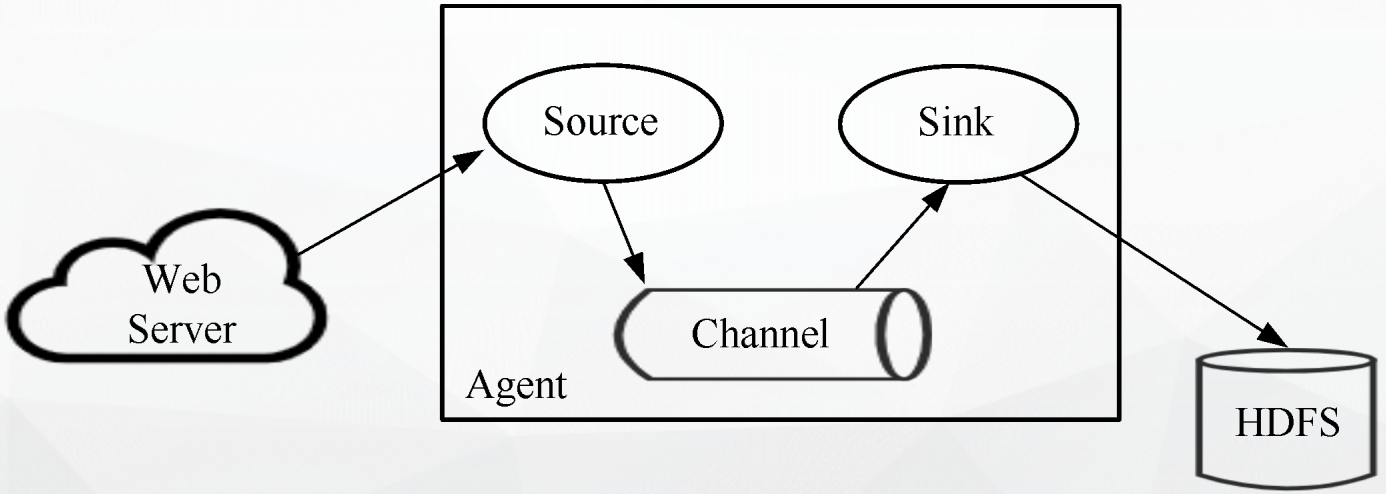
ETL

网络数据采集

5.1.4数据采集方法

1.系统日志采集

Flume是Cloudera提供的一个高可用的，高可靠的，分布式的海量日志采集、聚合和传输的系统，Flume支持在日志系统中定制各类数据发送方，用于收集数据；同时，Flume提供对数据进行简单处理，并写到各种数据接受方（可定制）的能力。



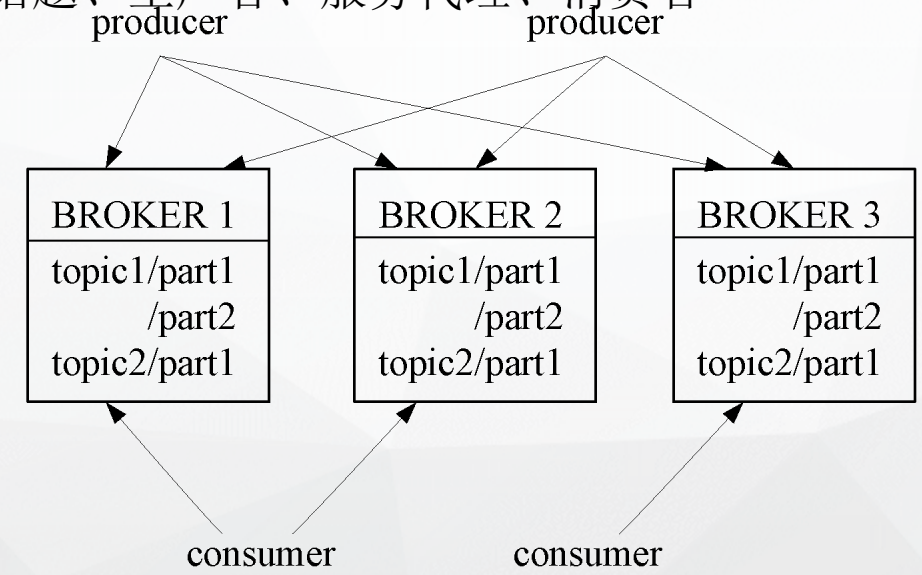
5.1.4数据采集方法

2.分布式消息订阅分发

分布式消息订阅分发也是一种常见的数据采集方式，其中，Kafka就是一种具有代表性的产品。

Kafka是由LinkedIn公司开发的一种高吞吐量的分布式发布订阅消息系统，用户通过Kafka系统可以发布大量的消息，同时也能实时订阅消费消息。

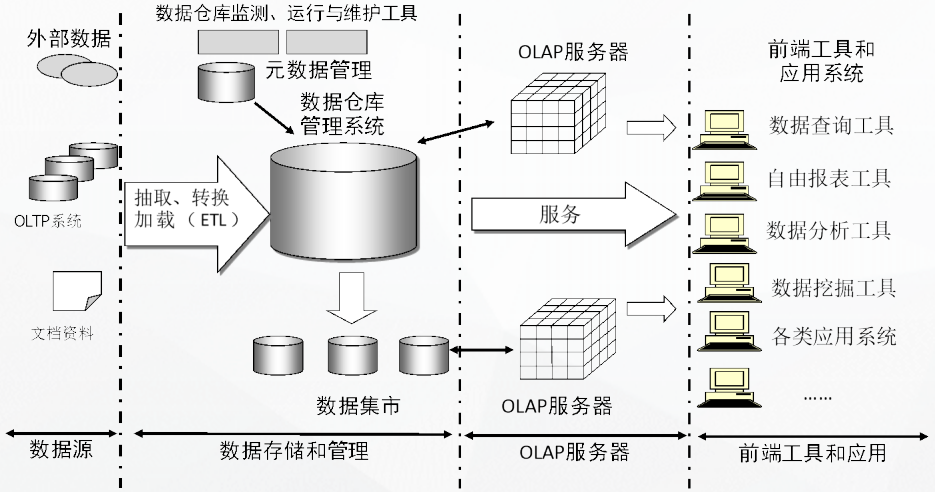
Kafka的架构包括以下组件：话题、生产者、服务代理、消费者



5.1.4数据采集方法

3. ETL

ETL是英文Extract-Transform-Load的缩写，常用于数据仓库中的数据采集和预处理环节（如图所示）。顾名思义，ETL从原系统中抽取数据，并根据实际商务需求对数据进行转换，并把转换结果加载到目标数据存储中。可以看出，ETL既包含了数据采集环节，也包含了数据预处理环节。



Kettle是一款国外开源的ETL工具，使用Java语言编写，可以在Windows、Linux、Unix上运行，数据抽取高效、稳定。

5.1.4数据采集方法

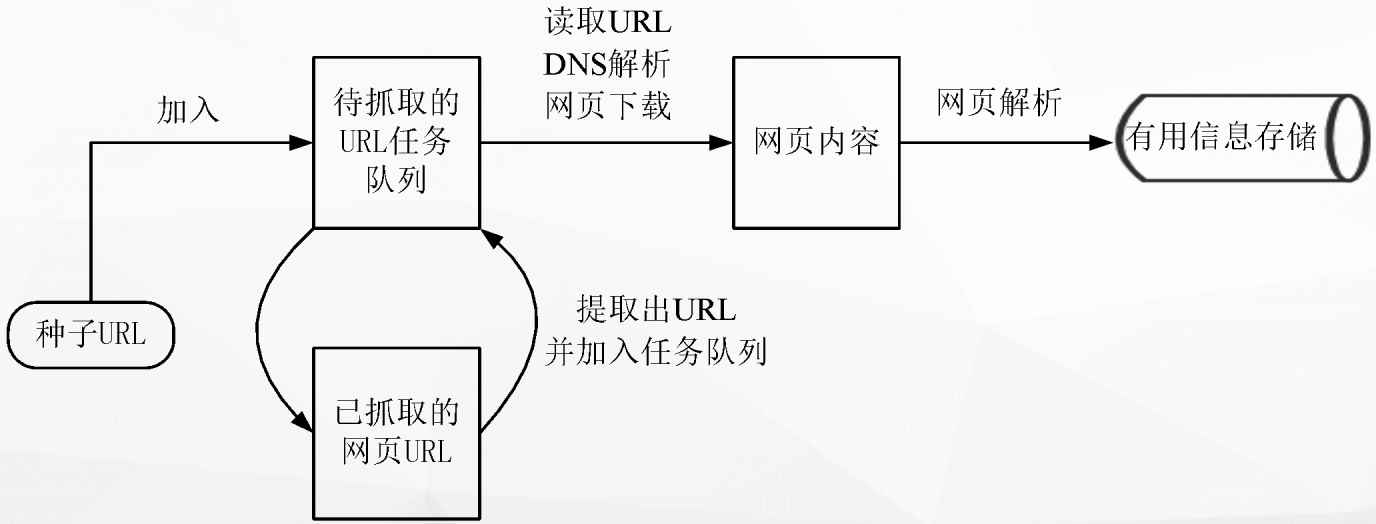
4.网络数据采集

网络数据采集是指通过网络爬虫或网站公开应用程序编程接口等方式从网站上获取数据信息。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、视频等文件的采集，文件与正文可以自动关联。网络数据采集的应用领域十分广泛，包括搜索引擎与垂直搜索平台搭建与运营，综合门户与行业门户、地方门户、专业门户网站数据支撑与流量运营，电子政务与电子商务平台的运营，知识管理与知识共享，企业竞争情报系统的运营，BI商业智能系统，信息咨询与信息增值，信息安全和信息监控等。

5.1.5 网络爬虫

1.什么是网络爬虫

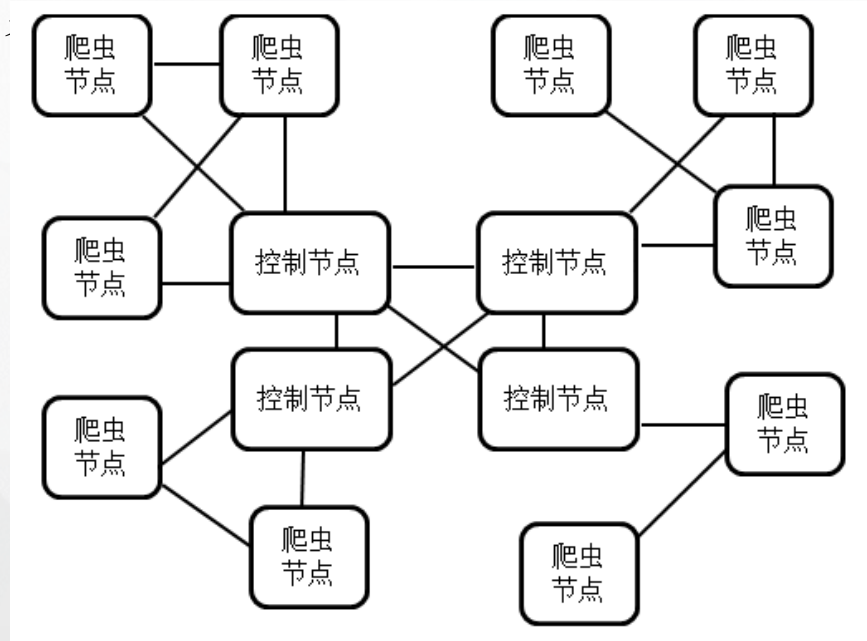
网络爬虫是一个自动提取网页的程序，它为搜索引擎从万维网上下载网页，是搜索引擎的重要组成部分。



5.1.5 网络爬虫

2.网络爬虫的组成

网络爬虫由控制节点、爬虫节点和资源库构成。网络爬虫的控制节点和爬虫节点的结构关系如图所示。从图中可以看出，网络爬虫中可以有多个控制节点，每个控制节点下可以有多个爬虫节点，控制节点之间可以互相通信，同时，控制节点和其下的各爬虫节点之间也可以进行互相通信，属于同一个控制节点下的



5.1.5 网络爬虫

3.网络爬虫的类型

5.1.5 网络爬虫

3.网络爬虫的类型

（1）通用网络爬虫。通用网络爬虫又称“全网爬虫”（（Scalable Web Crawler））， 爬行对象从一些种子URL扩充到整个Web，该架构主要为门户站点搜索引擎和大型Web服务提供商采集数据。通用网络爬虫的结构大致可以包括页面爬行模块 、页面分析模块、链接过滤模块、页面数据库、URL队列和初始URL集合。为提高工作效率，通用网络爬虫会采取一定的爬行策略。常用的爬行策略有：深度优先策略和广度优先策略。

5.1.4数据采集方法

3.网络爬虫的类型

（2）聚焦网络爬虫。聚焦网络爬虫（Focused Crawler），又称“主题网络爬虫（Topical Crawler）”，是指选择性地爬行那些与预先定义好的主题相关页面的网络爬虫。和通用网络爬虫相比，聚焦爬虫只需要爬行与主题相关的页面，极大地节省了硬件和网络资源，保存的页面也由于数量少而更新快，还可以很好地满足一些特定人群对特定领域信息的需求。聚焦网络爬虫的工作流程较为复杂，需要根据一定的网页分析算法过滤与主题无关的链接，保留有用的链接并将其放入等待抓取的URL队列。然后，它将根据一定的搜索策略从队列中选择下一步要抓取的网页URL，并重复上述过程，直到达到系统的某一条件时停止。另外，所有被爬虫抓取的网页将会被系统存储，进行一定的分析、过滤，并建立索引，以便用于之后的查询和检索；对于聚焦网络爬虫来说，这一过程所得到的分析结果还可能对以后的抓取过程给出反馈和指导。聚焦网络爬虫常用的策略包括：基于内容评价的爬行策略、基于链接结构评价的爬行策略、基于增强学习的爬行策略和基于语境图的爬行策略。

5.1.4数据采集方法

3.网络爬虫的类型

（3）增量式网络爬虫。增量式网络爬虫（Incremental Web Crawler）是指对已下载网页采取增量式更新和只爬行新产生的或者已经发生变化网页的爬虫，它能够在一定程度上保证所爬行的页面是尽可能新的页面。和周期性爬行和刷新页面的网络爬虫相比，增量式爬虫只会在需要的时候爬行新产生或发生更新的页面，并不重新下载没有发生变化的页面，可有效减少数据下载量，及时更新已爬行的网页，减小时间和空间上的耗费，但是增加了爬行算法的复杂度和实现难度。增量式爬虫有两个目标：保持本地页面集中存储的页面为最新页面和提高本地页面集中页面的质量。为实现第一个目标，增量式爬虫需要通过重新访问网页来更新本地页面集中页面内容。为了实现第二个目标，增量式爬虫需要对网页的重要性排序，常用的策略包括广度优先策略和PageRank优先策略等。

5.1.4数据采集方法

3.网络爬虫的类型

（4）深层网络爬虫。深层网络爬虫将Web页面按存在方式分为表层网页（Surface Web）和深层网页（Deep Web，也称Invisible Web Page或Hidden Web）。表层网页是指传统搜索引擎可以索引的页面，以超链接可以到达的静态网页为主构成的Web页面。深层网页是那些大部分内容不能通过静态链接获取的、隐藏在搜索表单后的、只有用户提交一些关键词才能获得的Web页面。深层网络爬虫体系结构包含6个基本功能模块（爬行控制器、解析器、表单分析器、表单处理器、响应分析器、LVS控制器）和两个爬虫内部数据结构（URL列表、LVS表）。

5.1.4数据采集方法

4. Scrapy爬虫

Scrapy是一套基于Twisted的异步处理框架，是纯Python实现的爬虫框架，用户只需要定制开发几个模块就可以轻松的实现一个爬虫，用来抓取网页内容或者各种图片。Scrapy运行于Linux/Windows/MacOS等多种环境，具有速度快、扩展性强、使用简便等特点。即便是新手，也能迅速学会使用Scrapy编写所需要的爬虫程序。

Scrapy可以在本地运行，也能部署到云端实现真正的生产级数据采集系统。Scrapy用途广泛，可以用于数据挖掘、监测和自动化测试。Scrapy吸引人的地方在于它是一个框架，任何人都可以根据需求对它进行修改。

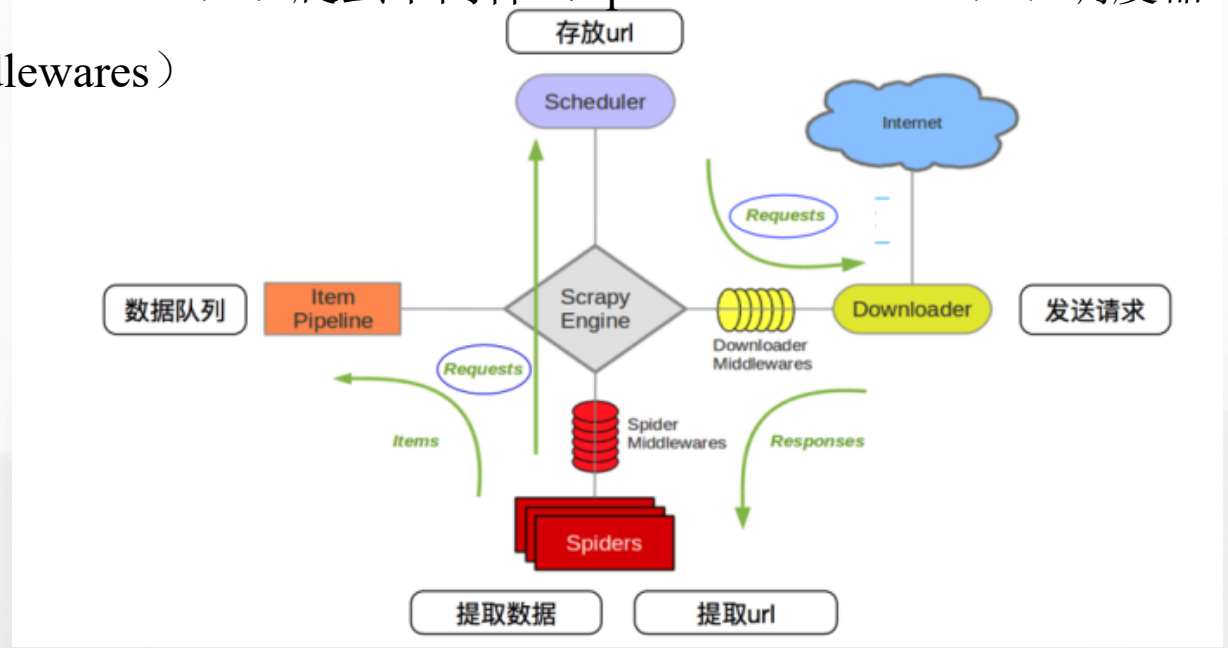
当然，Scrapy只是Python的一个主流框架，除了Scrapy外，还有其他基于Python的爬虫框架，包括Crawley、Portia、Newspaper、Python-goose、Beautiful Soup、Mechanize、Selenium和Cola灯。

5.1.4数据采集方法

4. Scrapy爬虫

（1）Scrapy体系架构

Scrapy体系架构包括以下组成部分： Scrapy引擎（Engine）、爬虫（Spiders）、下载器（Downloader）、调度器（Scheduler）、项目管道（Item Pipeline）、下载器中间件（Downloader Middlewares）、爬虫中间件（Spider Middlewares）、调度器中间件（Scheduler Middlewares）



5.1.4数据采集方法

4. Scrapy爬虫

(2) Scrapy工作流

Scrapy工作流也叫作“运行流程”或叫作“数据处理流程”，整个数据处理流程由Scrapy引擎进行控制，其主要的运行步骤如下：

①Scrapy引擎从调度器中取出一个链接（URL）用于接下来的抓取；

②Scrapy引擎把URL封装成一个请求并传给下载器；

③下载器把资源下载下来，并封装成应答包；

④爬虫解析应答包；

⑤如果解析出的是项目，则交给项目管道进行进一步的处理；

⑥如果解析出的是链接（URL），则把URL交给调度器等待抓取。

5.1.4数据采集方法

5.反爬机制

为什么会有反爬机制？原因主要有两点：第一，在大数据时代，数据是十分宝贵的财富，很多企业不愿意让自己的数据被别人免费获取，因此，很多企业都为自己的网站运用了反爬机制，防止网页上的数据被爬走；第二，简单低级的网络爬虫，数据采集速度快，伪装度低，如果没有反爬机制，它们可以很快地抓取大量数据，甚至因为请求过多，造成网站服务器不能正常工作，影响了企业的业务开展。

反爬机制也是一把双刃剑，一方面可以保护企业网站和网站数据，但是，另一方面，如果反爬机制过于严格，可能会误伤到真正的用户请求，也就是真正用户的请求被错误当成网络爬虫而被拒绝访问。如果既要和“网络爬虫”死磕，又要保证很低的误伤率，那么又会增加网站研发的成本。

5.2 数据清洗

5.2.1数据清洗的内容

5.2.2 数据清洗的注意事项

5.2.1数据清洗的内容

图 数据清洗的内容

5.2.1数据清洗的内容

数据清洗的内容主要包括

（1）缺失值处理。由于调查、编码和录入误差，数据中可能存在一些缺失值，需要给予适当的处理。常用的处理方法有：估算、整例删除、变量删除和成对删除。（a）估算：最简单的办法就是用某个变量的样本均值、中位数或众数代替缺失值。这种办法简单，但没有充分考虑数据中已有的信息，误差可能较大。另一种办法就是根据调查对象对其他问题的答案，通过变量之间的相关分析或逻辑推论进行估计。例如，某一产品的拥有情况可能与家庭收入有关，可以根据调查对象的家庭收入推算拥有这一产品的可能性。（b）整例删除：剔除含有缺失值的样本。由于很多问卷都可能存在缺失值，这种做法的结果可能导致有效样本量大大减少，无法充分利用已经收集到的数据。因此，只适合关键变量缺失，或者含有异常值或缺失值的样本比重很小的情况。（c）变量删除：如果某一变量的缺失值很多，而且该变量对于所研究的问题不是特别重要，则可以考虑将该变量删除。这种做法减少了供分析用的变量数目，但没有改变样本量。（d）成对删除：是用一个特殊码(通常是9、99、999等)代表缺失值，同时保留数据集中的全部变量和样本。但是，在具体计算时只采用有完整答案的样本，因而不同的分析因涉及的变量不同，其有效样本量也会有所不同。这是一种保守的处理方法，最大限度地保留了数据集中的可用信息。

5.2.1数据清洗的内容

（2）异常值处理。根据每个变量的合理取值范围和相互关系，检查数据是否合乎要求，发现超出正常范围、逻辑上不合理或者相互矛盾的数据。例如，用1-7级量表测量的变量出现了0值，体重出现了负数，都应视为超出正常值域范围。SPSS、SAS、和Excel等计算机软件都能够根据定义的取值范围，自动识别每个超出范围的变量值。具有逻辑上不一致性的答案可能以多种形式出现：例如，许多调查对象说自己开车上班，又报告没有汽车；或者调查对象报告自己是某品牌的重度购买者和使用者，但同时又在熟悉程度量表上给了很低的分值。发现不一致时，要列出问卷序号、记录序号、变量名称、错误类别等，便于进一步核对和纠正。

5.2.1数据清洗的内容

（3）数据类型转换。数据类型往往会影响到后续的数据处理分析环节，因此，需要明确每个字段的数据类型，比如，来自A表的“学号”是字符型，而来自B表的字段是日期型，在数据清洗的时候就需要对二者的数据类型进行统一处理。

（4）重复值处理。重复值的存在会影响数据分析和挖掘结果的准确性，所以，在数据分析和建模之前需要进行数据重复性检验，如果存在重复值，还需要进行重复值的删除。

5.2.2 数据清洗的注意事项

在进行数据清洗时，需要注意如下事项：

（1）数据清洗时优先进行缺失值、异常值和数据类型转换的操作，最后进行重复值的处理。

（2）在对缺失值、异常值进行处理时，要根据业务的需求进行处理，这些处理并不是一成不变的，常见的填充包括：统计值填充（常用的统计值有均值、中位数、众数）、前/后值填充（一般使用在前后数据存在关联，比如数据是按照时间进行记录的）、零值填充。

（3）在数据清洗之前，最为重要的对数据表的查看，要了解表的结构和发现需要处理的值，这样才能将数据清洗彻底。

5.2.2 数据清洗的注意事项

（4）数据量的大小也关系着数据的处理方式。如果总数据量较大，而异常的数据（包括缺失值和异常值）的量较少时，可以选择直接删除处理，因为这并不太会影响到最终的分析结果；但是，如果总数据量较小，则每个数据都可能影响分析的结果，这个时候就需要认真去对数据进行处理（可能需要通过其他的关联表去找到相关数据进行填充）。

（5）在导入数据表后，一般需要将所有列一个个地进行清洗，来保证数据处理的彻底性，有些数据可能看起来是正常可以使用的，实际上在进行处理时可能会出现问题（比如某列数据在查看时看起来是数值类型，但是其实这列数据的类型却是字符串，这就会导致在进行数值操作时无法使用）。

5.3 数据转换

5.3.1数据转换策略

5.3.2 平滑处理

5.3.3 规范化处理

5.3.1数据转换策略

常见的数据转换策略包括：

（1）平滑处理。帮助除去数据中的噪声，常用的方法包括分箱、回归和聚类等。

（2）聚集处理。对数据进行汇总操作。例如，每天的数据经过汇总操作可以获得每月或每年的总额。这一操作常用于构造数据立方体或对数据进行多粒度的分析。

（3）数据泛化处理。用更抽象（更高层次）的概念来取代低层次的数据对象。例如，街道属性可以泛化到更高层次的概念，如城市、国家，再比如年龄属性可以映射到更高层次的概念，如年轻、中年和老年。

（4）规范化处理。将属性值按比例缩放，使之落入一个特定的区间，比如0.0~1.0。常用的数据规范化方法包括Min-Max规范化、Z-Score规范化和小数定标规范化等。

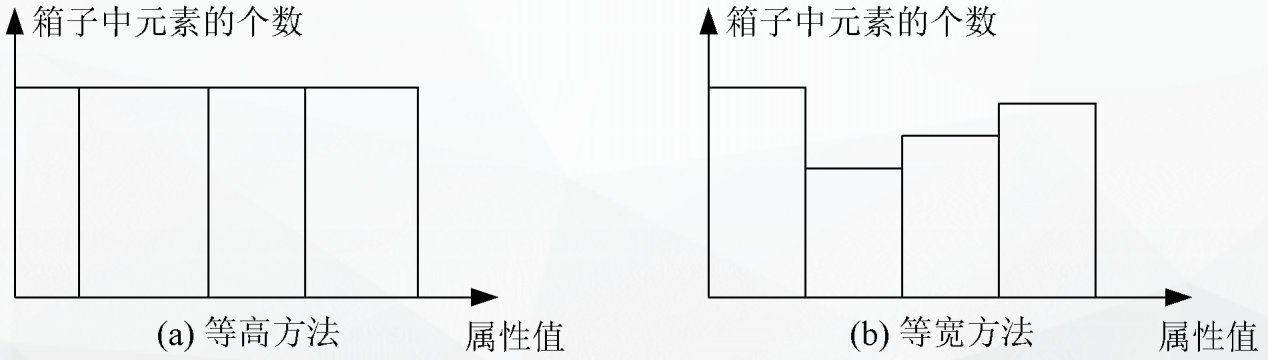
（5）属性构造处理。根据已有属性集构造新的属性，后续数据处理直接使用新增的属性。例如，根据已知的质量和体积属性，计算出新的属性——密度。

5.3.2 平滑处理

1.分箱

分箱（Bin）方法通过利用被平滑数据点的周围点（近邻），对一组排序数据进行平滑，排序后的数据被分配到若干箱子（称为 Bin）中。

如图5所示，对箱子的划分方法一般有两种，一种是等高方法，即每个箱子中元素的个数相等，另一种是等宽方法，即每个箱子的取值间距（左右边界之差）相同。



5.1.4数据采集方法

1.分箱

这里给出一个实例介绍分箱方法。假设有一个数据集X={4,8,15,21,21,24,25,28,34}，这里采用基于平均值的等高分箱方法对其进行平滑处理，则分箱处理的步骤如下：

（1）把原始数据集X放入以下三个箱子：

箱子1：4,8,15　　箱子2：21,21,24　　箱子3：25,28,34

（2）分别计算得到每个箱子的平均值：

箱子1的平均值：9　　箱子2的平均值：22　　箱子3的平均值：29

（3）用每个箱子的平均值替换该箱子内的所有元素：

箱子1：9,9,9　　箱子2：22,22,22　　箱子3：29,29,29

（4）合并各个箱子中的元素得到新的数据集{9,9,9,22,22,22,29,29,29}。

5.3.2 平滑处理

1.分箱

此外，还可以采用基于箱子边界的等高分箱方法对数据进行平滑处理。利用边界进行平滑时，对于给定的箱子，其最大值与最小值就构成了该箱子的边界，利用每个箱子的边界值（最大值或最小值）可替换该箱子中的所有值。这时的分箱结果如下：

箱子1：4,4,15

箱子2：21,21,24

箱子3：25,25,34

合并各个箱子中的元素得到新的数据集{4,4,15,21,21,24,25,25,34}。

5.3.2 平滑处理

2.回归

可以利用拟合函数对数据进行平滑。例如，借助线性回归方法（包括多变量回归方法），就可以获得多个变量之间的拟合关系，从而达到利用一个（或一组）变量值来预测另一个变量取值的目的。如图所示，利用回归分析方法所获得的拟合函数，能够帮助平滑数据并除去其中的噪声。

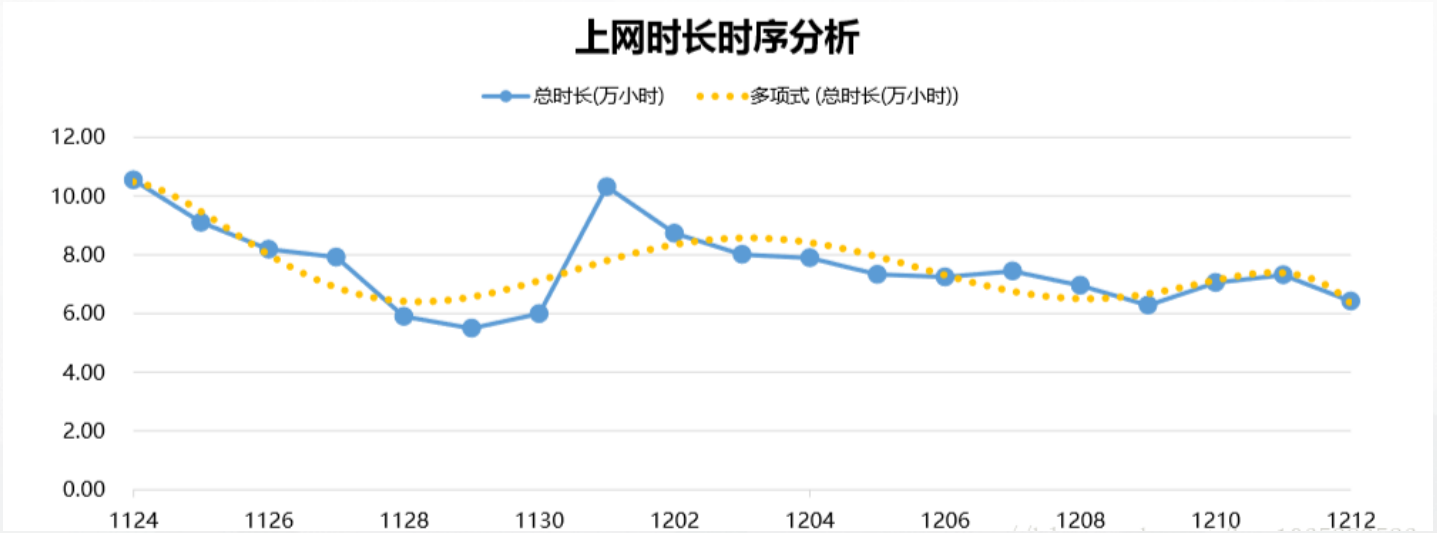


图 对数据进行线性回归拟合

5.3.2 平滑处理

3.聚类

通过聚类分析方法可帮助发现异常数据。如图所示，相似或相邻近的数据聚合在一起形成了各个聚类集合，而那些位于这些聚类集合之外的数据对象，则被认为是异常数据。

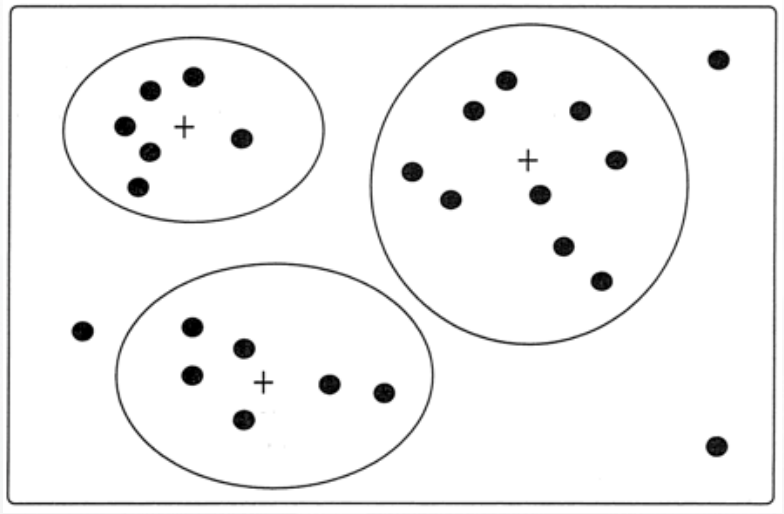


图 基于聚类分析方法的异常数据监测

5.3.3 规范化处理

1. Min-Max规范化

Min-Max规范化方法对被转换数据进行一种线性转换，其转换公式如下：

x=（待转换属性值−属性最小值）/（属性最大值−属性最小值）

例如，假设属性的最大值和最小值分别是87000元和11000元，现在需要利用Min-Max规范化方法，将“顾客收入”属性的值映射到0~1 的范围内，则“顾客收入”属性的值为72400元时，对应的转换结果如下：

(72400−11000)/(87000−11000) = 0.808

Min-Max规范化比较简单，但是也存在一些缺陷，当有新的数据加入时，可能导致最大值和最小值的变化，需要重新定义属性最大值和最小值。

5.3.3 规范化处理

2. Z-Score规范化

Z-Score规范化的主要目的就是将不同量级的数据统一转化为同一个量级，统一用计算出的Z-Score值衡量，以保证数据之间的可比性。其转换公式如下：

z=（待转换属性值−属性平均值）/属性标准差

假设我们要比较学生A与学生B的考试成绩，A的考卷满分是100分（及格60分），B的考卷满分是700分（及格420分）。很显然，A考出的70分与B考出的70分代表着完全不同的意义。但是从数值来讲，A与B在数据表中都是用数字70代表各自的成绩。那么如何能够用一个同等的标准来比较A与B的成绩呢？Z-Score就可以解决这一问题。

假设A班级的平均分是80，标准差是10，A考了90分；B班的平均分是400，标准差是100，B考了600分。通过上面的公式，我们可以计算得出，A的Z-Score是1（即（90-80）/10），B的Z-Socre是2（即（600-400）/100），因此，B的成绩更为优异。若A考了60分，B考了300分，则A的Z-Score是-2，B的Z-Score是-1，这时，A的成绩比较差。

5.3.3 规范化处理

2. Z-Score规范化

Z-Score的优点是不需要知道数据集的最大值和最小值，对离群点规范化效果好。此外，Z-Score能够应用于数值型的数据，并且不受数据量级的影响，因为它本身的作用就是消除量级给分析带来的不便。

但是Z-Score也有一些缺陷。首先，Z-Score对于数据的分布有一定的要求，正态分布是最有利于Z-Score计算的。其次，Z-Score消除了数据具有的实际意义，A的Z-Score与B的Z-Score与他们各自的分数不再有关系，因此，Z-Score的结果只能用于比较数据间的结果，数据的真实意义还需要还原原值。

5.3.3 规范化处理

3. 小数定标规范化

小数定标规范化方法通过移动属性值的小数位置来达到规范化的目的。所移动的小数位数取决于属性绝对值的最大值。其转换公式为：

x=待转换属性值/(10的k次方)

其中，k为能够使该属性绝对值的最大值的转换结果小于1的最小值。

比如，假设属性的取值范围是-957~924，则该属性绝对值的最大值为957，很显然，这时k=3。当属性的值为426时，对应的转换结果如下：

426/10的3次方 = 0.426

小数定标法的优点是直观简单，缺点是并没有消除属性间的权重差异。

5.4 数据脱敏

5.4.1数据脱敏原则

5.4.2数据脱敏方法

5.4.1数据脱敏原则

数据脱敏不仅要执行“数据漂白”，抹去数据中的敏感内容，同时也需要保持原有的数据特征、业务规则和数据关联性，保证开发、测试以及大数据类业务不会受到脱敏的影响，达成脱敏前后的数据一致性和有效性，具体如下：

（1）保持原有数据特征。数据脱敏前后必须保证数据特征的保持，例如：身份证号码由十七位数字本体码和一位校验码组成，分别为区域地址码（6 位）、出生日期（8 位）、顺序码（3 位）和校验码（1 位）。那么身份证号码的脱敏规则就需要保证脱敏后依旧保持这些特征信息。

（2）保持数据之间的一致性。在不同业务中，数据和数据之间具有一定的关联性。例如：出生年月或年龄和出生日期之间的关系。同样，身份证信息脱敏后仍需要保证出生年月字段和身份证中包含的出生日期之间的一致性。

5.4.1数据脱敏原则

（3）保持业务规则的关联性。保持数据业务规则的关联性是指数据脱敏时数据关联性以及业务语义等保持不变，其中数据关联性包括：主外键关联性、关联字段的业务语义关联性等。特别是高度敏感的账户类主体数据，往往会贯穿主体的所有关系和行为信息，因此需要特别注意保证所有相关主体信息的一致性。

（4）多次脱敏之间的数据一致性。相同的数据进行多次脱敏，或者在不同的测试系统进行脱敏，需要确保每次脱敏的数据始终保持一致性，只有这样才能保障业务系统数据变更的持续一致性以及广义业务的持续一致性。

5.4.2数据脱敏方法

数据脱敏的方法主要包括：

（1）数据替换。用设置的固定虚构值替换真值。例如将手机号码统一替换为13900010002。

（2）无效化。通过对数据值的截断、加密、隐藏等方式使敏感数据脱敏，使其不再具有利用价值，例如将地址的值替换为“******”。数据无效化与数据替换所达成的效果基本类似。

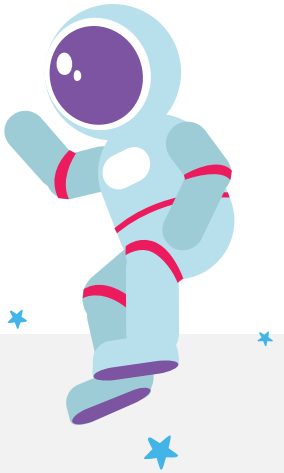
（3）随机化。采用随机数据代替真值，保持替换值的随机性以模拟样本的真实性。例如用随机生成的姓和名代替真值。

（4）偏移和取整。通过随机移位改变数字数据，例如把日期“2018-01-02 8:12:25”变为“2018-01-02 8:00:00”。偏移取整在保持了数据的安全性的同时，保证了范围的大致真实性，此项功能在大数据利用环境中具有重大价值。

（5）掩码屏蔽。掩码屏蔽是针对账户类数据的部分信息进行脱敏时的有力工具，比如银行卡号或是身份证号的脱敏。比如，把身份证号码“220524199209010254”替换为“220524********0254”。

（6）灵活编码。在需要特殊脱敏规则时，可执行灵活编码以满足各种可能的脱敏规则。比如用固定字母和固定位数的数字替代合同编号真值。

大数据与统计系 | 大数据系列课程



大数据

大数据与统计系数字教师网
http://hssj.wxcgi.com/