高等院校计算机应用系列教材

案例驱动式数据采集与预处理

陈苏红　主编

清華大學出版社
北　京

内容简介

数字经济飞速发展，“数据”是驱动行业创新、企业决策与社会进步的核心战略资源，被誉为“未来的石油”。海量数据的价值挖掘需要高效的采集、预处理等技术，这是大数据人才的核心能力，行业创新与企业决策也依赖高质量的数据采集与预处理等技术，因此相关技术的系统学习与实践尤为重要。通过本书即可开展数据采集、预处理、分析与可视化相关技术的系统学习。

本书共 9 章，内容包括概述、静态网页爬取、动态网页爬取、提升网络爬虫效率、Scrapy 框架初探、Scrapy 框架深入、系统日志数据采集、ETL 工具 Kettle、数据分析与可视化。本书构建了从数据基础概念到高级应用的完整知识体系，系统讲解 Python 数据采集与处理全流程技术，助力读者快速掌握数据采集、预处理、存储、分析及可视化的核心技术与实战技巧。

本书沿着“案例 + 知识”这一主线，以问题为导向，采用任务驱动的模式推进。除第 1 章概述外，每章通过案例贯穿，将案例设计为多个迭代版本，每个版本解决一两个问题，以提出问题、学习知识、解决问题为主脉络。读者在学习本书时，沿着清晰的案例路径，即可将理论与实践相结合，快速掌握相关技能。每章具有较完整的知识体系和真实的项目案例，章节中的“练一练”和“课后习题”可以帮助读者及时巩固所学知识，拓展知识的深度与广度。

本书覆盖了数据采集领域的常见数据源，如互联网数据、日志文件、企业业务系统数据等，适合作为大数据、人工智能、计算机、软件工程等相关专业的教材，也可作为数据采集与处理领域从业人员的实战参考用书。

图书在版编目(CIP)数据

案例驱动式数据采集与预处理 / 陈苏红主编 .
北京 : 清华大学出版社 , 2026. 7. -- (高等院校计算机
应用系列教材). -- ISBN 978-7-302-71983-0

Ⅰ . TP274

中国国家版本馆 CIP 数据核字第 20260UE981 号

责任编辑：刘金喜
封面设计：高娟妮
版式设计：思创景点
责任校对：成凤进
责任印制：刘海龙

出版发行：清华大学出版社
网　　址：https://www.tup.com.cn，https://www.wqxuetang.com
地　　址：北京清华大学学研大厦 A 座　　邮　　编：100084
社 总 机：010-83470000　　邮　　购：010-62786544
投稿与读者服务：010-62776969，c-service@tup.tsinghua.edu.cn
质 量 反 馈：010-62772015，zhiliang@tup.tsinghua.edu.cn
印 装 者：三河市东方印刷有限公司
经　　销：全国新华书店
开　　本：185mm×260mm　　**印　　张：**22.25　　**字　　数：**584 千字
版　　次：2026 年 8 月第 1 版　　**印　　次：**2026 年 8 月第 1 次印刷
定　　价：79.80 元

产品编号：108576-01

前　言

在数字经济飞速发展的今天，“数据”已成为驱动行业创新、企业决策和社会进步的核心战略资源，被誉为“未来的石油”。然而，海量数据的价值挖掘离不开高效的采集、清洗、处理、分析与可视化技术，这也成为当前大数据领域人才亟须掌握的核心能力。为帮助读者系统掌握数据采集与处理的全流程技术，我们结合多年教学经验与工程实践，编写了这本案例驱动式教材。

本书是特色应用型课程“数据采集与预处理”的配套教材，以Python为核心编程语言，遵循“学以致用”理念，继承了课程建设团队此前出版的湖北省一流本科课程配套教材《案例驱动式Python基础与应用》的设计思路：以实际案例为主线、以问题为导向、以任务驱动模式推进。除第1章概述外，其余各章均以一个真实项目案例贯穿，每个案例设计2～4个层层递进的迭代版本，每个版本聚焦解决一两个核心问题，形成“提出问题—引出所需知识点—学习知识—运用知识解决问题”的完整闭环，让读者在实践中理解技术原理、掌握实战技能。

本书共9章，构建了从数据基础概念到高级应用的完整知识体系，系统讲解Python数据采集与处理全流程技术，助力读者快速掌握数据采集、预处理、存储、分析及可视化的核心技术与实战技巧。

第1章为概述，介绍数据与大数据的基本概念、类型、组织形式、价值及处理流程，同时讲解Python IDLE与PyCharm的安装配置，为后续学习奠定基础。

第2章聚焦静态网页爬取，以“贝壳网房源信息爬取”为案例，详细讲解网络爬虫的概念、工作流程，以及使用requests、urllib库获取网页数据，通过BeautifulSoup、XPath解析数据，并将结果保存为CSV、JSON文件的方法。

第3章探讨动态网页爬取，以“人民邮电出版社图书信息爬取”为案例，对比逆向分析法与Selenium模拟法的技术特点，并系统介绍动态网页的判定方法、核心数据接口定位、参数逆向分析，以及将爬取的数据存储到MySQL和MongoDB数据库的实现方案。

第4章围绕爬虫效率提升展开，以“证券之星股票数据爬取”为案例，讲解进程与线程的概念及区别，通过单进程、多进程(Process类和Pool进程池)、多线程技术优化爬虫性能，并对比不同方案的效率差异。

第5章介绍Scrapy框架的初步使用，以“古诗文网古诗名句爬取”为案例，讲解Scrapy框架的结构、工作流程、项目创建、爬虫编写、数据容器定义及数据存储方法，帮助读者快速上手框架开发。

第6章深入Scrapy框架应用，以“豆瓣网电影排行榜全站电影信息爬取”为案例，拓展框架高级特性，包括CrawlSpider爬虫实现全网站爬取、Scrapy动态网页爬取、中间件开发、反爬策略应对等，提升爬虫的健壮性与扩展性。

第7章专注系统日志数据采集，以“百里半教育平台用户行为数据采集”为案例，介绍日

志采集工具Flume的原理、安装与使用，重点讲解Flume采集系统端口数据、采集日志文件到HDFS、采集MySQL数据到HDFS的具体方法。

第8章以“电商用户商品偏好分析”为案例，介绍Kettle的安装配置、核心组件(如转换与作业)，以及数据抽取、转换、加载的实战流程，实现不同数据源的整合与处理。

第9章以“电商销售数据分析”为案例，讲解使用pandas行数据获取、清洗(如重复值、缺失值、异常值处理)、分析(如描述性统计、排序、分组与聚合等)，并通过折线图、柱状图、散点图等Python可视化工具展示数据规律，挖掘数据价值。

本书注重理论与实践结合，案例贯穿每个章节，每章均配套真实项目案例，代码详细可复现，适合作为大数据、人工智能、计算机、软件工程相关专业的教材，也可作为数据采集与处理领域从业人员的实战参考用书。无论是零基础的初学者，还是有一定基础的技术爱好者，都能通过本书循序渐进地掌握数据采集与处理的核心技术，提升解决实际问题的能力。

本书由陈苏红主编、统稿和校订。其中，第 1至6 章及第 8 章由陈苏红编写，第 7 章由付忠旺、陈苏红编写，第 9 章由金兰编写。本书为特色应用型课程“数据采集与预处理”配套教材，是本校与武汉厚溥数字科技有限公司开展校企协同建设的落地成果。企业教师深度参与课程方案设计、内容开发与教材编撰，充分体现产教融合的建设思路。在此，谨向校企双方相关领导的鼎力支持致以谢意，同时向课程建设团队金兰和付忠旺老师的辛勤付出，表示由衷感谢。

为便于教学，本书配有教学课件、教学大纲、授课教案、微课视频、案例源代码、习题答案及教材配套软件，读者可通过扫描下方二维码下载。


教学课件


教学大纲


授课教案


微课视频


案例源代码


习题答案


配套软件

编写过程中，我们得到了众多同行的帮助，也参阅了大量相关资料，在此衷心感谢给予支持和帮助的同事、朋友以及相关领域的专家学者，同时也感谢读者的选择与支持。

由于技术发展迅速，书中难免存在疏漏之处，恳请广大读者批评指正。

服务邮箱：476371891@qq.com、wkservice@vip.163.com。

编者

2026年4月

目 录

第 1 章

概　述

俗话说“得数据者得天下”，大数据本身就是一种资源，蕴藏着价值，但沉睡的资源是很难创造价值的，它必须经过采集、清洗、处理、分析、可视化等加工处理过程，才能真正产生价值。数据采集和预处理是其中具有关键意义的第一个环节。通过数据采集，我们可以获取传感器、互联网、日志文件、企业业务系统等数据，采集得到的数据需要进行预处理，数据预处理包括数据清洗、数据转换和数据脱敏等操作，以用于后续的数据分析与可视化。数据清洗可对数据文件中明显的错误值、缺失值、异常值、可疑数据，选用适当方法进行“清理”，使“脏”数据变为“干净”数据，有利于后续的统计分析得出可靠的结论。数据转换是把原始数据转换成符合目标算法要求的数据。数据脱敏的目的是实现对敏感数据的可靠保护。

在工程应用中，数据采集与预处理的对象是数据，乃至大数据，因此我们先要从了解数据入手，包括数据的概念、数据类型和组织形式；进而了解大数据，包括大数据来源、大数据采集的概念和采集方法；然后再按数据处理的一般流程，基于已经采集到的数据进行存储和清洗、转换、脱敏等预处理工作；最后对数据进行分析与可视化。本章系统介绍数据、大数据、大数据采集、存储、预处理、分析及可视化的相关概念，为后续各章节做铺垫。另外，本书的所有代码及实验都在Windows操作系统(Windows 10或以上版本)下完成，并且采用Python作为编程语言。因此，本章最后给出了数据采集、预处理、分析与可视化的基础软件Python IDLE与PyCharm的安装与使用方法，其他软件根据案例的需要在后续章节单独给出安装与使用方法。

1.1　数据

本节介绍数据的概念、数据类型、数据组织形式和数据的价值。

1.1.1　数据的概念

数据是对客观事物的性质、状态及相互关系等进行记载的物理符号或这些物理符号的组合，这些符号是可识别的、抽象的。它既指狭义上的数字，也可以是具有一定意义的文字、字母、数字符号的组合、图形、图像、视频、音频等，还是客观事物的属性、数量、位置及其相互关系的抽象表示，如“0、1、2…”“阴、雨、下降、气温”“学生的档案记录、货物的运输情况”等都是数据。数据经过加工后就成为信息，信息与数据既有联系，又有区别。数据是信息的载体，数据的排列组合即呈现信息，信息是较为宏观的概念，传达某种概念或方法，而数据则是构成信息的基本单位。从信息论的观点来看，描述信源的数据是信息和数据冗余之和，

即：数据=信息+数据冗余。由此可见，信息可以简单地理解为数据中包含的有用的内容。

随着人类社会信息化进程的加快，我们的日常生产和生活中每天都在不断产生大量的数据。数据已经渗透到当今每一个行业和业务职能领域，成为重要的生产要素。数据推动着企业的发展，支撑着企业的决策，使得各级组织的运营更为高效，是构成企业核心竞争力的关键要素。数据资源和物质资源、人力资源一样，已成为国家的重要战略资源，影响着国家和社会的安全、稳定与发展，因此被称为“未来的石油”。

1.1.2 数据类型

在计算机中，常见的数据类型包括文本、图片、音频、视频等。

1. 文本

文本数据也称字符型数据，是指不能参与算术运算的字符。在计算机中，文本数据一般保存在文本文件中。文本文件是一种由若干行字符构成的计算机文件，常见格式包括ASCII、MIME 和 TXT。

2. 图片

图片数据是指由图形、图像等构成的平面媒体。在计算机中，图片数据一般用图片格式的文件来保存。图片的格式很多，大体可以分为点阵图和矢量图两大类，我们常用的 BMP、JPG 等格式都是点阵图，而Flash动画制作软件所生成的SWF和Photoshop绘图软件所生成的PSD等格式属于矢量图。

3. 音频

数字化的声音数据就是音频数据。在计算机中，音频数据一般用音频文件来保存。音频文件是指存储声音内容的文件，把音频文件用一定的音频程序播放，就可以还原以前录下的声音。音频文件的格式很多，包括 CD、WAV、MP3、MID、WMA、RA等。

4. 视频

视频数据是指连续的图像序列。在计算机中，视频数据一般用视频文件来保存。视频文件常见的格式有MPEG-4、AVI、DAT、RM、MOV、ASF、WMV、DivX等。

1.1.3 数据组织形式

数据组织是按照一定的方式和规则对数据进行归并、存储、处理的过程，涉及数据存储和管理的具体方式。在计算机系统中，数据组织包括文件和数据库两种主要形式。

1. 文件

文件是最基本的数据组织形式之一，是具有文件名的一组相关信息的集合，用于存储和管理数据。文件可以是文本文件、图像文件、音频文件等，它们根据数据的性质和用途进行分类和组织。计算机系统中的很多数据都是以文件形式存在的，如Word文件、文本文件、网页文件、图片文件等。一个文件的文件名包含主名和扩展名。其中，扩展名用来表示文件的类型，如文本文档、图片、音频、视频等。在计算机中，文件是由文件系统负责管理的。

2. 数据库

计算机系统中另一种非常重要的数据组织形式就是数据库。数据库是按照数据结构来组织、存储与管理数据的仓库，是一个长期存储在计算机内、有组织的、可共享的、统一管理的大量数据的集合。数据库的存储空间很大，可同时供多个用户读写。数据库中的数据需要遵循一定规则，这些规则能够使用户更合适地组织数据、更方便地维护数据、更严密地控制数据和更有效地利用数据。今天，数据库已经成为计算机软件开发的基础和核心。数据库在人力资源、固定资产、制造业、电信、销售、银行管理等领域发挥着至关重要的作用。从1968年IBM公司推出第一个大型商用数据库管理系统IMS开始到现在，数据库先后经历了层次数据库、网状数据库和关系型数据库等阶段，数据库技术在各个方面都实现了高速发展。特别是关系型数据库已经成为目前数据库产品中最重要的一员，20世纪80年代以来，几乎所有的数据库厂商新出的数据库产品都支持关系型数据库。随着云计算的发展和大数据时代的到来，越来越多的半关系型和非关系型数据需要进行存储管理；同时，海量数据使得单个数据中心无法存储所有的数据，所以需要使用分布式存储架构进行数据存储。于是越来越多的非关系型数据库(Not only SQL，NoSQL)开始出现，它们有更好的高并发读写性能，也更利于存储海量数据。

1.1.4 数据的价值

数据的价值主要在于可以为人们提供答案。

首先，数据具有描述价值。对于数据收集者来说，通过对数据进行初步加工，可以得到关于事物的描述性信息。例如，企业的业务数据在被收集之后，可以加工处理为报表。对业务人员来说，这些报表描述了业务发展的状况，使他们能够对自己负责的业务有量化、全面的认识，进而调整工作策略；而对管理层来说，业务数据提供了企业的发展信息，有助于管理人员做出更符合企业利益的决策。

其次，数据具有时间价值。在过去，数据一旦实现了基本用途，往往就会被删除，这一方面是由于当时的存储技术落后，人们需要删除旧数据来存储新数据，另一方面则是因为人们没有认识到数据的潜在价值。实际上，数据的价值会随着时间维度的累积而越来越高。例如，通过分析历史销售情况，能够得到消费者对产品的偏好、产品所处的生命周期、产品的销售季节规律等信息。这些信息对产品的营销和研发具有指导价值。

再次，数据具有预测价值。通过历史数据，配合特定算法，可以实现对未来数据的预测。例如，通过历史购买数据来为客户推荐其未来可能购买的商品，通过历史销售数据来预测未来的市场需求。

随着信息技术快速发展，日常生活日益信息化、智能化，数据不会因为不断被使用而削减价值，反而会因为不断重组而产生更大的价值。例如，将一个地区的物价和地价走势、高档轿车的销售数量、二手房转手的频率、出租车密度等各种不相关的数据整合到一起，可以更加精准地预测该地区房价走势。这种方式已经被国外很多房地产网站所采用。而这些被整合过的数据，下一次还可以出于别的目的而重新整合。也就是说，数据没有因为被使用一次或两次而造成价值的衰减，反而会在不同的领域产生更多的价值。

人类进入信息社会以后，数据以自然方式增长，其产生不以人的意志为转移。从1986年到2010年的20多年时间里，全球数据的数量增长了100倍，今后的数据量增长速度将更快，我们

正生活在一个“数据爆炸”的时代。在1分钟内，新浪微博新增约2万条微博，推特新增约10万条推文，App Store完成约4.7万次应用下载，淘宝售出约6万件商品，百度处理约90万次搜索查询。在数据爆炸的今天，人类一方面对知识充满渴求，另一方面为数据的复杂特征而困惑。数据爆炸对科学研究提出了更高的要求，人类需要设计出更加灵活高效的数据存储、处理和分析工具，来应对大数据时代的挑战，这必将带来云计算、数据仓库、数据挖掘等技术和应用的提升或根本性改变。

练一练

【练一练1-1单选题】下列选项中，最准确描述“数据”概念的是(　　)。

A. 数据是存储在计算机中的文字信息

B. 数据是未经加工的原始记录，它可以是数字、文字、图像、声音等多种形式

C. 数据是指通过统计分析后得出的结论或知识

D. 数据是只能以二进制形式存储在计算机中的信息

【练一练1-2单选题】下列选项中，最全面概括数据基本类型的是(　　)。

A. 文本数据和数值数据

B. 字符数据和二进制数据

C. 静态数据和动态数据

D. 文本、图片、音频、视频

【练一练1-3单选题】下列选项中，最准确描述数据组织形式的是(　　)。

A. 结构化数据、半结构化数据、非结构化数据

B. 文本和图像

C. 字符与二进制

D. 静态和动态

【练一练1-4单选题】下列选项中，最准确阐述数据价值的是(　　)。

A. 数据本身具有直接的经济价值

B. 数据的价值在于其数量，数量越多价值越高

C. 数据的价值主要体现在通过分析和挖掘获得的洞察和决策支持

D. 数据的价值仅在于其准确性和完整性

1.2　大数据

大数据，也称巨量资料或海量数据。IBM提出了大数据的5V特点：Volume(大量)、Velocity(高速)、Variety(多样)、Value(低价值密度)、Veracity(真实性)。大数据是以容量大、类型多、存取速度快、应用价值高为主要特征的数据集合。大数据也可以定义为来自各种源头的大量非结构化或结构化数据，其中非结构化数据越来越成为数据的主要部分。IDC的调查报告显示，企业中80%的数据都是非结构化数据，这些数据呈指数级增长，年增速达60%。

大数据时代，人们常讲“三分技术、七分数据”“得数据者得天下”，大数据的获取显得尤为重要，也是大数据分析与应用的前提。因此，大数据采集是重要且非常必要的。

1.2.1 大数据采集

数据采集，又称数据获取，是数据分析的前提，也是数据分析过程中相当重要的一个环节。传统的数据采集是指利用一种装置，从系统外部采集数据并传输到系统内部。数据采集技术广泛应用于各个领域，如摄像头、麦克风、传感器等。被采集的数据是已被转换为电信号的各种物理量，如温度、水位、风速、压力等，可以是模拟量，也可以是数字量。采集一般使用采样方式，即每隔一定时间(即采样周期)对同一点的数据重复采集。采集的数据大多是瞬时值，也可以是某段时间内的一个特征值。

大数据采集是大数据分析及应用的第一步，但大数据采集的重点却不在于数据本身，而在于如何能够利用数据真正地解决商业运营中的实际问题。只有对所需数据进行全面、准确的采集，形成一定规模，再对这些数据进行分析，所得出的结果才能对决策行为起到指导性作用。

在互联网行业快速发展的今天，大数据采集更多的是指从传感器、智能设备、企业信息系统、社交网络和互联网平台等获取数据的过程。大数据的来源包括RFID(Radio Frequency Identification，射频识别)、传感器、用户行为、社交网络交互及移动互联网等。大数据的类型包括结构化、半结构化及非结构化等。数据源的种类多，数据的类型繁杂，数据量大，而且数据产生的速度快，传统的数据采集方法完全无法胜任，这就需要采用针对大数据的采集技术。

大数据采集与传统的数据采集既有联系又有区别。大数据采集是在传统的数据采集基础之上发展起来的，一些经过多年发展的数据采集架构、技术和工具都被继承下来。同时，由于大数据本身具有数据量大、数据类型丰富、处理速度快等特性，因此大数据采集又表现出不同于传统数据采集的一些特点，如表1-1所示。

表1-1 传统数据采集与大数据采集的区别

	传统数据采集	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，包括结构化数据、半结构化数据和非结构化数据
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

1.2.2 大数据来源

大数据的来源非常多样化，可按数据的产生主体、类型、来源系统进行划分。

1. 按数据的产生主体划分

数据的产生主体非常多样，主要包括以下几个方面。

(1) 个人用户：通过社交媒体、博客、电子邮件、即时通信等方式生成大量用户生成内容(User Generated Content，UGC)。

(2) 企业：在生产、运营、销售等过程中产生各种业务数据。

(3) 政府机构：在公共服务、社会治理等方面产生大量数据。

(4) 物联网设备：如智能家居设备、智能城市设施、工业生产设备等，通过传感器实时采集各种物理量数据。

2. 按数据的类型划分

(1) 结构化数据：具有固定格式和有限长度，并可以通过数据库二维表来表达的数据，如企业内部的财务数据、销售数据等。

(2) 非结构化数据：数据格式不固定，无法用数据库二维表来表达的数据，如社交媒体文章、视频、音频、图片等。

(3) 半结构化数据：具有一定的结构，但并不完全符合数据库二维表的形式，如电子邮件、日志文件等。

3. 按数据的来源系统划分

数据来源系统是指用于收集和存储大数据的系统，主要包括以下几种。

(1) 企业信息系统：如ERP、CRM等，用于收集和存储企业内部的业务数据。

(2) 物联网平台：用于收集和处理物联网设备产生的数据，如智能家居平台、智能城市管理平台等。

(3) 社交媒体平台：如微博、微信、抖音等，用户在这些平台上可生成大量UGC数据。

(4) 电子商务平台：如淘宝、京东等，用户在购物过程中可产生大量交易数据。

(5) 大数据处理平台：如Hadoop、Spark等，用于处理和分析大规模数据。

(6) 科学实验系统：围绕某个具体的研究主题，利用实验主动产生所需数据，或者利用模拟的方式直接获得仿真数据。

大数据采集的主要数据源包括传感器数据、互联网数据、日志文件、企业业务系统数据等。

(1) 传感器数据。

传感器是一种能够感知物理状态或化学状态的设备，广泛应用于工业自动化、智能家居、环境监测等领域。传感器可以采集各种类型的数据，如温度、湿度、压力、流量等，这些数据对于监控生产过程、预测设备故障和维护设备正常运行等方面具有重要意义。

(2) 互联网数据。

互联网中各类网站及社交媒体平台都提供了丰富的数据资源，包括网页内容、新闻文章、用户评论、图片、视频等。这些互联网数据一方面来源于网站发布，另一方面来源于互联网用户生成。这些数据可以通过相应的平台方提供的数据接口得到。如果没有数据接口，则需要采用网络爬虫技术、工具来自动抓取。所谓“网络爬虫”，是在遵循一定的爬虫协议下，在互联网上到处或定向抓取网页数据的程序。该程序实现的一般方法是，定义一个初始页面，该页面通常会包含指向其他页面的统一资源定位符(Uniform Resource Locator，URL)，于是这些网址加入爬虫的抓取队列，进入新页面后再递归地进行上述操作。网络爬虫首先抓取目标网页的全部数据，然后按预定的规则解析提取局部的目标数据，最后将数据存储起来，以备后续的数据操作。

(3) 日志文件。

大数据行业中，许多公司的业务平台每天都会产生大量日志文件。日志文件一般由业务系统产生，用于记录对业务系统的各种操作，包括系统日志、应用程序日志、安全日志、数据库日志及Web访问日志等。其中：系统日志记录系统级别的事件，如系统启动、关闭、错误消息；应用程序日志记录特定应用程序的运行情况，包括错误、警告、信息性消息；安全日志记录安全相关事件，如登录尝试、权限变更、安全策略执行等；数据库日志记录数据库事务、查询、错误等信息；Web访问日志记录网站访问数据内容，如HTTP请求和响应、状态码、网络故障等。这些日志中埋藏着巨大的价值，是决策支持系统的重要数据来源。

(4) 企业业务系统数据。

企业业务系统数据是企业运营过程中产生和存储的各类数据的集合，这些数据对于企业的决策支持、业务优化、故障排查和审计等方面都具有重要作用。一些企业会使用传统的关系型数据库MySQL和Oracle或非关系型数据库Redis、MongoDB来存储业务系统数据。企业每时每刻产生的业务数据，以一行记录的形式被直接写入数据库。除直接从业务系统数据库取得数据进行分析外，也可以采用构建数据仓库的模式，为企业决策提供数据源。数据仓库是一个面向主题的、集成的、相对稳定的、反映历史变化的数据集合。企业可以借助ETL(Extract-Transform-Load，抽取—转换—加载)工具，将分散在企业不同业务系统中的数据抽取、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。通过采集不同业务系统的数据并将其统一保存到一个数据仓库中，可以为分散在企业不同位置的商务数据提供统一的视图，从而满足企业各种商务决策分析的需求。

在采集企业业务系统数据时，首先要明确数据需求，包括数据类型、字段及采集频率等。其次要设计数据采集方法。由于数据种类复杂，在进行数据分析前，必须通过数据抽取技术从数据的原始格式中抽取出需要的数据，丢弃一些不重要的字段。然后，要对数据进行清洗和验证。数据清洗包括去除重复项、处理缺失值和异常值、调整数据格式和类型等；数据验证包括验证数据的准确性和一致性。必要时还需对数据进行转换，将数据转换成不同的格式。最后，按照数据需求或预定义的数据仓库模型，将数据加载到数据仓库中。

1.2.3 大数据采集方法

不同来源、不同类型的数据有不同的采集方式。例如，摄像头、温度计等机器数据可以通过传感器获取，企业根据其性质将其转化为数据资产或作为隐私数据进行相应处理；分布在各网络平台和网页上的大量数据，可以通过网络爬虫进行数据抓取；公开数据，如国家开放数据、企业公共数据、个人共享数据等，可通过公共平台下载；企业内部数据可分为可转化为资产的数据和内部隐私数据，其中可转化为资产的数据可通过企业洽谈、购买等方式获取，而内部隐私数据通常不能直接获取，甚至某些企业在进行内部运营系统开发时，会要求工程师驻场开发并签订保密协议。

归纳起来，大数据的采集主要分为如下主要途径。

(1) 传感器设备数据采集。通过传感器、摄像头和其他智能终端自动采集信号、图片或录像来获取数据，如温度、湿度、压力等物理量的实时获取，为数据分析提供丰富的数据源。

(2) 网络数据采集。通过网络爬虫或网站公开API等方式从互联网、新闻媒体、学术文献等公共数据源上获取数据信息，适用于结构化和非结构化数据的采集，能够有效获取大量、实时的互联网数据。

(3) 业务系统数据库数据采集。通过直接连接数据库获取数据，适用于企业内部或公共数据库的数据采集，如获取MySQL、Oracle等传统关系型数据库，以及Redis、MongoDB和HBase等非关系型数据库中的数据。

(4) 系统日志数据采集。系统日志可以分为用户行为日志、业务变更日志、系统运行日志3种。其中，用户行为日志记录了用户在系统中的操作行为信息；业务变更日志记录了用户使用某种功能后对业务(对象、数据)进行操作的信息；系统运行日志记录了服务器资源、网络资源等的运行情况。例如，Hadoop的Chukwa、Cloudera的Flume、Facebook的Scribe等都是系统日志

采集工具。系统日志数据采集可从系统或应用程序的日志文件中提取运行状态、用户行为等信息，对于分析系统性能、用户行为具有重要意义。

(5) 数据仓库导入。数据仓库是从各个业务数据库中导入的结构化数据集合，用于支持更复杂的查询和分析操作，为企业提供了深入的数据洞察和预测能力，是大数据时代企业底层基建的重要组成部分。

练一练

【练一练1-5单选题】大数据采集的核心挑战和关键步骤是(　　)。

A. 确保数据的实时性，快速从各种来源抓取数据

B. 数据的准确性和完整性是采集过程中唯一重要的因素

C. 大数据采集不需要考虑数据的来源和格式多样性

D. 数据的存储和备份是大数据采集的主要任务

【练一练1-6单选题】在大数据采集中，数据的来源通常不包括(　　)。

A. 社交媒体平台(如微博、Facebook)

B. 企业内部数据库和事务系统

C. 物联网设备(如智能传感器、RFID标签)

D. 未经授权的个人计算机和移动设备

【练一练1-7单选题】下列中不属于常见的大数据采集方法的是(　　)。

A. 日志采集法　　B. 网络爬虫技术

C. 数据仓库导入　　D. 手工录入法

1.3 数据存储

在实际应用中，采集到的目标数据需要持久化存储，以便后期数据处理与分析。数据存储主要有文件存储和数据库存储两种方式。其中，文件存储包括普通文件存储和分布式文件存储(如HDFS，即Hadoop Distributed File System)；数据库存储包括常见的关系型数据库(如MySQL)和非关系型数据库(如MongoDB或Redis)。

1. 文件存储

普通文件存储是一种基础的数据存储方式，它将数据以文件的形式存储到本地计算机中，比较适合中小规模的数据存储。而分布式文件存储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远程存储节点(一台计算机)相连，或者由若干不同的逻辑磁盘分区或卷标组合在一起，形成一个完整的、有层次的文件系统。分布式文件存储是实现大规模数据集高效存储的关键。

2. 数据库存储

文件存储方式虽然能够对采集的数据进行存储，但会将数据保存为一堆零散的文件，缺乏清晰的结构，不利于后续对数据的进一步处理。当采集的数据量较大时，可以使用数据库进行存储。数据库不仅可以对数据进行分类保存，还能有效避免存储重复的数据，甚至提供高效的检索和访问方式。

根据存储数据时所用数据规模不同，当今互联网中的数据库主要分为关系型数据库和非关系型数据库两种，具体如下。

关系型数据库是采用关系模型(二维表格形式)组织数据的数据库系统，由数据表和数据表之间的关系组成，主要包含数据行、数据列、数据表、数据库4个核心元素。

非关系型数据库也被称为NoSQL数据库，通常是一种非关系型、分布式的数据存储系统。与关系型数据库相比，非关系型数据库无须事先为要存储的数据建立字段，没有固定的结构，既可以包含不同的字段，也可以存储各种格式的数据。其种类繁多，按照不同的数据模型，可以分为列存储数据库、键值存储数据库和文档型数据库。其中，键值存储数据库采用键值结构存储数据，每个键分别对应一个特定的值，其典型代表是Redis；文档型数据库的结构与键值存储数据库类似，它采用文档(如JSON或XML等格式)结构存储数据，每个文档中包含多个键值对，其典型代表是MongoDB。

在工程应用中，文件存储和数据库存储各有利弊。文件存储比较适合中小型数据采集项目，数据库存储比较适合大型数据采集项目，我们可以根据实际需求进行选择。本书后面也会在具体案例中介绍将数据存储到文件和数据库中的方法。

1.4 数据预处理

数据预处理(Data Preprocessing)是对原始数据进行的一系列处理过程，旨在为后续的数据分析工作提供可靠的、高质量的数据，以减少数据分析过程中的错误和偏差。数据预处理的主要步骤包括数据清洗、数据集成、数据转换和数据脱敏等，如图1-1所示。经过这些步骤，我们可以从大量的数据属性中提取出一部分对目标输出有重要影响的属性，降低源数据的维度，去除噪声，为数据分析算法提供干净、准确且有针对性的数据，减少数据分析算法的数据处理量，改进数据质量，提高分析效率。

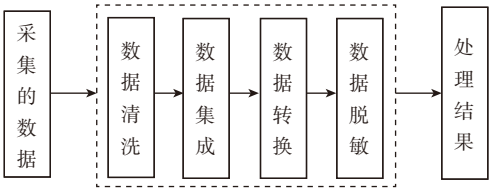

图1-1 数据预处理

1. 数据清洗

数据清洗是数据预处理的第一步，主要目的是去除数据中的噪声、异常值、缺失值、重复值，并对数据类型有误的数据进行处理。

(1) 异常值处理。识别并处理数据中的极端或不合理值，这些值可能对数据分析结果产生不良影响。异常值的处理方法包括删除、替换为平均值或中位数，或者使用更复杂的统计或机器学习方法进行处理。

(2) 缺失值处理。处理数据中的缺失或未知值。常见的处理方法包括删除含有缺失值的记录、使用统计值(如均值、中位数)进行填充，或者通过插值方法(如回归插补、多重插补)进行估算。

(3) 重复值处理。识别并删除数据中的重复记录，以避免在后续分析中产生误导。

(4) 数据类型转换。数据类型往往会影响后续的数据分析环节，因此，需要明确每个字段的

数据类型。例如，来自A表的“学号”是字符型，而来自B表的该字段是日期型，那么需要对两者的数据类型进行统一处理。

2. 数据集成

数据集成是将来自不同数据源的数据合并到一个数据存储中的过程。这通常需要解决数据的不一致性、冗余性及冲突检测与消除问题。

(1) 数据一致性。各个数据源中的实体数据与现实世界中的实体匹配模式保持一致。同一实体的属性在不同的数据库中可能命名不同，识别并统一这种差异是十分必要的。

(2) 数据冗余。识别并删除数据中的重复信息，以降低数据存储和处理的复杂性。

(3) 数据冲突检测与消除。同一实体的属性在不同数据源中采用的单位、编码等可能不同。这种语义差异是数据集成需要重点检测并解决的问题。

3. 数据转换

数据转换是将数据进行转换、归并或规范化处理，以改变数据的分布或满足特定分析方法的要求。常见的数据转换策略如下。

(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，对每天的数据进行汇总操作可以获得每月或每年的总额。聚集处理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；再如，年龄属性可以映射到更高层次的概念，如青年、中年和老年。

(4) 归一化/标准化。将数据转换为统一的尺度，以便于不同数据之间的比较。例如，使用z-score标准化将数据转换为均值为0、标准差为1的分布。

(5) 离散化。将连续型数据转换为离散型数据，以便于某些分类算法的处理。

(6) 属性构造处理。根据已有的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。

(7) 其他转换。如对数变换、指数变换、平方根变换等，以改善数据的分布特性。

4. 数据脱敏

数据脱敏(Data Masking)，又称数据去隐私化或数据变形，是在保留数据原始特征的条件下，根据给定的规则、策略按需对敏感数据进行变换、修改的技术机制，以防止这些重要数据在共享和移动时发生泄露，旨在保护敏感数据在非可信环境中的安全使用，使原有数据的使用范围和共享对象得以拓展，是大数据环境下最有效的敏感数据保护方法之一。

数据脱敏通常遵循以下几条原则。

(1) 不可逆性。脱敏算法通常应是不可逆的，以防止使用非敏感数据推断或重建敏感的原始数据。但在一些特定场合，也存在可恢复式数据脱敏的需求。

(2) 数据特征保留。脱敏后的数据应保留原数据的大部分特征，以便继续用于开发或测试等场合。

(3) 引用完整性。如果被脱敏的字段是数据表主键，那么与之相关的引用记录必须同步更改。

(4) 全面脱敏。对所有可能生成敏感数据的非敏感字段也应进行脱敏处理。

(5) 自动化与可重复性。脱敏过程应是自动化且可重复的，以适应数据的变化。

数据脱敏的常用方法包括但不限于以下几种。

(1) 替换法。将特定字符或字符序列替换为其他符号(如星号“*”)或特定字符串。例如，电话号码脱敏后可能呈现为“(123)456-*”，身份证号码则变为“3301*****3456”。

(2) 掩码法。保留数据的部分特征以维持其基本结构，如信用卡号常保留前六位(发卡机构标识)和后四位(校验码)，中间部分用星号或其他字符替换。

(3) 加密法。利用复杂的数学算法对敏感数据进行编码，使其在未授权情况下无法解读。常见的加密算法包括对称加密(如AES)、非对称加密(如RSA)和哈希函数(如SHA)。

(4) 数据扰动。在数据集中引入微小、随机的变化，使个体数据点难以被识别，但保持整体数据分布、相关性和趋势不变。

(5) 数据分割。将敏感数据拆分为多个部分，并分别存储在不同的物理或逻辑位置，以降低单一攻击导致数据泄露的风险。

(6) 伪数据生成。创建与真实数据在统计特性上高度相似但无实际对应关系的假数据，用于测试、开发等非生产环境。

如今，数据脱敏广泛应用于需要保护敏感信息的各个行业，尤其是金融、政府和医疗行业。这些行业在应用开发、测试、培训等活动中普遍使用真实数据，导致数据在暴露期间面临严重的泄露风险。通过数据脱敏，企业能够根据数据使用目标，制定精确、灵活的脱敏策略，从而实现在不同工具、应用程序和环境中的快速、一致的访问限制。

随着技术的不断进步，数据脱敏技术也在不断发展和完善。现代数据脱敏系统通常具备自动化、智能化、可扩展等特点，能够支持大规模数据集的快速脱敏处理，并满足不同场景下的个性化需求。同时，随着数据保护法规的日益严格，数据脱敏在保障数据安全、促进数据合规性方面发挥着越来越重要的作用。

总之，数据脱敏是保护敏感数据的重要手段之一，通过遵循一定的原则和采用合适的方法，可以有效地降低数据泄露风险，保障数据安全。

1.5 数据分析

数据分析是指用适当的统计和分析方法，对收集到的大量数据进行分析，将其加以汇总、理解和消化，并提取出有用信息，进而进行详细研究和概括总结的过程。该过程旨在最大化地开发数据的功能，发挥数据的作用。

常见的数据分析方法如下。

(1) 描述性统计分析。通过概括和总结数据集的主要特征来提供对数据的直观理解。

(2) 探索性数据分析(EDA)。通过绘图和统计手段深入理解数据集的结构、特征和模式。

(3) 推理统计学。从样本中得出关于总体的信息，包括参数估计和假设检验。

(4) 回归分析。研究自变量与因变量之间的关系，揭示自变量的变化如何影响因变量的变化。

(5) 聚类分析。将数据集中的观察值划分为相似的组(簇)，以发现数据中的内在结构和模式。

数据分析在各个领域都有广泛的应用，包括商业、金融、医疗保健、教育等，具体如下。

(1) 商业决策。帮助企业制定精准的产品策略和营销策略。

(2) 金融风险评估。评估借款人的信用风险，制定贷款政策。

(3) 医疗保健。预测疾病发生率，评估治疗效果，优化治疗方案。

(4) 教育行业。评估学生学习效果，预测学生未来表现，制订教学计划。

总之，数据分析是一个复杂而重要的过程。它能够帮助企业和组织从数据中提取有价值的信息，指导业务决策，优化业务流程，提高运营效率，并实现商业智能和预测。随着大数据技术的不断发展，数据分析将在更多领域发挥更为重要的作用。

1.6 数据可视化

数据可视化是关于数据视觉表现形式的科学技术研究，是通过图形和图像处理、计算机视觉及用户界面等技术手段，将大型数据集中的数据以图形图像形式表示，并利用数据分析和开发工具发现其中未知信息的处理过程。

数据可视化方法丰富，根据其可视化原理的不同，可划分为基于几何的技术、面向像素的技术、基于图标的技术、基于层次的技术、基于图像的技术和分布式技术等。常见的数据可视化工具包括折线图、柱状图、饼图、散点图、热力图和地图等，每种类型都有其特定的应用场景和优势，具体如下。

(1) 折线图。用于显示数据随时间或其他连续变量的变化趋势，是分析时间序列数据的常用工具。

(2) 柱状图。用于比较不同类别或组之间的数据大小，能够清晰地展示各类别之间的数量差异。

(3) 饼图。用于显示不同类别或组所占比例的相对大小，适合展示整体与部分之间的关系。

(4) 散点图。用于显示两个变量之间的关系，如相关性或聚类，是探索性数据分析中的重要工具。

(5) 热力图。用于显示矩阵数据的强度、密度或分布情况，常用于展示大量数据的密集程度和分布情况。

(6) 地图。用于显示地理位置相关的数据，如人口分布、销售地点等，是地理空间数据可视化的重要手段。

数据可视化的应用场景十分广泛，涵盖商业、科学、医学、金融、社会学、智慧城市、金融风控等诸多领域。它能够将复杂、零散的原始数据直观呈现，帮助使用者快速理解数据内涵，精准挖掘数据中隐藏的规律、变化趋势与有效信息，可用于展示市场动态、销售业绩、网络流量、生物研究数据等各类数据内容。

数据可视化依托丰富多样的图表类型实现数据展示，以适配不同数据的分析与展示需求。在大数据与人工智能技术持续迭代的背景下，数据可视化技术也在不断升级和完善。未来，数据可视化将朝着实时化、交互化、个性化的方向持续发展，更好地适配用户多元化、高质量的数据分析需求。随着技术的持续进步与行业应用的不断拓展，数据可视化的应用边界将持续拓宽，在各行业的数据处理中发挥更为重要的支撑作用。

结合行业实际应用情况，数据可视化的典型应用场景主要有以下几个。

(1) 商业智能。通过数据可视化，企业可以分析销售数据、市场趋势和消费者行为等信息，为制定营销策略和产品开发提供决策支持。

(2) 科学研究。科研人员可以利用数据可视化工具展示实验结果和数据分析结果，帮助同行更好地理解和验证研究结论。

(3) 社会调查。社会学家可以通过数据可视化展示社会现象和人口统计数据等信息，揭示社会问题的本质与规律。

(4) 医疗健康。医疗机构可以利用数据可视化工具监测患者的生命体征和疾病发展趋势等信息，为临床诊断与治疗提供辅助支持。

(5) 智慧城市。城市管理者可以利用可视化技术监控交通流量、空气质量等关键指标，提升城市管理效率。

(6) 金融风控。银行和金融机构利用大数据可视化监测交易行为，识别欺诈模式，控制金融风险。

随着大数据技术的不断发展和应用场景的不断拓展，大数据可视化将呈现以下发展趋势。

(1) 智能化。结合人工智能和机器学习技术，提升数据可视化的自动化和智能化水平。

(2) 实时性。实现数据的实时采集、处理和可视化展示，满足对实时性要求较高的应用场景。

(3) 移动化。支持移动设备访问和交互，提升数据可视化的便捷性和灵活性。

(4) 个性化。根据用户需求和数据特点，提供个性化的数据可视化解决方案。

综上所述，大数据领域的数据可视化是数据科学中的重要组成部分，它通过图形化手段将复杂数据转化为易于理解和分析的视觉形式，为各个领域提供了强大的数据分析和决策支持能力。随着技术的不断发展和应用场景的不断拓展，大数据可视化将在未来发挥更加重要的作用。

练一练

【练一练1-8单选题】在数据存储中，(　　)通常用于处理大规模数据集，并支持高效的数据检索和分析。

A. 关系数据库管理系统(RDBMS)　　B. 分布式文件系统(如HDFS)

C. 文本文件　　D. 文档型数据库存储

【练一练1-9单选题】在数据预处理阶段，通常用于识别和纠正数据中的错误或不一致性的是(　　)。

A. 数据清洗　　B. 数据集成

C. 数据转换　　D. 数据脱敏

【练一练1-10单选题】在数据分析过程中，主要用于探索数据中的模式、趋势和关系，而无须事先指定模型的方法是(　　)。

A. 回归分析　　B. 聚类分析

C. 决策分析　　D. 描述性统计分析

【练一练1-11单选题】在数据可视化中，(　　)最适合用于展示数据随时间的变化趋势。

A. 条形图　　B. 饼图

C. 折线图　　D. 散点图

1.7 基础软件安装与运行

基于Python的数据采集、分析与可视化所涉及的软件、工具及Python第三方库众多，但最基础的软件是Python IDLE及其集成开发工具PyCharm。在预备章节中，需将这两个软件安装、测试并配置好，后续相关软件则根据实际需要进行安装与配置。

官方Python解释器是一个跨平台的Python集成开发环境，支持Windows、macOS和Linux操

作系统。目前官方发布的最新版本是3.13.5，但考虑到本地操作系统的版本、Python运行的稳定性及Python第三方库的兼容性，本书选择安装Python 3.9.9。

1.7.1 Python 3.9.9的安装与使用

1. Python 3.9.9的下载与安装

访问Python官网的下载页面，如图1-2所示。找到 Python 3.9.9版本，根据计算机是32位还是64位下载对应的安装包。通常情况下，64位的操作系统居多，应下载文件名形如python-3.9.x-amd64.exe的安装包。

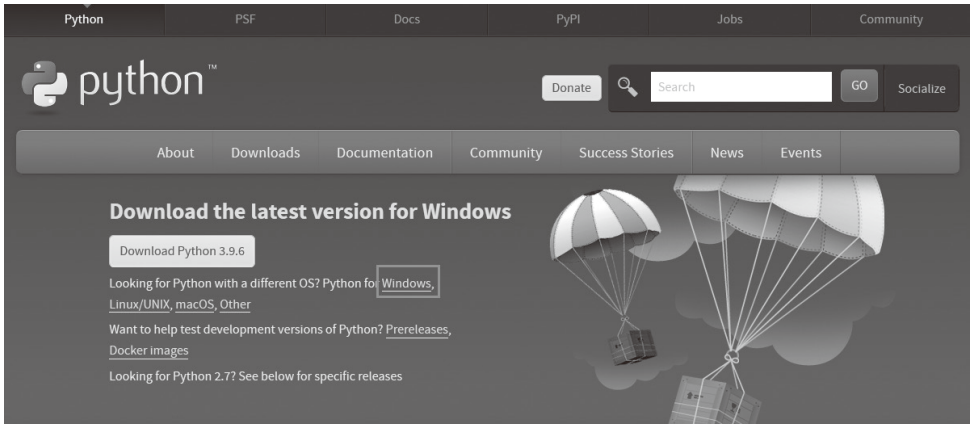

图1-2　Python官方下载页面

单击图1-2中的Windows超链接，进入Windows版本的下载页面，根据操作系统版本选择相应的安装包。此处选用Python 3.9.9的64位安装包。

下载完成后，找到安装包文件 python-3.9.9-amd64 ，右击，选择“以管理员身份运行”。

在图1-3所示的安装向导界面中，系统提供了“默认安装”和“自定义安装”两种方法。为便于后续在命令行中直接调用Python，需选中“Add Python 3.9 to PATH”复选框，该操作会自动将Python安装路径添加到系统环境变量中，省去手动配置的步骤。

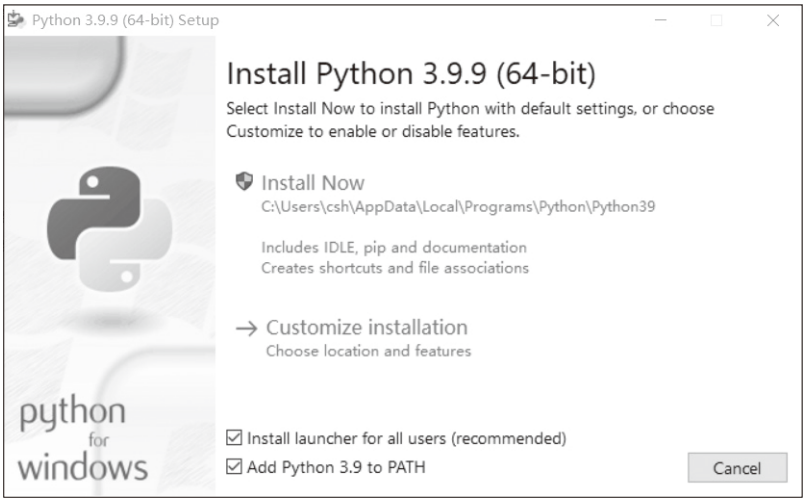

图1-3　选择Customize installation(自定义安装)并勾选Add Python 3.9 to PATH

完成后，单击Customize installation(自定义安装)进入下一步，以便根据需求调整安装组件和路径。

进入自定义安装页面后，如图1-4所示，保留所有默认选中的功能选项，单击Next按钮进入下一步。

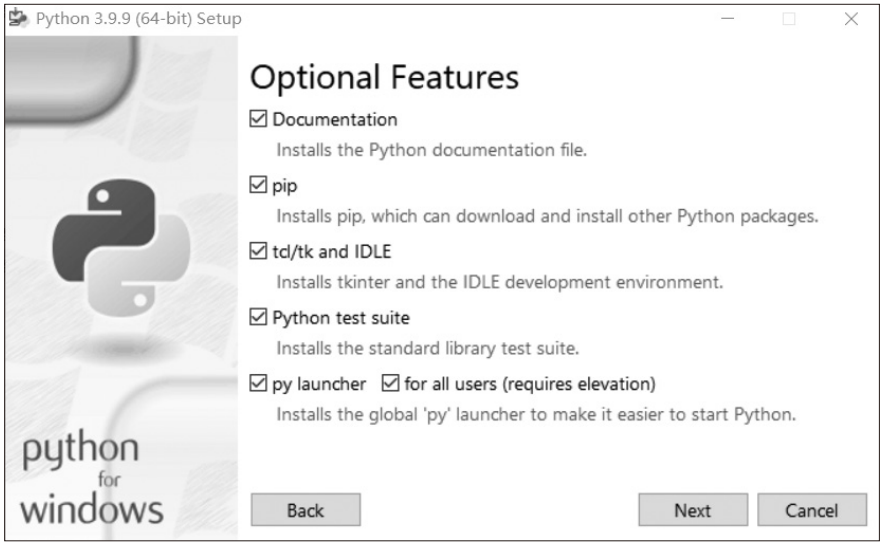

图1-4　保留默认选中的功能选项

进入下一个安装页面后，选中Install for all users(为所有用户安装)以确保多账户可用；同时需指定安装路径(如D:\Python39)，注意路径中避免包含中文字符或空格，以防后续运行时出现路径识别问题。完成设置后，单击Install按钮，启动安装程序。待安装进度条完成后，系统会提示安装成功，如图1-5所示，单击Close按钮，关闭安装向导即可。

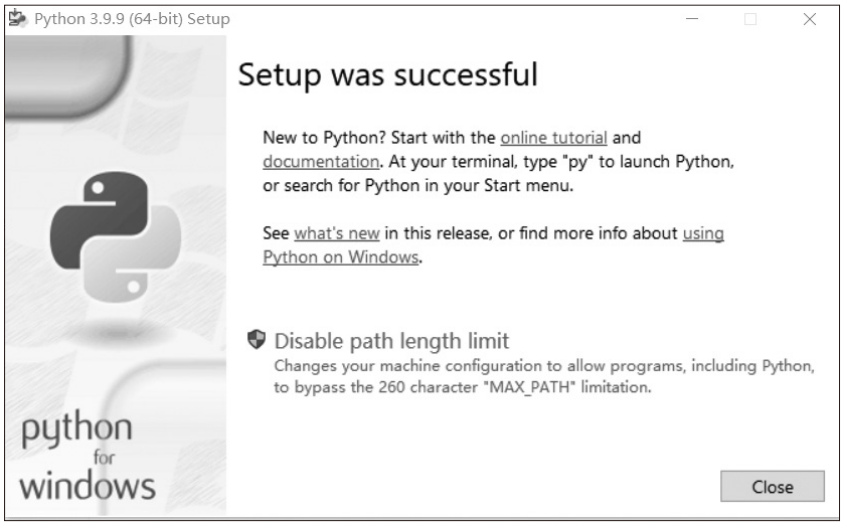

图1-5　Python安装成功

2. 安装成功的测试

(1) 命令行使用。按“Win+R”快捷键，打开“运行”对话框，输入cmd并回车，打开命令提示符。在命令提示符中输入python，进入 Python 交互式环境，当出现“>>>”提示符后，即

可输入Python代码，例如输入print('Hello, World!')并回车，将会输出相应内容，如图1-6所示。在Python交互式环境中输入exit()，即可退出交互式环境。

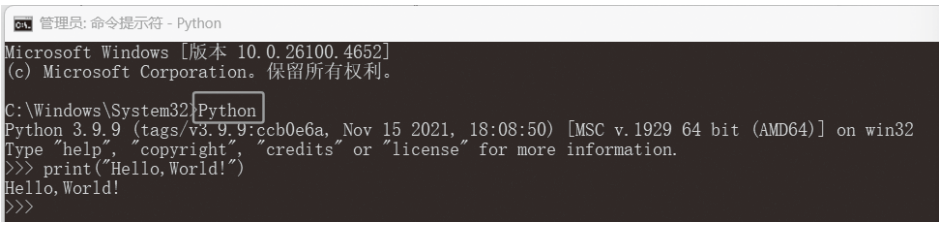

图1-6　命令行执行python指令

(2) 使用IDLE编辑器。单击“Windows开始菜单”，找到Python 3.9文件夹，再单击IDLE (Python 3.9 64 - bit)。默认打开的是Python Shell界面，可逐行输入语句并执行，如图1-7所示。若要编写多行代码程序，可单击File→New File(或按“Ctrl+N”快捷键)创建一个空白编辑器，编写完代码后，单击Run→Run module(或按F5快捷键)运行，此时系统会提示保存文件，将其保存为.py格式即可。

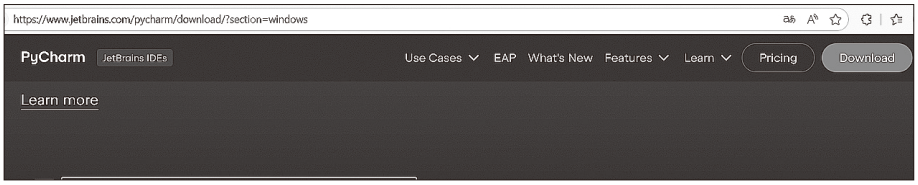

图1-7　IDLE集成开发环境窗口

1.7.2　PyCharm的安装与使用

Python IDLE仅适用于编写小型脚本或单个Python程序的开发，而对于复杂的、包含多个Python文件的项目集成开发，则需要使用集成开发环境PyCharm。

PyCharm是由JetBrains公司开发的一款专为Python语言设计的集成开发环境(Integrated Development Environment，IDE)。凭借强大的功能、智能的代码辅助和丰富的工具支持，它已成为全球 Python 开发者广泛使用的主流工具之一，具有如下优点。

1. 智能代码辅助

提供实时语法检查、自动补全(支持变量、函数、模块及第三方库)、代码重构(重命名、提取函数等)功能，大幅提升编码效率。

内置对Python语法的深度理解，能识别潜在错误并给出修复建议，减少调试时间。

2. 全面的项目管理

支持大型项目的目录结构管理，可便捷地组织模块、包及资源文件。

集成版本控制工具，如Git、SVN、Mercurial等，方便团队协作和代码提交、分支管理。

3. 强大的调试与测试工具

可视化调试器，支持断点设置、单步执行、变量监控、调用栈查看等，帮助轻松定位代码问题。

内置单元测试框架，如unittest、pytest等，可直接运行测试用例并生成报告。

4. 丰富的扩展生态

支持海量插件，如Django、Flask框架工具、数据科学库(如Pandas、NumPy)集成、Markdown预览等，可按需扩展功能。

兼容多种开发场景，包括 Web 开发、数据科学、机器学习、自动化测试等。

5. 跨平台支持

可在 Windows、macOS、Linux 系统上稳定运行，保持一致的操作体验。

6. 集成终端与工具

内置命令行终端，可直接执行Python命令、虚拟环境操作或系统指令。支持Jupyter Notebook集成，方便数据科学开发者进行交互式计算和可视化。

另外，PyCharm分为以下两个主要版本，以满足不同需求。

- 社区版(Community)：免费开源，包含核心Python开发功能(代码补全、调试、基础项目管理等)，适合初学者、学生及小型项目开发者。
- 专业版(Professional)：付费版本(支持免费试用)，新增Web开发(Django/Flask)、数据库工具、远程开发、Docker/Kubernetes集成等高级功能，适合企业开发者、Web开发人员及数据科学家。

总之，PyCharm以“开箱即用”的便利性和对Python生态的深度适配，平衡了易用性与专业性。无论是新手入门还是资深开发者处理复杂项目，都能显著提升开发效率，是Python开发领域的标杆工具之一。

1.7.2.1 PyCharm的安装

1. 访问官网下载

首先，访问 PyCharm 官方网站：https://www.jetbrains.com/pycharm/。PyCharm 分为两个版本：专业版(Professional Edition)功能全面，包含Web开发、数据库工具等高级特性，但需付费使用(支持免费试用)；社区版(Community Edition)为免费版本，涵盖 Python 基础开发所需的核心功能，适合个人学习、小型项目开发等场景。

根据自身需求选择版本即可。此处以社区版为例，如图1-8所示，单击对应版本的Download按钮下载安装包(当前示例为社区版 2025.1.3.1)。需要注意的是，PyCharm 版本越新，功能可能越丰富，但占用的存储空间也越大，因此无须刻意追求最高版本，选择与自身开发需求相匹配的版本即可。

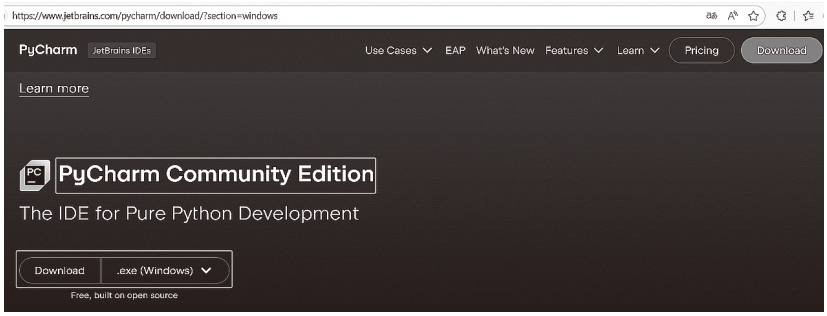

图1-8 下载PyCharm社区版

2. 安装PyCharm

下载完成后，双击安装包 pycharm-community-2025.1.3.1 开始安装。根据安装向导(见图1-9)进行安装，安装过程中可以选择安装路径(见图1-10)、是否创建桌面快捷方式(见图1-11)等选项。

图1-9　PyCharm安装向导

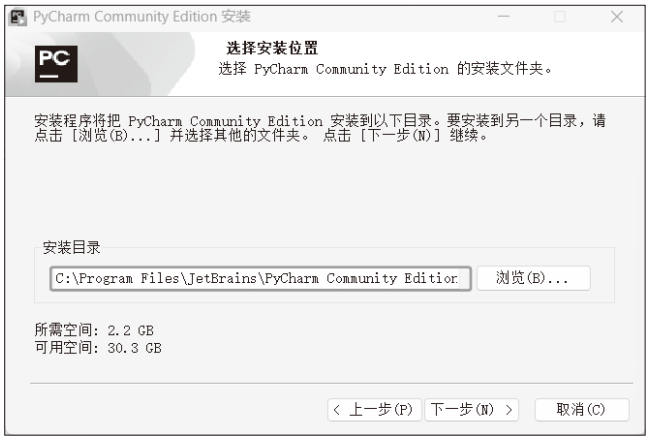

图1-10　设置安装路径(可默认)

图1-11　创建桌面快捷方式

接着按默认设置继续执行安装，安装完成后，桌面上会出现PyCharm的快捷方式。

1.7.2.2 PyCharm的使用

1. 首次启动

双击桌面上的PyCharm快捷方式启动程序。若为首次启动，可能会出现语言和区域选择，建议语言选择English(后续可在设置中更改语言)，区域选择China Mainland；还可能出现与系统区域相关的设置(如时间格式等)，保持默认或按需选择即可。完成初始设置后，将进入PyCharm欢迎主界面，如图1-12所示。

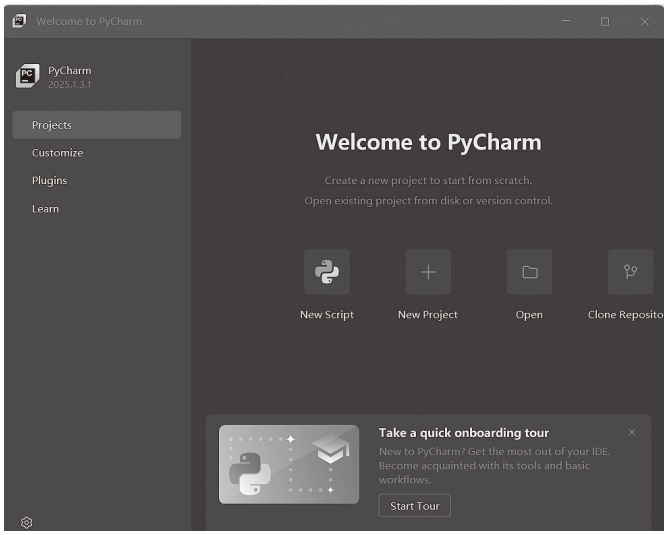

图1-12 PyCharm欢迎主界面

若单击New Project创建一个新项目，需先自定义项目名称，并指定存储路径(如D:\Project)，如图1-13所示。

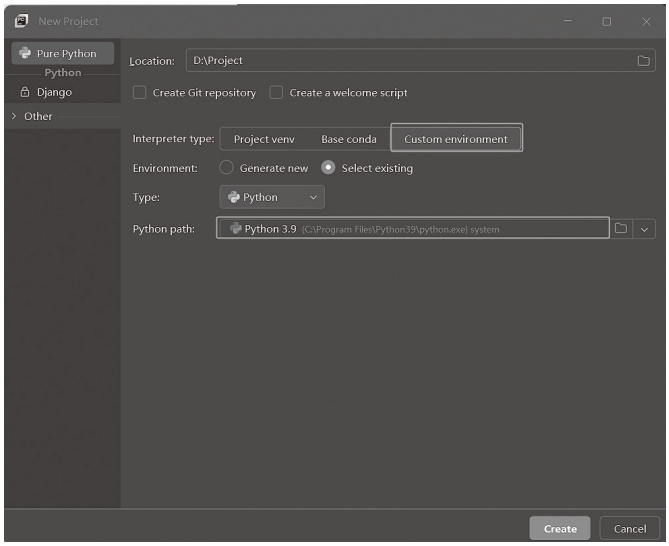

图1-13 设置项目存储路径

2. 创建和编辑文件

(1) 在项目视图中，右击项目名称，选择New→Python File创建一个新的Python文件。

(2) 输入文件名后按Enter键，文件即被创建。

(3) 双击文件打开编辑界面，输入Python代码。

3. 运行程序

在编辑界面中，右击代码编辑区域，选择“Run‘文件名’”运行程序。或者在代码编辑区域内右击，选择“Run‘当前文件名’”运行当前选中的文件。运行结果会在底部的Run窗口中显示。

4. 调试程序

(1) 在代码编辑区域左侧，单击行号旁的空白区域可以设置断点。

(2) 右击代码编辑区域，选择“Debug‘文件名’”启动调试模式。

(3) 程序会在断点处暂停，此时可以检查变量的值、单步执行代码等。

5. 安装和使用插件

PyCharm支持安装各种插件来增强其功能。

(1) 进入设置界面(单击File→Settings)，选择Plugins选项。

(2) 在Marketplace中搜索需要的插件，单击Install按钮进行安装。

(3) 安装完成后，根据插件的提示进行配置和使用。

通过这种方式安装插件具有安全性高、操作简便(无须手动配置)、版本兼容性自动适配、支持一键更新与卸载等优点，但也存在安装过程高度依赖网络环境、部分插件需付费、对小众需求支持有限及更新延迟等缺点。

6. 配置Python解释器和环境

(1) 在设置界面中，选择“Project: 项目名称”→Python Interpreter来配置Python解释器和环境，如图1-14所示。

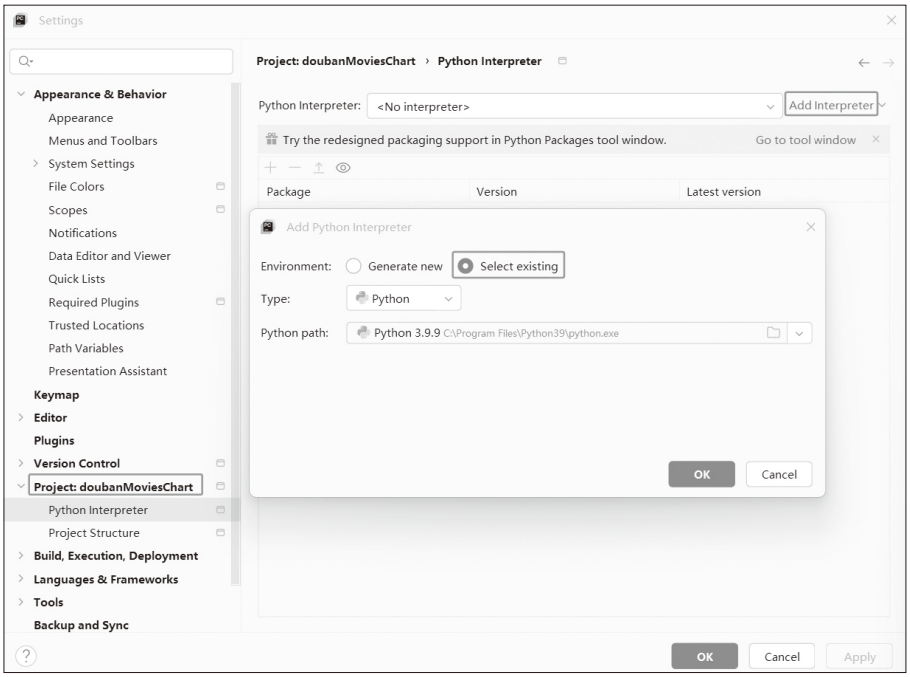

图1-14　配置Python项目的Python解释器和环境

(2) 单击Add Interpreter按钮添加新的解释器，或者选择已有的解释器。如果需要，还可以配置虚拟环境等高级选项。

7. 自定义设置

PyCharm提供了丰富的自定义设置选项，包括主题、字体大小、快捷键等。进入设置界面后，可以根据个人喜好进行配置。

练一练

【练一练1-12单选题】在Windows系统中安装Python时，若未选择Add Python to PATH选项，最可能导致的问题是(　　)。

A. 无法成功安装Python解释器

B. 在命令提示符中直接输入python无法启动Python交互式环境

C. 无法通过pip命令安装第三方库

D. Python脚本文件无法保存为.py 格式

【练一练1-13单选题】在安装PyCharm时，必要的步骤是(　　)。

A. 选择“商业版”进行下载

B. 跳过安装过程中的所有配置选项

C. 选择“将py文件与Pycharm关联”选项

D. 在安装完成后直接运行Python脚本

本章你学到了什么

本章我们主要介绍了以下内容。

- 数据的概念、类型、组织形式和价值。
- 大数据的概念、来源及采集方法。
- 数据存储、预处理、分析与可视化各个环节的主要工作及方法。
- 基础软件Python IDLE及PyCharm的安装与使用方法。

课后练习题

一、单项选择题

1. (　　)是指对客观事件进行记录并可以鉴别的符号。

A. 符号　　B. 信息

C. 数据　　D. 集合

2. (　　)是指从传感器和智能设备、企业在线系统、企业离线系统、社交网络和互联网平台等获取数据的过程。

A. 数据采集　　B. 数据预处理

C. 数据分析　　D. 数据可视化

3. (　　)的任务主要包括数据清洗、数据集成、数据转换和数据脱敏等。

A. 数据采集　　B. 数据预处理

C. 数据分析　　D. 数据可视化

4. 决定数据质量好坏的重要因素是(　　)。

A. 数据清洗　　B. 数据转换

C. 数据分析　　D. 数据可视化

5. (　　)数据转换策略可帮助去除数据中的噪声。

A. 泛化处理　　B. 规范化处理

C. 归一化处理　　D. 平滑处理

二、简答题

1. 简述数据采集的主要方式。

2. 简述数据预处理的含义，并列举几种常见的数据预处理方法。

三、操作题

1. 在自己的计算机上搭建Python+PyCharm环境。

2. 编写一个简单的Python程序，测试环境可用。

第 2 章

网络数据采集——静态网页爬取

案例1　贝壳网房源信息爬取：贝壳房源网(即贝壳找房平台)是一个居住产业数字化服务平台，致力于推进居住服务的产业数字化、智能化进程。

贝壳新房武汉市楼盘页面主要展示了该平台中武汉市挂牌出售的全部楼盘信息，包括楼盘名称、地址、每平方米均价、总价、户型、面积等，如图2-1所示。本案例通过爬取该网页来获取武汉房源信息，主要包括楼盘名称、地址、每平方米均价、总价，并将数据保存到文件houses_data.csv中，部分房源数据如图2-2所示。本案例将从获取网页数据、解析网页数据并提取房源信息、组织数据并保存3个主要步骤来完成房源信息的获取。

图2-1　贝壳新房武汉楼盘

	A	B	C	D	E
1	楼盘	地址	均价	总价	
2	恺德雲麓	武汉市硚口区古调路以西、田正街以北恺德雲麓项目	17000元/㎡	154-224(万/套)	
3	方岛金茂智慧科学城	四新大道与芳草路交汇处	19000元/㎡	410-475(万/套)	
4	龙湖御湖境	洪达巷龙湖·御湖境	32000元/㎡	345-745(万/套)	
5	中建锦绣双城	莲花湖大道与西环路交汇处	7300元/㎡	100-165(万/套)	
6	越秀汉阳星汇云锦	晴川大道越秀汉阳星汇云锦	25000元/㎡	300-880(万/套)	
7	千禧城	古田三路与解放大道交汇处	18000元/㎡	259-312(万/套)	
8	汉北玺园	汉口北大道东端南侧（汉口北大道胜海站）	9000元/㎡	87-126(万/套)	
9	佳阳翠湖里	平江大道以西（口岸联检大楼旁）	8400元/㎡	31-103(万/套)	
10	中核·锦城	汉阳大道中核世纪广场对面	8500元/㎡	147-227(万/套)	
11	万博玖珑湾	甲宝山路万博玖珑湾	11000元/㎡	133-143(万/套)	

图2-2　贝壳新房武汉市部分楼盘信息

本案例主要解决以下3个问题。

- 如何获取网页数据。
- 如何解析网页并提取房源信息。
- 如何组织数据并存储。

本案例涉及的知识点范围如图2-3所示。

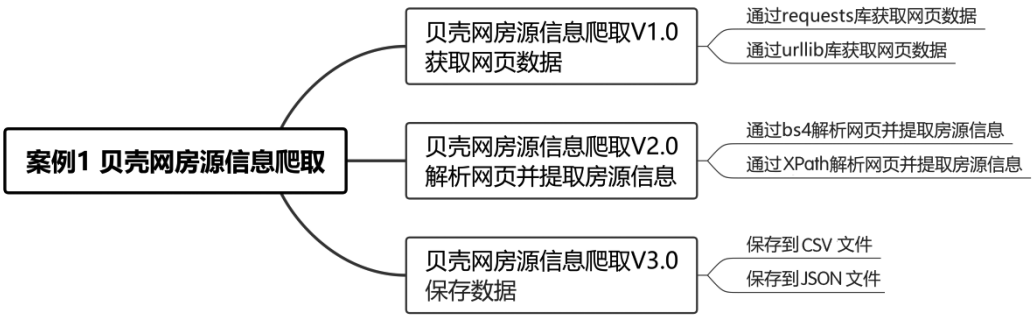

图2-3　贝壳网房源信息的爬取案例涉及知识点思维导图

2.1　网络爬虫的概念

网络爬虫是一种按照一定规则自动抓取互联网信息的程序或脚本。由于互联网数据的多样性和资源的有限性，根据用户需求定向抓取相关网页并进行分析，已成为目前主流的爬取策略。

网络爬虫程序通过模拟人类浏览网页、输入、单击、提交等操作，可以大幅提高工作效率，是目前非常热门的一个应用方向。网络爬虫程序结合数据分析技术，可以从海量杂乱无章的信息中按照既定规则进行过滤，从而得到有用的信息，进一步为商业决策提供支持。Python提供了丰富的用于编写网络爬虫程序的内置标准库和第三方扩展库，包括urllib、requests、scrapy、selenium等；也提供了较为丰富的网页数据提取库，如re、BeautifulSoup、lxml、PyQuery等。

利用Python相关技术可以爬取感兴趣的文本或图片，也可以爬取想看的视频等，凡是能通过浏览器访问的数据都可以通过网络爬虫获取。因此，网络爬虫的本质是模拟浏览器打开网页，并获取网页中我们真正想要的那部分数据(局部数据)。

2.2　网络爬虫的工作流程

网络爬虫通过模拟人工浏览网页的方式打开网页并提取所需数据，其工作的基本流程如图2-4所示。首先需要打开网页并获取网页的全部数据，这一步可以通过发送请求来实现。其次，获取响应内容。如果服务器正常响应，会收到response，即所请求的网页内容，包含HTML或JSON字符串或二进制数据(如视频、图片)等。再次，解析内容。如果是HTML源码，Python可以利用正则表达式或标准库、扩展库等提取用户需要的目标信息；如果是JSON数据，则可以转换成JSON对象进行解析；如果是二进制数据，则可以保存到文件或进一步处理。最后，保存内

容。以文本、图片、音频和视频等多种形式将解析得到的数据保存到本地，可以保存为文本文件或特定格式的文件，也可以保存到数据库(如MySQL、Redis、MongoDB等)。

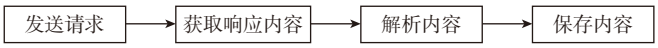

图2-4　网络爬虫的工作流程

1. 发送请求

浏览器向指定网址所在的服务器发送消息，该过程称为HTTP请求(Request)。请求可以包含额外的Header等信息，然后等待服务器响应。一个Request主要包括请求方式、请求URL、请求头和请求体信息，具体如图2-5所示。

图2-5　Request报文结构

(1) 请求方式：主要是GET(获取)和POST(传送)两种类型，还有HEAD、PUT、DELETE和PATCH等类型。

(2) 请求URL：一个网页文档、一张图片或一个视频等，都可以用URL唯一地确定。

(3) 请求头：包含请求时的头部信息，如User-Agent、Host和Cookies等。

(4) 请求体：请求时额外携带的数据，如表单提交时的表单数据等。

Python中有多种库可以用来处理HTTP请求，如原生的urllib库及第三方库requests。在Python2中，urllib和urllib2是两个相互独立的模块；而Python 3.x以上版本将urllib和urllib2合并为一个库。requests库是对urllib库的封装，其提供的方法与HTTP的方法一一对应，使用起来更加便捷，具体如表2-1所示。

表2-1　requests库方法与HTTP协议方法比较

requests库方法	HTTP协议方法	功能一致性
requests.get()	GET	一致
requests.head()	HEAD	一致
requests.post()	POST	一致
requests.put()	PUT	一致
requests.patch()	PATCH	一致
requests.delete()	DELETE	一致

2. 获取响应内容

服务器接收到浏览器发送的消息后，能够根据消息内容做相应处理，然后将消息回传给浏

览器，该过程称为HTTP响应。浏览器接收到服务器的响应信息后，会对信息进行相应处理，然后将信息显示出来。

(1) 响应状态：有多种响应状态，如200表示成功、301表示跳转、404表示找不到页面、502表示服务器错误等。

(2) 响应头部：包括内容类型、内容长度、服务器信息、设置Cookie等多个属性，如图2-6所示。其中，Content-Type属性用于告知客户端返回资源的文件类型和字符编码。

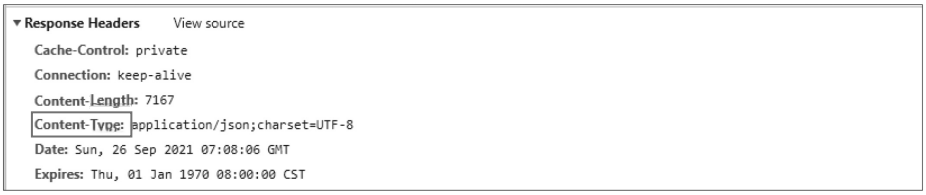

图2-6　Response响应头

(3) 响应体：响应体是响应中最主要的部分，包含了请求资源的内容，如网页HTML、JSON、XML等文本形式的数据，或者图片、音视频文件等二进制数据。如果网站数据是通过Ajax动态加载的，可能会导致请求获取的页面与浏览器实际显示的页面不一致。此时，需要通过Selenium、Splash、Ghost等库来模拟JavaScript渲染。

3. 解析网页内容

通过发送请求获取到的文本类型的数据，还需要进一步解析其中的内容，然后提取有用的局部数据，常见的文本格式有HTML、JSON、XML等。HTML格式的数据一般都是开放访问的，而大部分XML和JSON格式的数据则需要授权才能通过调用获取。

HTML(Hyper Text Markup Language，超文本标记语言)是一种标识性语言。它通过一系列标签将网络上的文档格式统一，使分散的Internet资源连接为一个逻辑整体。标准的HTML文件都具有一个基本的整体结构，即文件的开头与结尾标志，以及头部与主体两大部分。头部中包含的标记用于定义页面的标题、序言、说明等内容，主体则是网页中要显示的核心内容。图2-7上方为HTML源代码，下方为浏览器中的预览效果。常用的HTML数据解析器有re、html.parser、BeautifulSoup、lxml等。

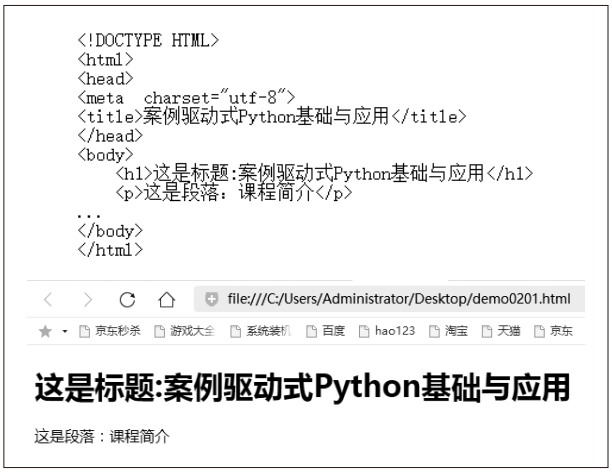

图2-7　HTML文档源代码与效果

JSON(JavaScript Object Notation，JavaScript对象表示法)是一种轻量级的数据交换格式，采用完全独立于编程语言的文本格式来存储和表示数据。其简洁、清晰的层次结构使JSON成为理想的数据交换语言。Python支持的所有数据类型(如字符串、数字、对象和数组等)都可以通过JSON来表示。其中，对象和数组是两种比较特殊且常用的类型。对象指在JavaScript 中使用花括号“{}”包裹起来的内容，其数据结构为{key1:value1, key2:value2, ...}的键值对结构。在面向对象的语言中，key为对象的属性，value为对应的值；键名可以使用整数和字符串来表示，值的类型可以是任意类型。数组指在JavaScript中使用方括号“[]”包裹起来的内容，其数据结构为["java", "javascript", "vb", ...] 的索引结构。对象和数组可以相互嵌套。JSON数据可以通过Python内置的json模块进行解析。

4. 清洗、组织数据并保存

爬虫抓取的数据经过解析后，可提取其中有价值的部分。这些数据可以根据人们的需求，通过编写相应的Python代码进行组织并保存，从而实现永久保存。如果数据量较小，可以直接使用文件保存；如果数据量较大，或者为了方便后续对数据进行查询和统计，则可以将数据存储到数据库中。

2.3　通过requests库获取网页数据

贝壳网房源信息的爬取V1.0：贝壳新房武汉市楼盘页面主要展示了该平台中武汉市挂牌出售的全部楼盘信息，那么如何获取该网页数据呢？

这可以通过Python第三方库requests来实现。

2.3.1　requests库简介

requests库是基于Python内置库urllib进行封装而形成的第三方库，以API的方式提供与HTTP方法相对应的功能。它将请求背后的复杂性抽象为简单的API，调用程序只需关注与服务器交互的数据，使用起来比urllib更加方便，可以节省大量工作，且完全满足HTTP的使用需求。

requests库是第三方库，因此使用前需要先安装，安装命令如下：

```
pip install requests
```

但更推荐采用国内镜像方式安装，例如采用清华大学的镜像资源方式安装，其安装命令如下：

```
pip install requests -i https://pypi.tuna.tsinghua.edu.cn/simple/
```

安装完成后，可以在Python IDLE交互式命令行下进行安装测试：

```
>>> import requests
# 获取"武昌首义学院"首页
>>> res = requests.get("http://www.wsyu.edu.cn/")
# 输出响应状态码
>>> print(res.status_code)
200
```

```
# 输出响应结果的前200个字符
>>> print(res.text[:200])
<!DOCTYPE html>
<html class="webplus-main"  >
<head>
  <meta charset="utf-8" />
  <meta name="renderer" content="webkit" />
  <meta http-equiv="X-UA-Compatible" content="IE=edge,chrome=1" />
```

2.3.2 requests库的常用方法

requests库是一个简洁且易于使用的处理HTTP请求的第三方库，提供了丰富的网页请求相关方法，如表2-2所示。

表2-2 requests库中网页请求的主要方法

requests库方法	描述
requests.request()	创建请求的通用方法
requests.get()	获取HTML网页的主要方法，对应于HTTP的GET
requests.post()	向HTML网页提交POST请求的方法，对应于HTTP的POST
requests.head()	获取HTML网页头信息的方法，对应于HTTP的HEAD
requests.put()	向HTML网页提交PUT请求的方法，对应于HTTP的PUT
requests.patch()	向HTML网页提交局部修改请求的方法，对应于HTTP的PATCH
requests.delete()	向HTML页面提交删除请求的方法，对应于HTTP的DELETE

1. requests.request()方法

requests.request()方法的语法格式如下：

```
requests.request(method,url,**kwargs)
```

参数如下。

- method：即请求方法，常见的有GET、POST，此外还有HEAD、PUT、PATCH、DELETE、OPTIONS。其中前6种即为HTTP协议所对应的请求方式，OPTIONS实际上是向服务器获取客户端与服务器之间能够交互的参数。请求方法不区分大小写。
- url：请求的URL地址。
- **kwargs：控制访问的参数，均为可选项，详情参见表2-3。在传递实参时，以关键字参数的形式传入，Python会自动将其解析为字典形式。

表2-3 **kwargs可选参数信息

可选参数名	描述
params	字典或字节序列，作为参数增加到URL中，一般用于GET请求，POST请求也可用(不常用)
data	字典、字节序列或文件对象，作为POST请求的参数
json	JSON格式的数据，作为POST请求的JSON参数
headers	字典，用于设置HTTP请求头
cookies	字典或CookieJar，表示请求中的Cookie
auth	元组，支持HTTP认证功能
files	字典类型，用于传输文件

(续表)

可选参数名	描述
timeout	设定超时时间，单位为秒
proxies	字典类型，用于设定访问代理服务器，可以增加登录认证
allow_redirects	True/False，默认为True，重定向开关
stream	True/False，默认为True，获取内容立即下载开关
verify	True/False，默认为True，认证SSL证书开关
cert	本地SSL证书

在表2-3中，headers参数可设置为浏览器信息User Agent，从而模拟浏览器发送请求。

User Agent(用户代理，简称UA)是一个特殊字符串头，用于服务器识别客户使用的操作系统及版本、CPU 类型、浏览器及版本、浏览器渲染引擎、浏览器语言、浏览器插件等信息。User Agent存放在请求的headers中，服务器就是通过查看headers中的UA来判断访问来源。在Python中，如果不手动设置User Agent，程序会使用内置默认UA。如果服务器校验User Agent，那么未设置User Agent的Python程序将无法正常访问网站。User Agent支持伪装，Python允许用户修改User Agent来模拟浏览器访问，即伪装成浏览器，从而隐藏爬虫程序的身份。

Windows 64平台的360安全浏览器的User Agent信息如下：

```
    Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML,like Gecko)
Chrome/78.0.3904.108 Safari/537.36.
```

对应的headers设置如下：

```
    headers = {'User-Agent':"Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36"}
```

通过requests.request()方法，伪装浏览器并以get方式访问“武昌首义学院”首页数据的代码如下：

```
    import requests
    header = {'User-Agent':"Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36"}
    url = "http://www.wsyu.edu.cn/"
    res = requests.request('get',url,headers=header)
    print(res.text)
```

2. requests库的get()方法

get()方法的语法格式如下：

```
    requests.get(url,params=None,**kwargs)
```

其中，url为请求的URL地址；params为URL中的额外参数，格式为字典或字节流，可选参数，默认为空；**kwargs的可选参数信息参见表2-3。get()是获取网页最常用的方法。在调用requests.get()函数后，返回的网页内容会保存为一个Response对象，例如：

```
    >>> import requests
    # 获取"武昌首义学院"首页
```

```
>>> res = requests.get("http://www.wsyu.edu.cn/")
>>> type(res)
<class 'requests.models.Response'>
# 返回Response对象
```

与浏览器的交互过程类似，requests.get()代表请求过程，返回的Response对象代表响应。返回的内容作为一个对象，便于操作。Response对象的主要属性如表2-4所示。

表2-4　Response对象的主要属性

属性	描述
status_code	HTTP请求的返回状态码，常见的状态码有200和404，200表示连接成功，404表示失败
text	HTTP响应内容的字符串形式，即URL对应的页面内容
encoding	HTTP响应内容的编码方式
apparent_encoding	从响应内容中分析出的编码方式(即网页的实际编码方式)
content	HTTP响应内容的二进制形式

status_code属性返回HTTP请求后的状态。在处理数据之前要先判断状态情况，如果请求未被响应，即意味着获取响应内容失败，需要进行异常处理并终止内容处理。text属性是请求的页面内容，以字符串形式展示。encoding属性非常重要，它给出了返回页面内容的编码方式，可以通过对encoding属性赋值来更改编码方式，以便进行中文字符的处理。apparent_encoding是从响应内容中分析出的编码方式，是备选编码方式。content属性是页面内容的二进制形式。例如：

```
>>> import requests
# 获取"武昌首义学院"首页
>>> res = requests.get("http://www.wsyu.edu.cn/")
>>> res.status_code       # 返回状态
200
# 中文字符显示为乱码(因内容过多，在此只给出了一部分数据)
>>> res.text
'<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
org/TR/xhtml1/DTD/xhtml1-transitional.dtd">\n<html xmlns="http://www.w3.org/1999/
xhtml">\n<head>\n  <meta http-equiv="Content-Type" content="text/html; charset=
utf-8" />\n  <title style="color:rgb(51,51,51); border-color:rgb(51,5);">æ\xad¦æ\
x98\x8cé¦\x96ä¹\x89å\xad¦é\x99¢â\x80\x94å\x8e\x9få\x8d\x8eä¸\xadç§\x9...'
# 默认的编码方式是ISO-8859-1，所以中文是乱码
>>> res.encoding
'ISO-8859-1'
# 更改编码方式为响应内容分析出的编码方式
# 此处apparent_encoding的值为utf-8
>>> res.encoding = res.apparent_encoding
# 更改完成，返回内容中的中文字符可以正常显示了
>>> res.text
'<!DOCTYPE html PUBLIC "-//W3C//DTD XHTML 1.0 Transitional//EN" "http://www.w3.
org/TR/xhtml1/DTD/xhtml1-transitional.dtd">\n<html xmlns="http://www.w3.org/1999/
xhtml">\n<head>\n  <meta http-equiv="Content-Type" content="text/html; charset= utf-
8" />\n  <title style="color:rgb(51,51,51); border-color:rgb(51,5);">武昌首义学院——
原华中科技大学武昌分校...'
```

除了属性，Response对象还提供了一些方法，如表2-5所示。

表2-5　Response对象的主要方法

方法	描述
json()	如果HTTP响应内容包含JSON格式数据，则该方法解析JSON数据
raise_for_status()	如果HTTP响应状态码不是200，则产生异常

json()方法能够在HTTP响应内容中解析存在的json数据，这将使解析HTTP更便利。raise_for_status()方法能在非成功响应后产生异常，即只要返回的请求状态码status_code不是200，该方法会产生一个异常，此时可以引入异常处理机制try...except。

requests库在处理HTTP请求时，可能会产生几种常用异常：当遇到网络问题时，如DNS查询失败、拒绝连接等，requests会抛出ConnectionError异常；当遇到无效的HTTP响应时，requests会抛出HTTPError异常；若请求URL超时，则抛出Timeout异常；若请求超过了设定的最大重定向次数，则会抛出TooManyRedirects异常。

获取一个网页的通用代码框架如下：

```
import requests
def getHTMLText(url):
    try:
        # 设置超时时长为30秒
        r = requests.get(url,timeout = 30)
        # 如果HTTP响应状态码不是200，则产生异常
        r.raise_for_status()
        # 设置编码方式
        r.encoding = r. apparent_encoding
        # 返回响应内容字符串
        return r.text
    except:
        return "产生异常"
url = "http://www.wsyu.edu.cn/"指定url
print(getHTMLText(url))#测试
```

3. requests库的post()方法

post()方法的语法格式如下：

```
requests.post(url,data=None,json=None,**kwargs)
```

requests.post()方法一般用于表单提交，向指定URL提交数据，可提交字符串、字典、文件等数据。其中，url为请求的URL地址；data为请求上传的数据，通常以字典形式传递；json为请求体中上传的JSON数据。post()方法的常用方式参见如下代码：

```
>>> import requests
>>> params = {'spam':1,'eggs':2,'bacon':0}
>>> res = requests.post("http://www.musi-cal.com/cgi-bin/query",data=params)
>>> res.text  # 输出信息过多，在此只输出部分信息
'<!DOCTYPE html>\r\n<html lang="ja" prefix="og:http://ogp.me/ns#">\r\n<head
prefix="og:http://ogp.me/ns# fb:http...'
>>> r = requests.post("http://httpbin.org/post",data = "helloWorld")
>>> r.text
```

```
    '{\n "args":{},\n "data":"helloWorld",\n "files":{},\n "form":{},\n "headers":{\n
"Accept":"*/*",\n "Accept-Encoding":"gzip,deflate",\n "Content-Length":"10",\n "Host":
"httpbin.org",\n  "User-Agent":"python-requests/2.26.0",\n  "X-Amzn-Trace-Id":"Root=1-
618cd757-1b398e34049f8917361daa82"\n},\n  "json":null,\n  "origin":  "21.181.60.150",\n
"url":"http://httpbin.org/post"\n}\n'
```

贝壳网房源信息的爬取V1.0

任务1：获取贝壳新房武汉市楼盘页面的房源数据。

解决方法(通过requests库获取网页内容)：

```
    def getHousesHtmlTextRequests(url):
        header = {'User-Agent':"Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36"}
        try:
            res = requests.get(url,headers=header)
            res.raise_for_status()
            res.encoding = res.apparent_encoding
            text = res.text
            return text
        except:
            return "获取房源数据失败"
    # 调用getHousesHtmlTextRequests()，通过requests库获取网页内容
    import requests
    url = 'https://wh.fang.ke.com/loupan/'
    print(getHousesHtmlTextRequests(url))
```

练一练

【练一练2-1代码填空】使用requests库中的get方法获取指定url的网页数据，请将代码补充完整。

```
    >>> import requests
    >>> url = 'https://www.qq.com/'
    >>> res = _____________________________
    >>> print(res.text)
```

【练一练2-2代码填空】使用requests库中的post方法获取网页数据，请将代码补充完整。

```
    >>> import requests
    >>> import json
    >>> url = 'https://fanyi.baidu.com/sug'
    >>> form_data = {'kw':'苹果',} #form_data为请求表单数据
    >>> response =_____________(_____________,data= _____________)
    >>> res = json.loads(response.text)
    >>> print(res)
```

2.4 通过urllib库获取网页数据

贝壳网房源信息的爬取V1.0：贝壳新房武汉市楼盘页面主要展示了该平台中武汉市挂牌出售的全部楼盘信息，那么如何获取该网页数据呢？

除通过requests库实现外，还可以使用Python内置的urllib库来实现。

实现网络爬虫，首先要爬取网页数据，即下载包含目标数据的网页。爬取网页需要爬虫向服务器发送一个HTTP请求，然后接收服务器返回的响应内容中的整个网页源代码。

利用Python完成该过程，既可以使用前面所介绍的第三方库requests，也可以使用Python的内置库urllib。使用这两个库爬取网页数据时，只需要关心请求的URL格式、传递的参数及请求头设置，而无须关心其底层实现细节。

2.4.1　urllib库简介

urllib库是Python内置的HTTP请求库，可视为处理URL的组件集合。urllib库包含以下四大模块。

(1) urllib.request模块：即请求模块，用于打开并读取URL。

(2) urllib.error模块：即异常处理模块，包含一些由urllib.request产生的异常，可以使用try进行捕捉处理。

(3) urllib.parse模块：即URL解析模块，包含一些解析URL的方法。

(4) urllib.robotparser模块：即robots.txt解析模块，用于解析robots.txt文本文件。它提供了一个单独的RobotFileParser类，通过该类提供了can_fetch()方法，用于测试爬虫是否可以下载一个页面。

2.4.2　urllib库的基本使用

urllib.request 模块提供了最基本的构造 HTTP 请求的方法，利用它可以模拟浏览器的请求发起过程。同时，该模块还支持处理授权验证(authenticaton)、重定向(redirections)、浏览器Cookies(cookies)及其他内容。

1. urlopen()函数

使用urllib.request.urlopen()函数可以很方便地打开一个网站并读取网页信息。urlopen()函数的语法格式如下：

```
urlopen(url[,data=None[,proxies]])
```

urlopen()返回一个HTTPResponse对象，然后可以像操作本地文件一样操作该对象，以获取远程数据。其中：参数url为目标资源在网站中的位置，可以是一个表示URL地址的字符串，也可以是一个Request对象；参数data用来指明向服务器发送的额外信息(默认为None，此时以GET方式提交数据，当用户设置data参数时，需要将数据提交方式改为POST)；参数proxies用于设置代理。urlopen()函数还有其他一些可选参数，具体可以参见Python帮助文档。

urlopen()函数返回的HTTPResponse对象提供了如下方法。

(1) read()、readline()、readlines()、fileno()、close()：这些方法的使用方式与文件对象完全一样。

(2) info()：返回一个http.client.HTTPMessage对象，表示远程服务器返回的头信息。

(3) getcode()：返回HTTP状态码。

(4) geturl()：返回请求的URL。

例如，在命令行输入如下语句：

```
>>> import urllib.request
>>> f = urllib.request.urlopen("https://www.baidu.com")
>>> type(f)
<class 'http.client.HTTPResponse'>
# 读取网页内容并解码
>>> html = f.read().decode('utf-8')
>>> print(html)
<html>
<head>
  <script>
      location.replace(location.href.replace("https://","http://"));
  </script>
</head>
<body>
   <noscript><meta http-equiv="refresh" content="0;url=http://www.baidu.com/"></
noscript>
</body>
</html>
```

由上述语句的输出结果可以看出，urlopen()函数的返回值类型是http.client模块的HTTPResponse类型对象。通过调用HTTPResponse对象的read()方法读取网页内容，且read()获取的是请求页内容的二进制形式，可以用decode()方法将网页信息进行解码。在此重点介绍了read()方法，其他方法如readline()、readlines()、fileno()、close()可参见Python帮助文档。

再在命令行中输入如下语句：

```
>>> import urllib.request
>>> f = urllib.request.urlopen("https://www.baidu.com")
>>> f.geturl()
https://www.baidu.com
>>> f.getcode()
200
>>> f.info()
Accept-Ranges:bytes
Cache-Control:no-cache
Content-Length:227
Content-Type:text/html
Date:Tue,05 Oct 2021 07:54:38 GMT
P3p:CP=" OTI DSP COR IVA OUR IND COM "
P3p:CP=" OTI DSP COR IVA OUR IND COM "
Pragma:no-cache
Server:BWS/1.1
Set-Cookie:BD_NOT_HTTPS=1; path=/; Max-Age=300
Set-Cookie:BIDUPSID=1DDFB66D1B5DE434565D428D1E1B78AA; expires=Thu,31-Dec-37
23:55:55 GMT; max-age=2147483647; path=/; domain=.baidu.com
Set-Cookie:PSTM=1633420478; expires=Thu,31-Dec-37 23:55:55 GMT; max-
age=2147483647; path=/; domain=.baidu.com
Set-Cookie:BAIDUID=1DDFB66D1B5DE4345EAC17D16421AA5B:FG=1; max-age=31536000;
expires=Wed,05-Oct-22 07:54:38 GMT; domain=.baidu.com; path=/; version=1; comment=bd
Strict-Transport-Security:max-age=0
```

```
Traceid:163342047802555540589717727333881334137
X-Frame-Options:sameorigin
X-Ua-Compatible:IE=Edge,chrome=1
Connection:close
```

由上述测试可知，HTTP状态码为200，表示服务器已成功处理了请求，即服务器提供了请求的网页。响应的头部信息与在浏览器中打开百度首页(https://www.baidu.com)后，按F12键，在Network选项卡中看到的响应标头信息一致。

2. 构造request对象

urlopen()的url参数，既可以是URL字符串，也可以是Request对象。当使用urlopen()函数发送请求时，如果希望执行更为复杂的操作(如增加HTTP报头)，则必须创建一个Request对象来作为urlopen()函数的参数。例如，在命令行中输入如下指令，同样可以访问百度首页的内容。

```
>>> import urllib.request
>>> request=urllib.request.Request("https://www.baidu.com")
>>> f = urllib.request.urlopen(request)
# 读取网页内容并解码
>>> html = f.read().decode('utf-8')
>>> print(html)
<html>
<head>
      <script>
      location.replace(location.href.replace("https://","http://"));
      </script>
</head>
<body>
       <noscript><meta http-equiv="refresh" content="0;url=http://www.baidu.
com/"></noscript>
</body>
</html>
```

3. 向服务器发送数据

在爬取网页时，可以通过HTTP请求向服务器传递数据，传递数据的方式分为GET和POST两种。两者的特点为：GET方式可以通过URL提交数据，即待提交的数据是URL的一部分。由于各种浏览器和服务器对URL的长度有限制，因此通过GET方式提交的数据内容长度也受限。例如，在命令行中输入如下指令测试GET方法，提交的数据data即为URL的一部分。

```
>>> import urllib.parse
>>> import urllib.request
>>> data = urllib.parse.urlencode({'word':'hello'})
>>> url='http://httpbin.org/get?%s' %data
>>> url
'http://httpbin.org/get?word=hello'
>>> res=urllib.request.urlopen(url)
>>> res.read()
```

```
    b'{\n "args":{\n "word":"hello"\n},\n "headers":{\n "Accept-Encoding":"identity",
\n "Host":"httpbin.org",\n "User-Agent":"Python-urllib/3.7",\n "X-Amzn-Trace-Id":
"Root=1-615c16e7-330778991b6aa41f674f37c9"\n},\n "origin":"219.140.64.130",\n "url":
"http://httpbin.org/get?word=hello"\n}\n'
```

POST方式将提交的数据放置在HTML HEADERS内，且对提交内容的长度无限制。例如，可以在命令行中输入如下指令来测试POST方式。

```
    >>> import urllib.parse
    >>> import urllib.request
    >>> data = bytes(urllib.parse.urlencode({'word':'hello'}),encoding='utf8')
    >>> res = urllib.request.urlopen('http://httpbin.org/post',data=data)#使用post方式
    >>> res.read()
    b'{\n "args":{},\n "data":"",\n "files":{},\n "form":{\n "word":"hello"\n},\n
"headers":{\n"Accept-Encoding":"identity",\n  "Content-Length":"10",\n
"Content-Type":"application/x-www-form-urlencoded",\n  "Host":"httpbin.org",\
n"User-Agent":"Python-urllib/3.7",\n"X-Amzn-Trace-Id":"Root=1-615c0cde-
07c1da2403a5de5a78a29053"\n},\n"json":null,\n"origin":"219.140.64.130",\n"url":
"http://httpbin.org/post"\n}\n'
```

对比上述GET和POST两种请求得到的响应数据发现，通过GET方式提交的数据存在于参数args字段中，而通过POST方式提交的数据存在于参数form(即表单)中。

4. 设置超时时间

通过request模块的urlopen方法向目标服务器发送请求时，如果长时间未获得响应，不应继续等待，而应立即终止该请求。否则，不仅会继续耗费本地客户端和对方服务器的资源，还会导致调用程序的用户在长时间内得不到响应，从而降低用户体验。

在实际项目中，调用urlopen方法时通常会加入timeout参数来指定超时时间，该参数的单位是秒，具体取值可根据实际项目需求确定，一般不宜过长。如果超过设定的时间后服务端仍未返回响应结果，就会抛出异常。下面代码示例演示了超时设置的效果：

```
import urllib.request
url = 'https://www.baida.com/'  #BAIDA.COM是一个国外的网站
# 设置超时时间为0.1秒
res = urllib.request.urlopen(url,timeout=0.1)
# 如果响应状态码为200，则成功返回
if res.getcode() == 200:
# 将结果以二进制方式读取后以utf-8解码输出
   print(res.read().decode('utf-8'))
```

运行上述程序，结果出现如下异常信息：

```
raise URLError(err)
urllib.error.URLError:<urlopen error timed out>
```

这说明在调用urlopen方法时传入了timeout=0.1的参数，即发出请求后经过0.1秒未得到响应，因此抛出异常。此时可以把timeout的参数值改成1或更大，就能看到正常返回的页面数据。因此，在实际项目中，可根据实际能接受的时间设置timeout参数值，以避免请求等待时间过长。

5. 设置headers属性来模拟浏览器发送请求

在前述代码示例中，都是直接通过urlopen方法发送HTTP请求来获取响应数据。然而，在访问某些网站(如豆瓣网)时，发送请求的同时需要携带浏览器访问时所带的HTTP头信息，从而把这个请求模拟成浏览器访问，否则可能无法获得预期的返回信息。通常这种做法也称为“使用User Agent隐藏身份”。

设置User Agent的方法有两种：一种是在创建Request对象时，填入headers参数(包含User Agent信息)，该headers参数要求为字典类型；另一种是在创建Request对象时不添加headers参数，待创建完成之后，使用add_header()方法添加headers。下面代码演示了创建Request对象时，填入headers参数(包含User Agent信息)的方式：

```
    import urllib.request
    url = 'https://www.douban.com/'  # 豆瓣网
    header = {'User-Agent':'Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36
(KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36'}
    # 创建Request对象时添加headers
    req = urllib.request.Request(url,headers = header)
    res = urllib.request.urlopen(req)
    if res.getcode() == 200:
       print(res.read().decode('utf-8'))
```

下面代码演示了在创建Request对象时不添加headers参数，待创建完成之后，使用add_header()方法添加headers的方式：

```
    import urllib.request
    url = 'https://www.douban.com/'  # 豆瓣网
    # 创建Request对象时不添加headers参数
    req = urllib.request.Request(url)
    req.add_header('User-Agent','Mozilla/5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36')
    # 通过add_header添加headers
    res = urllib.request.urlopen(req)
    if  res.getcode() == 200:
        print(res.read().decode('utf-8'))
```

6. 异常处理

urllib.error 模块为 urllib.request 所引发的异常定义了异常类，其基础异常类是 URLError。异常产生的原因可能是没有连接网络、服务器连接失败、找不到指定的服务器等。异常的主要类型有URLError 和 HTTPError两种。

URLError 是 OSError 的一个子类，用于处理程序在遇到问题时引发的此异常(或其派生异常)，其属性 reason 为引发异常的原因。

HTTPError 是 URLError 的一个子类，用于处理特定的 HTTP 错误(如认证请求时)，其属性 code为 HTTP 状态码，reason为引发异常的原因，headers为导致 HTTPError 的特定 HTTP 请求的响应头。

例如，若将百度的网址输入错误，可能导致无法连接服务器，从而引发[WinError 10060]异常。相应的异常处理代码如下：

```
import urllib.request
import urllib.error
request = urllib.request.Request('https://www.baidux.com/')
try:
    res = urllib.request.urlopen(request)
except urllib.error.URLError as e:
    print(e)
```

运行结果：

```
<urlopen error [WinError 10060] 由于连接方在一段时间后没有正确答复或连接的主机没有反应，连接尝试失败。>
```

如果能够成功连接服务器，但服务器无法处理请求内容，urlopen()函数会抛出HTTPError异常。其异常处理代码如下:

```
import urllib.request
import urllib.error
myURL1 = urllib.request.urlopen("https://www.runoob.com/")
print(myURL1.getcode())# 200
try:
    myURL2 = urllib.request.urlopen("https://www.runoob.com/no.html")
except urllib.error.HTTPError as e:
    if e.code == 404:
        print(404)
```

运行结果：

```
404
```

使用urllib.request时，若需同时捕获HTTPError和URLError，应注意由于HTTPError是URLError的子类，必须先捕获HTTPError，再捕获URLError，否则HTTPError将无法被捕获。

贝壳网房源信息的爬取V1.0

任务2：获取贝壳网房源板块的页面数据

解决方法(通过urllib库获取网页内容)：

```
def getHousesHtmlTextUrllib(url):
    header = {'User-Agent':"Mozilla/5.0 (Windows NT 10.0; WOW64) AppleWebKit/537.36 (KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36"}
    try:
        req = urllib.request.Request(url,headers = header)
        res = urllib.request.urlopen(req)
        if res.getcode() == 200:
            text = res.read().decode('utf-8')
            return text
    except:
        return "获取房源数据失败"
# 调用getHousesHtmlTextUrllib()，通过urllib库获取网页内容
import urllib.request
url = 'https://wh.fang.ke.com/loupan/' #目标页面url
print(getHousesHtmlTextUrllib(url)) #输出所获取的网页文本内容
```

练一练

【练一练2-3代码填空】使用urllib库中的urlopen方法获取网页数据，请将代码补充完整。

```
>>> import urllib.request
>>> url = 'https://www.qq.com'
>>> res = urllib.request._____________
>>> text = res._____________.decode('utf-8')
>>> print(text)
```

【练一练2-4代码填空】使用urllib并设置headers获取网页数据，请将代码补充完整。

```
>>> import urllib.request
>>> url = 'http://www.baidu.com'
>>> headers = {'_____________________________':'Mozilla/5.0 (Windows NT 10.0; WOW64)
AppleWebKit/537.36 (KHTML,like Gecko) Chrome/78.0.3904.108 Safari/537.36'}
# 创建headers信息
# 创建Request对象req时添加headers
>>> req = urllib.request.Request(url,headers = headers)
# 通过打开Request对象req获取网页内容
>>> res = urllib. _____________________
>>> text = res.read().decode('utf-8')
>>> print(text)
```

2.5　通过BeautifulSoup解析网页

贝壳网房源信息的爬取V2.0：贝壳新房武汉市楼盘页面主要展示了该平台中武汉市挂牌出售的全部楼盘信息。那么，在获取网页信息的基础上，如何进一步解析网页并提取房源信息？这可以使用BeautifulSoup来实现。

在贝壳网房源信息的爬取V1.0中，使用Python第三方库requests或内置库urllib可以将整个网页的内容全部爬取下来。但是，这些数据的信息往往非常庞大，不仅整体上给人混乱的感觉，而且大部分数据并不是用户所关心的。针对这种情况，需要对爬取的全局数据进行过滤和筛选，选取所关心的数据。为此，Python支持一些解析网页的技术，包括正则表达式、XPath、BeautifulSoup和JsonPath等。

BeautifulSoup库(也称为beautifulsoup4或bs4)，约定的引用方式为from bs4 import BeautifulSoup或 import bs4，其主要使用BeautifulSoup类的构造方法将HTML网页文档对应的标签树转换为BeautifulSoup对象。因此，一个BeautifulSoup对象对应一个HTML或XML文档的全部内容。下面介绍如何使用BeautifulSoup解析网页信息。

2.5.1　beautifulsoup4库简介

获取HTML网页并将其转换为字符串后，需要进一步解析HTML页面格式，提取有用信息，这时需要借助处理HTML或XML的函数库。

beautifulsoup4库(也称Beautiful Soup库或bs4库)用于解析、处理HTML和XML。其最大的优点就是能根据HTML和XML语法建立解析树，并提供一些简单的方法和Python函数，用于高效