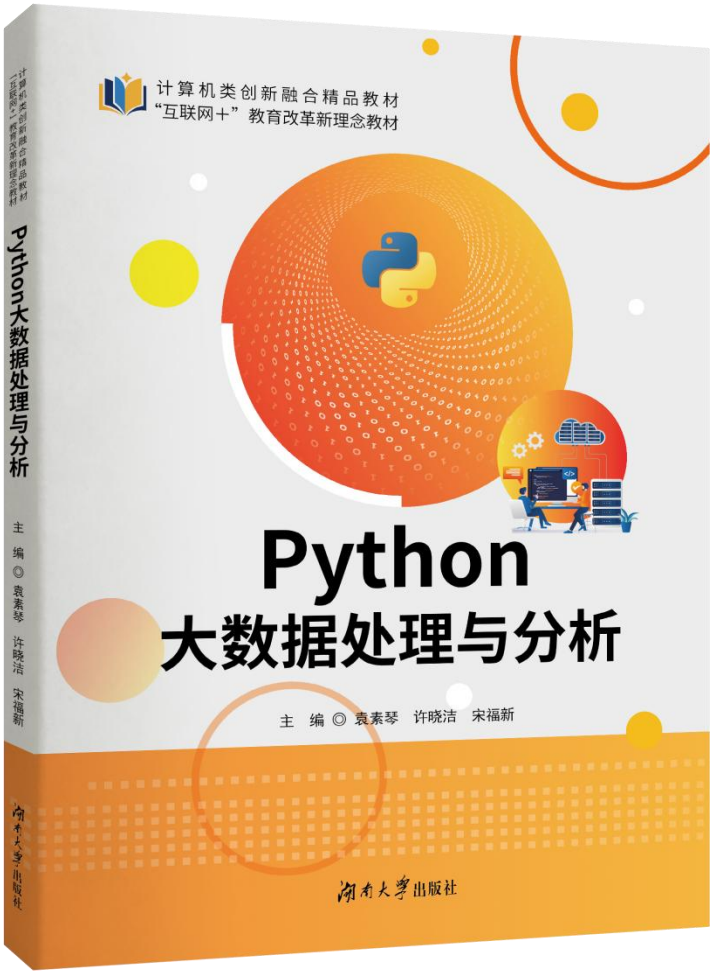
python 大数据处理与分析



类目：计算机类
书名：python 大数据处理与分析
主编：袁素琴 许晓洁 宋福新
出版社：湖南大学出版社
开本：大 16 开
书号：978-7-5667-4829-4
使用层次：通用
出版时间：2026 年 7 月
定价：69.80 元
印刷方式：双色
是否有资源：有

Python
大数据处理与分析

计算机类创新融合精品教材
“互联网+”教育改革新理念教材

Python大数据处理与分析

主　编◎袁素琴　许晓洁　宋福新

湖南大学出版社

责任编辑：杨琛晖
封面设计：旗语书装

ISBN 978-7-5667-1879-6
9 787566 718796 >
定价：69.80元

计算机类创新融合精品教材
“互联网+”教育改革新理念教材

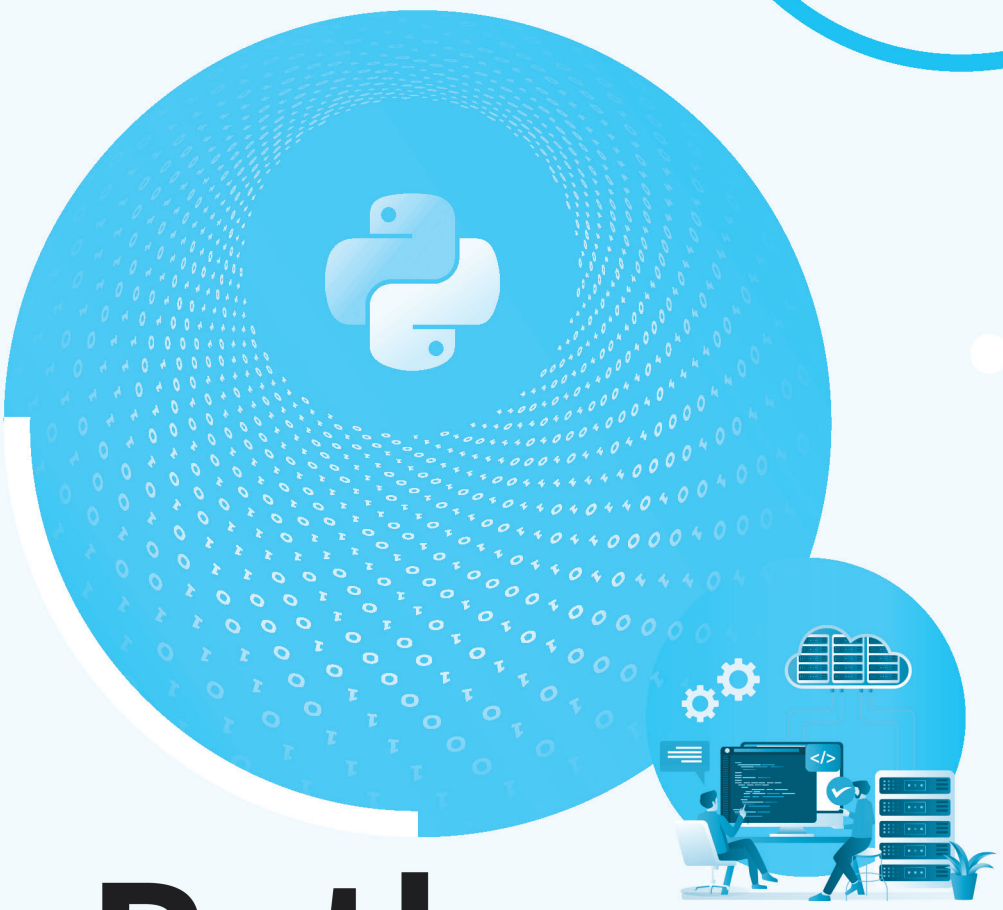


Python
大数据处理与分析

主　编◎袁素琴　许晓洁　宋福新

湖南大学出版社

计算机类创新融合精品教材
“互联网 +”教育改革新理念教材



Python 大数据处理与分析

主　编 ◎ 袁素琴　许晓洁　宋福新
副主编 ◎ 张　巍　张　旭　袁诗怡
熊　凯　赵玉明　张　蕾
赵翊凡　马　宁　罗文秀
王　凯

湖南大学出版社
·长 沙·

图书在版编目(CIP)数据

Python 大数据处理与分析 / 袁素琴，许晓洁，宋福新主编. -- 长沙：湖南大学出版社，2026.7. --ISBN 978-7-5667-4829-4

Ⅰ. TP312.8

中国国家版本馆 CIP 数据核字第 2026GU6786 号

Python 大数据处理与分析

PYTHON DASHUJU CHULI YU FENXI

主　　编：袁素琴　许晓洁　宋福新

责任编辑：杨琛晖

印　　装：河北龙大印务有限公司

开　　本：889 mm×1194 mm　1/16　**印　　张**：22　**字　　数**：534 千字

版　　次：2026 年 7 月第 1 版　**印　　次**：2026 年 7 月第 1 次印刷

书　　号：ISBN 978-7-5667-4829-4

定　　价：69.80 元

出 版 人：李文邦

出版发行：湖南大学出版社

社　　址：湖南·长沙·岳麓山　**邮　　编**：410082

电　　话：0731-88822559(营销部)，88821174(编辑室)，88821006(出版部)

传　　真：0731-88822264(总编室)

网　　址：http://press.hnu.edu.cn

电子邮箱：xiaoshulianwenhua@163.com

版权所有，盗版必究

图书凡有印装差错，请与营销部联系

PREFACE 前言

数据分析是用适当的统计分析方法对收集的大量数据进行分析，提取有用信息和形成结论，并对数据加以详细研究和概括总结的过程。这一过程也是质量管理体系的支持过程。在实际工作中，数据分析可帮助人们做出判断，以便采取适当的行动。数据分析是数学与计算机科学相结合的产物。利用 Python 简单易学且功能强大的编程能力，可以方便、快捷地进行大数据分析。

本书以合理的结构和经典的案例对 Python 数据分析的功能进行详细介绍，由浅入深、循序渐进，具有较高的实用价值。通过对本书的学习，读者可以掌握 Python 数据分析的基本知识和应用技巧。

一 本书特点

◇ 实例丰富

本书的实例不管是数量还是种类，都非常丰富。从数量上来说，本书结合大量的 Python 数据分析实例，详细讲解 Python 数据分析的原理与应用知识要点，让读者在学习的过程中潜移默化地掌握 Python 数据分析的应用技巧。

◇ 突出提升技能

本书从全面提升 Python 数据分析实际应用能力的角度出发，结合大量实例来讲解如何使用 Python 进行数据分析，使读者了解 Python 数据分析的基本原理并能够独立完成各种 Python 数据分析应用操作。

本书中有很多实例本身就是 Python 数据分析项目案例，经过编者的精心提炼和改编，不但保证读者能够学好知识点，而且能够帮助读者掌握实际的操作技能，同时培养读者对 Python 数据分析的实践操作能力。

◇ 案例化教学，实操性强

本书的编者都是在高校从事 Python 数据分析教学研究多年的一线人员，具有丰富的教学实践经验与教材编写经验，多年的教学工作使他们能够准确地把握学生的心理与实际需求，本书是编者总结多年的开发经验及教学的心得体会，历时多年的精心准备，力求全面、细致地展现 Python 数据分析应用领域的各种功能和使用方法。

全书把 Python 数据分析理论知识分解并融入一个个实践操作的训练项目中，增强了本书的实用性。

二 本书的基本内容

全书分为 6 个项目，具体内容为 Python 数据分析基础、NumPy 数据探索和预处理、Pandas 数据处理与分析、Matplotlib 数据可视化、SciPy 统计推断与探索性分析、Scikit-learn 库模型与数据预处理。

编　者

2026 年 2 月

CONTENTS 目录

项目一 Python 数据分析基础 …… 1

任务一 认识 Python 数据分析工具 …… 2
任务二 Python 开发环境 …… 14
任务三 Python 库管理 …… 29
任务四 Python 基础知识 …… 38
任务五 Python 流程控制 …… 55
【项目实战】…… 70
【课后习题】…… 72

项目二 NumPy 数据探索和预处理 …… 75

任务一 NumPy 数组运算 …… 76
任务二 NumPy 数据操作 …… 89
任务三 数据加载、统计与预处理 …… 103
【项目实战】…… 112
【课后习题】…… 117

项目三 Pandas 数据处理与分析 …… 119

任务一 Pandas 数据采集 …… 120
任务二 Pandas 数据预处理与探索 …… 138
任务三 Pandas 描述性统计分析 …… 166
【项目实战】…… 173
【课后习题】…… 187

项目四 Matplotlib 数据可视化 …… 189
任务一 认识图表窗口 …… 190
任务二 图表通用属性设置 …… 201
任务三 核心绘图函数 …… 218
【项目实战】 …… 236
【课后习题】 …… 241

项目五 SciPy 统计推断与探索性分析 …… 243
任务一 相关性分析 …… 244
任务二 假设检验 …… 250
任务三 t 检验 …… 262
任务四 方差分析 …… 266
【项目实战】 …… 269
【课后习题】 …… 276

项目六 Scikit-learn 库模型与数据预处理 …… 279
任务一 Scikit-learn 库数据集 …… 280
任务二 数据预处理 …… 286
任务三 分类算法 …… 309
任务四 回归算法 …… 317
任务五 降维算法 …… 323
任务六 聚类分析 …… 330
【项目实战】 …… 333
【课后习题】 …… 343

参考文献 …… 345

项目一

Python数据分析基础

★ ★ ★

思政目标

- 培养严谨的科学态度和逻辑思维能力，增强运用数据解决实际问题的责任意识。
- 树立数据伦理意识和规范意识，理解技术应用的社会价值，为未来数据驱动的创新奠定基础。

技能目标

- 掌握 Python 的安装、交互式运行，并能熟练配置和使用 IDLE Shell、Jupyter Notebook、PyCharm 等主流开发环境。
- 理解 Python 模块体系，掌握第三方库的安装、加载和导入方法。
- 理解并应用 Python 核心数据类型、运算符、输入/输出、数据转换和内置函数，并能自定义函数处理基本业务逻辑。
- 熟练运用顺序、选择(if)和循环(for/while)结构，以及流程控制语句(break/continue)，完成程序逻辑的设计与调试。

项目导读

Python 凭借强大的生态系统，已成为数据分析领域的首选语言。本项目将从环境配置、基础语法、流程控制和库管理等方面，全面提升学员的 Python 编程能力，并通过一系列实例(如零售门店营业额分析、学生成绩判断等)深化理解，为后续进行数据清洗、统计分析、数据可视化等高级数据分析任务打下坚实且高效的基础。

任务一 认识 Python 数据分析工具

任务引入

小李是一名对技术充满激情的大一新生。他注意到，许多公司都在招聘 Python 数据分析师，并且开出了优厚薪资的待遇。为了提升自己的就业竞争力，小李决定通过完成一个实际的数据分析项目来学习 Python。他深知“工欲善其事，必先利其器”，遂将熟悉 Python 数据分析工具作为首要任务。然欲善用 Python，必先识其本源。如何便捷安装 Python？Python 内置函数与库函数又该如何理解？

知识准备

Python 是一门简单易学且功能强大的编程语言，它拥有高效的高级数据结构，并且能够简单又高效地面向对象编程。

一、Python 发展与起源



Python 发展与起源

Python 由荷兰国家数学和计算机科学研究中心的 Guido van Rossum(吉多·范罗苏姆)于 20 世纪 90 年代初设计。作为一门叫作 ABC 语言的替代品，Python 既提供了高效的高级数据结构，还能简单、有效地面向对象编程，广泛应用于系统管理任务的处理和 Web 编程。Python 现今已经成为最受欢迎的程序设计语言之一。

1995 年，Guido van Rossum 在弗吉尼亚州的国家创新研究公司(CNRI)继续他在 Python 上的工作，发布了该软件的多个版本。

2000 年 5 月，Guido van Rossum 和 Python 核心开发团队转到 BeOpen.com 并组建了 BeOpen PythonLabs 团队。同年 10 月，BeOpen PythonLabs 团队转到 Digital Creations(现为 Zope Corporation)。

2001 年，Python 软件基金会(PSF)成立，这是一个专为拥有 Python 相关知识产权而创建的非营利组织。

在 2000 年 10 月 16 日，Python 2 发布，该系列的稳定版本是 Python 2.7。2004 年以后，Python 的使用率呈线性增长。

2008 年 12 月 3 日，Python 3 发布，该版本不完全兼容 Python 2。2011 年 1 月，Python 3 被 TIOBE 编程语言排行榜评为 2010 年度语言。

2021 年 10 月 4 日，Python 发布 3.10 版本。2023 年 10 月，Python 发布 Python 3.12.0，2023 年 12 月，Python 发布 Python 3.12.1，2024 年 2 月，Python 发布 Python 3.12.2。

2024 年 10 月，Python 发布 Python 3.13.0，Python 3.13.0 是当时 Python 编程语言的最新稳定发布版，包含多项针对语言、实现和标准库的改变。最大的变化包括一个新的交互式解释器，以及支持对于在自由线程模式(PEP 703)下运行和即时编译器(PEP 744)。

2025 年 10 月，Python 发布 Python 3.14.0，该版本主要新增了延迟评估注解(PEP 649)及模板字符串等特性。所有版本均享有 5 年的标准维护支持，持续推动语言向高性能与现代化演进。

二、Python 数据分析的基本流程

Python 数据分析的基本流程通常包括以下几个步骤：数据收集、数据清洗与预处理、数据探索性分析(EDA)、数据建模与分析、模型评估与优化、结果解释与呈现以及最后的决策支持与实施。每个步骤都可以通过 Python 中的不同工具和库来完成，从而确保数据分析的效率和质量。

1. 数据收集

数据收集是数据分析过程的第一步。Python 提供了多种方法来收集数据，数据的来源可以是本地文件、数据库、API 或者网络爬虫。常见的数据收集方式如下。

- 本地文件：通过 Python 的内置函数读取本地 CSV、Excel 或 JSON 文件。常用的有 read_csv、read_excel、openpyxl(处理 Excel 文件)。
- 数据库：通过 Python 的数据库连接库(如 sqlite3、SQLAlchemy 或 PyMySQL)连接到关系型数据库，提取数据。
- API 数据：使用 Python 的 requests 库来调用 RESTful API，获取从服务器端返回的数据。
- 网络爬虫：使用 Python 的 BeautifulSoup、Scrapy 或 Selenium 等库，抓取网页上的数据，特别适用于需要从网页提取信息的情况。

数据收集的关键是确保获取的数据是高质量和具有代表性的，能满足分析目标的需求。

2. 数据清洗与预处理

数据清洗是 Python 数据分析中至关重要的一步，因为真实世界中的数据往往包含许多噪声、缺失值和不一致性。常见的数据清洗步骤如下。

- 去除重复值：通过 Pandas 中的 drop_duplicates()方法删除重复的数据行。
- 处理缺失值：在数据中，缺失值是常见问题。可以通过 Pandas 的 fillna()方法对缺失值进行填充(例如使用均值、中位数或其他数据填充)，或者使用 dropna()删除含有缺失值的行或列。
- 数据类型转换：将数据转换为适合分析的格式。

- 处理异常值：异常值可能会影响分析结果。通过可视化或统计方法识别异常值，并决定是剔除还是替换。
- 数据标准化与归一化：有时数据的范围不同，这时可以使用 Scikit-learn 库中的 StandardScaler 或 MinMaxScaler 对数据进行标准化或归一化。

3. 数据探索性分析(EDA)

数据探索性分析(EDA)是 Python 数据分析流程中的一个重要步骤，旨在帮助分析师理解数据的结构、分布、趋势以及变量之间的关系。Python 提供了强大的数据可视化和分析工具。

- 基本统计分析：使用 Pandas 计算数据的描述性统计量(均值、标准差、最大值、最小值等)。
- 数据可视化：使用 Matplotlib 库对数据进行可视化。通过直方图、散点图、箱线图等方式查看数据的分布、趋势以及变量之间的关系。
- 相关性分析：通过计算皮尔逊(Pearson)相关系数来分析不同变量之间的线性关系。Pandas 提供了 corr()方法来计算变量间的相关性。

通过 EDA 可以发现数据中的潜在模式、异常点以及各特征之间的关系，为后续的建模和分析提供有价值的见解。

4. 数据建模与分析

数据建模是 Python 数据分析过程中至关重要的一步，主要目的是利用数据建立数学模型，并进行预测或分类分析。Python 提供了丰富的机器学习库，常用的包括回归分析、分类分析、聚类分析和降维分析。

- 回归分析：用于预测一个连续的变量，例如使用 Scikit-learn 的线性回归模型。
- 分类分析：用于将数据划分为不同的类别，例如使用逻辑回归、决策树、随机森林等分类模型。
- 聚类分析：用于将数据分成不同的组，例如使用 K-means 聚类。
- 降维分析：当数据维度很高时，可以使用主成分分析(PCA)来降维，便于后续分析。

5. 模型评估与优化

在建模阶段完成后，必须对模型进行评估，确保它能够有效地进行预测或分类。Python 提供了多种评估指标，通常取决于问题类型。

- 回归问题：使用均方误差(MSE)、均方根误差(RMSE)等指标评估模型表现。
- 分类问题：使用准确率、召回率、F1 分数、混淆矩阵等评估分类模型。
- 交叉验证：使用交叉验证技术来评估模型的泛化能力。

6. 结果解释与呈现

在模型评估完成后，分析师需要将分析结果传达给相关方。Python 提供了多种工具来呈现分析结果。

- 数据可视化：通过 Matplotlib 等可视化库展示数据的趋势、关系和分析结果，可以使用折线图、条形图、热力图等方式展示结果。

• 报告生成：可以使用 Jupyter Notebook 生成交互式报告，或者使用 Matplotlib 和 Seaborn 生成图表并嵌入报告中。

7. 决策支持与实施

数据分析的最终目的是支持决策。分析师需要通过洞察数据为企业或组织提供决策依据。例如，根据数据分析的结果，提供改善业务运营、调整市场策略、优化产品设计等方面的建议。

三、下载和安装 Python



下载和安装 Python

Python 是一种功能强大的解释型脚本语言，广泛应用于各种开发场景。要让代码顺利运行，首要步骤是在计算机上安装 Python 解释器，才可以解析并执行编写的脚本。

1. Python 下载

打开 Python 官网下载界面 https://www.python.org/downloads/，如图 1-1 所示。在列表框中显示不同版本的 Python。

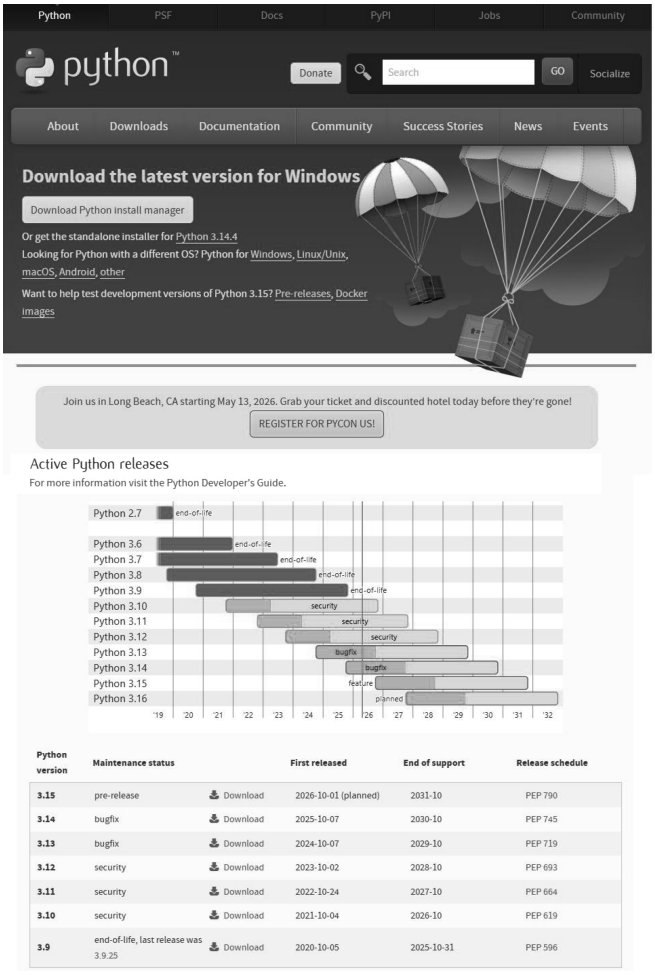


图 1-1 Python 官网下载界面

以 Python 3.14.0 版为例，直接单击 Python 官网下载界面上的“Download”按钮，下载

Python 3.14.0 的安装程序 python-3.14.0-amd64.exe(64 位的完整离线安装包)。

提示

每个版本的 Python 根据不同的计算机安装系统分为不同的安装软件。计算机安装系统包括 Windows、Linux/UNIX、macOS。一般大学生都使用 Windows 系统，因此，这里只介绍在 Windows 环境下下载 Python 3.14.0 软件及其安装过程。本书中介绍的程序也是在该系统下演示的。

2. 软件安装

(1)双击安装文件 python-3.14.0-amd64.exe，弹出安装界面“Install Python 3.14.0(64-bit)”，下面介绍该界面中的选项。

- Install Now：默认安装且默认安装路径不能更改(一般默认安装在 C 盘)。
- Customize installation：自定义安装。
- Use admin privileges when installing py.exe：默认勾选该复选框，为所有用户安装启动器。
- Add python.exe to PATH：勾选该复选框，将 Python 自动加到环境变量中，默认未勾选该复选框。

在该界面中勾选“Add python.exe to PATH”复选框，如图 1-2 所示。这样可以将 Python 命令工具所在目录添加到系统环境变量中，以后开发程序或运行 Python 命令会非常方便。



图 1-2　安装界面

提示

勾选“Add python.exe to PATH”复选框这步非常重要，若不勾选该复选框，则在软件安装完成而进行检查时，命令行提示符显示“‘python’不是内部或外部命令，也不是可运行的程序或批处理文件”，如图 1-3 所示。若要解决这个问题，则需要手动在计算机的环境变量中添加 Python 安装路径。

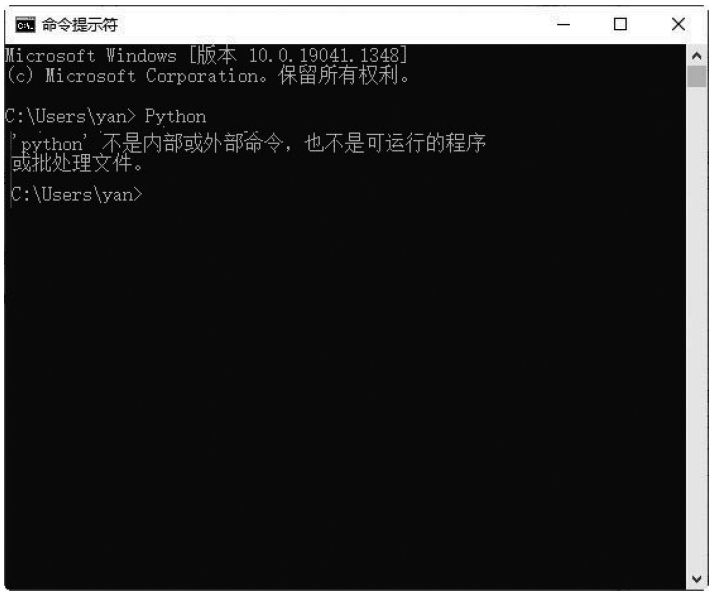


图 1-3　安装错误信息

(2)单击“Customize installation”(自定义安装)按钮，弹出选项设置界面“Optional Features”，保持默认参数设置，如图 1-4 所示。

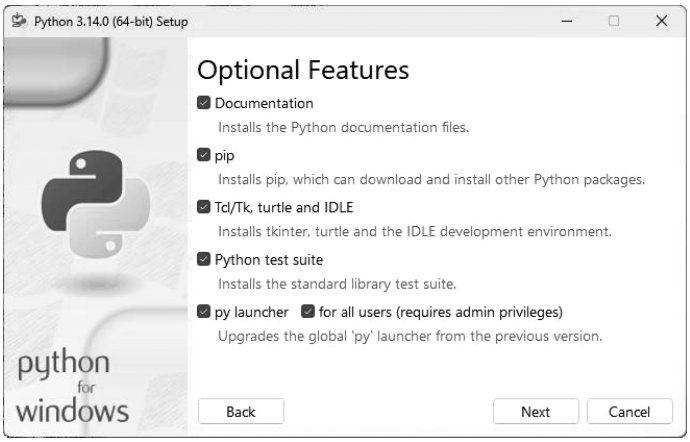


图 1-4　选项设置界面

- Documentation：勾选该复选框，安装 Python 帮助文档。
- pip：勾选该复选框，安装下载 Python 的工具 pip。pip 是现代通用的 Python 包管理工具，英文全称是 python install packages。
- Td/Tk，turtle and IDLE：安装标准库测试套件，包括 tkinter 和 IDLE。
- py launcher：安装 Python 的发射器。
- for all users(requires admin privileges)：适用于所有用户。

(3)单击“Next”(下一步)按钮，进入下一个高级设置界面“Advanced Options”，在“Customize install location”文本框中更改安装地址(不建议安装在 C 盘)，其余保持默认设置，设置完毕后如图 1-5 所示。

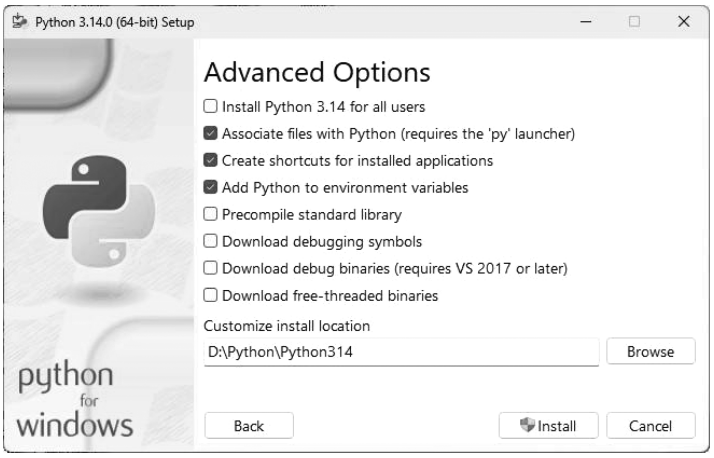


图 1-5　高级设置界面

(4)确定好安装路径后，单击“Install”(安装)按钮，此时会显示安装进度，如图 1-6 所示。由于系统需要复制大量文件，所以需要等待几分钟。在安装过程中，可以随时单击“Cancel”按钮来终止安装过程。

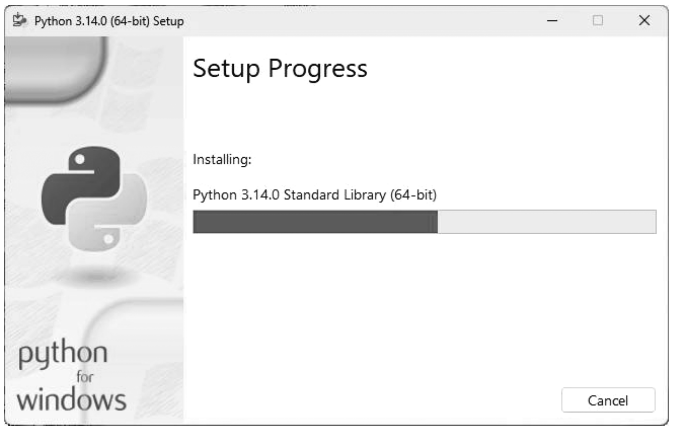


图 1-6　安装进度

(5)安装结束后，会出现“Setup was successful”(安装成功)界面，如图 1-7 所示。单击“Close”(关闭)按钮，即可完成 Python 3.14.0 的安装工作。

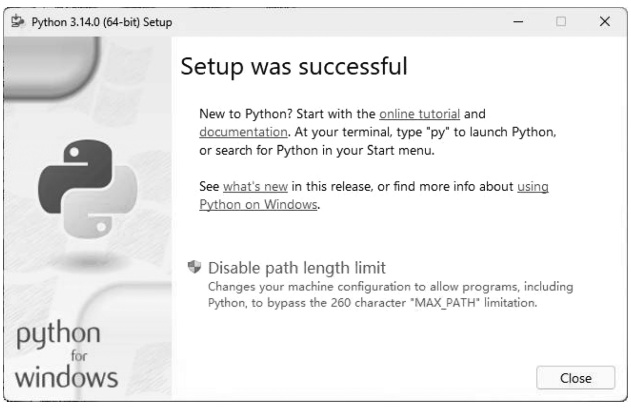


图 1-7　安装成功界面

3. 安装检查

Python 安装结束后，需要检查安装是否成功。在计算机的“开始”栏中输入“cmd”，打开“命令提示符”界面，输入“python”，按 Enter 键，出现如图 1-8 所示的运行结果，表示 Python 安装成功。

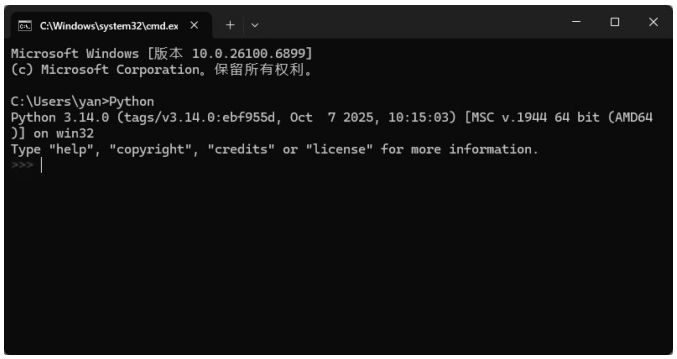


图 1-8 安装检查

实例 1.1 ——Python 的交互式运行环境

安装好 Python 后，接下来通过一个简单的加减法运算实例，帮助读者熟悉 Python 的交互式运行环境。

【操作步骤】

(1)按键盘上的 windows+R 组合键，打开“运行”对话框。

(2)在“打开”文本框中输入“cmd”，单击“确定”按钮或按 Enter 键，打开命令提示符窗口，如图 1-9 所示。

(3)在命令提示符后输入“python”命令，按 Enter 键，显示 Python 的版本信息，同时启动交互式运行环境，并显示命令提示符>>>，如图 1-10 所示。

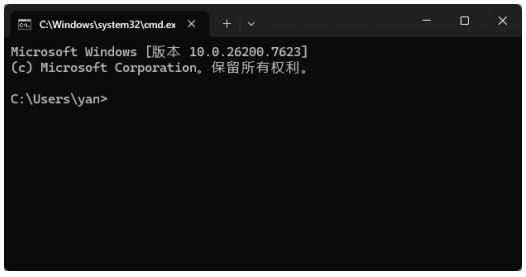


图 1-9 命令提示符窗口

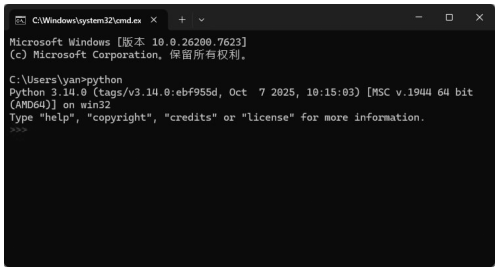


图 1-10 显示 Python 的版本信息

(4)在命令提示符>>>后输入下面的代码。

```
>>>10+1-2
```

(5)按 Enter 键执行命令，即可在命令行窗口中立即看到执行结果，并输出一个新的命令提示符>>>，如图 1-11 所示。

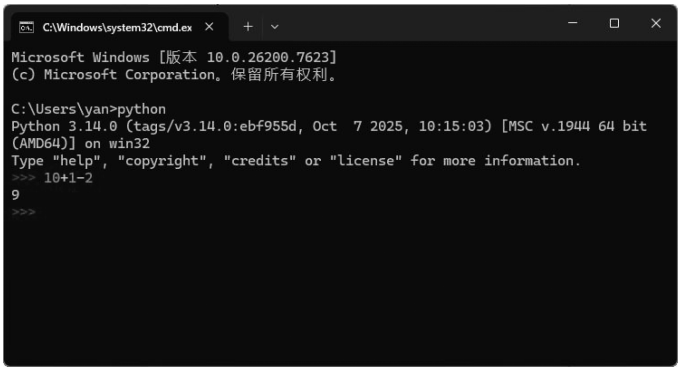


图 1-11　程序运行结果

(6)按下 Ctrl+Z 快捷键，或者输入 exit()命令，然后按 Enter 键，即可退出交互式运行环境，回到 Windows 命令行程序，如图 1-12 所示。

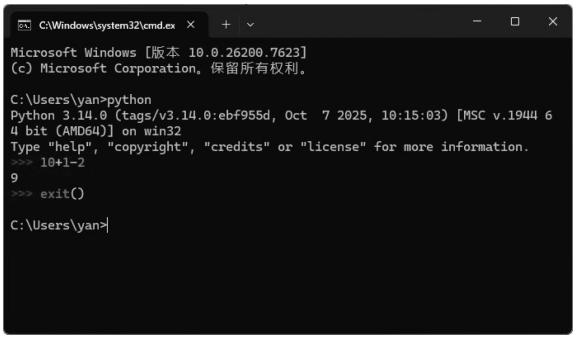


图 1-12　退出交互式运行环境

(7)在根目录后输入下面的代码。

C:\Users\yan>10+1-2

Windows 命令行程序不识别上面的代码，会显示报错，如图 1-13 所示。

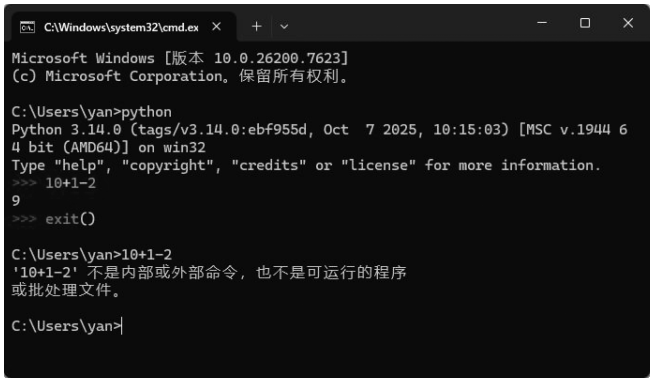


图 1-13　Windows 命令行程序报错

四、Python 语法规范

Python 作为一种高级编程语言，其语法规范对于编写清晰、易于维护的代码至关重要，遵循这些规范可以帮助我们写出更加专业和易于理解的 Python 代码。

1. 行宽

每行代码尽量不超过 79 个字符，以便于在控制台查看代码，增强代码可读性。

2. 空行

空行并不是 Python 语法的一部分，其作用主要是分隔两段不同功能或含义的代码，便于代码的维护或重构。通常，函数之间或类的方法之间用空行分隔，表示一段新的代码的开始。类和函数入口之间也用一行空行分隔，以突出函数入口的开始。其他使用空行的情况如下。

- 模块级的函数定义、类定义之间，通常应留出两行空行，这有助于区分全局层面的不同功能模块。
- 类内部的方法(_ _ init _ _ 方法与 hello 方法)定义之间，通常留出一行空行。
- 在函数内部可以使用空行来分隔逻辑相关的代码段(如数据预处理、核心计算与结果返回之间)，使函数逻辑更有层次感。
- 源文件末尾应保留一行空行，这是符合 POSIX 标准和 Python 解释器处理习惯的通用准则。

例如：

```
class A:

def _ _ init _ _ (self):
pass

def hello(self):
pass



def main():
pass
```

3. 空格符号

(1)在二元运算符两边各空一格[=，－，+=，==，>，in，is not，and]。例如：

```
# 规范
i = i + 1
x = x * 2 - 1
# 不规范
i=i+1
x = x*2-1
```

(2)在函数的参数列表中，逗号、#和冒号之后要有空格。例如：

```
# 规范
print("输出", {a: 1, b: 2})
#不规范
print("输出",{a:1,b:2})
```

(3)函数的参数列表中，默认值等号两边不要添加空格。例如：

```
# 规范
print('运算', c1, c2, sep='\n')
# 不规范
print('运算', c1, c2, sep = '\n')
```

(4)左括号之后，右括号之前不要加多余的空格。例如：

```
# 规范
spam(ham[1], {eggs: 2})
# 不规范
spam( ham[1], { eggs : 2 } )
```

(5)字典对象的左括号之前不要加多余的空格。例如：

```
# 规范
dict['key'] = list[index]
# 不规范
dict ['key'] = list [index]
```

(6)不使用额外的空格。例如：

```
# 规范
x = 1
y = 2
# 不规范
x             = 1
y             = 2
```

(7)如果注释位于代码旁边，建议与代码之间空两格。例如：

```
# 规范
x = 5  # 这是一个注释
# 不规范
x = 5# 这是一个注释
```

4. 编码方式声明

使用 UTF-8 编码方式声明的 Python 程序头文件：

```
# -*- coding: utf-8 -*-
```

因为 Python 的源代码文件默认使用 ASCII 编码，如果源代码中包含非 ASCII 字符(如中文、日文等)，就需要指定正确的编码方式以确保解释器能够正确地读取和处理这些字符。

5. 解释器路径声明

在 Unix/Linux/macOS 系统中，通常在第一行添加解释器路径声明：

```
#! /usr/bin/env python3
```

这行代码告知操作系统应使用哪个 Python 解释器来执行该脚本，使得脚本可以直接通过 ./script.py 的方式运行。在 Windows 系统中，这一行会被忽略，不起作用。当通过 py 命令执行脚本时，该启动器会特意识别并解析解释器路径声明，以便在安装了多个 Python 版本的环境中选择正确的解释器来运行脚本。为了代码的可移植性，编写需要跨平台运行的脚本时，加上解释器路径声明是一个好习惯。

6. 模块导入

在程序头部，使用 import 关键字来声明脚本所依赖的模块。例如：

```
import os
import sys
import math
```

声明了对 os、sys 和 math 模块的依赖，表示脚本文件中可以使用这些模块提供的功能。

7. 全局配置与常量定义

在 Python 模块中通常在导入语句(import)之后定义全局配置或常量，这些配置为整个脚本提供统一的参数控制，便于后续维护和逻辑调整。根据 PEP 8 规范，全局常量应使用全大写字母并以下划线分隔单词，以便与普通变量区分。

```
DEBUG = True      # 设置为 True 时开启调试模式，通常用于控制日志输出或错误捕获深度。
MAX_ITEMS = 100      # 全局逻辑常量。用于限制程序处理的最大条目数(本例设为100)。
```

任务二 Python 开发环境

任务引入

小李经过调查发现，虽然 Python 集成开发环境有很多，但是常用的也就几种。经过综合考虑，小李决定使用 PyCharm 编辑器。PyCharm 作为一款针对初学者使用的 Python 编辑器，配置简单、功能强大，使用起来省时省心。如果能熟练使用 PyCharm，一定能为其数据分析项目的开发带来便利。那么，小李该如何下载安装 PyCharm 呢？又该如何配置才能让 PyCharm 更好地为他服务？PyCharm 编辑器怎么进行 Python 编程？

知识准备

安装完 Python 后，在安装路径里的目录下会有一个 python.exe 文件，该文件是 Python 解释器，其作用是把 Python 代码转换为字节码，并进一步执行这些指令，确保程序能够正常运行。在使用 Python 进行编程时，可以选择不同的开发工具来提升效率，例如功能强大的 PyCharm，它适合大型项目开发；或者 Jupyter Notebook，它以交互式环境著称，非常适合数据分析和代码测试。无论是哪种工具，都能更高效地编写和调试 Python 代码。

一、IDLE Shell

Python 默认的集成开发环境 IDLE Shell 是一个功能完备的交互式编程客户端，在安装 Python 时会自动安装。在“开始”菜单中单击 Python 3.14 文件夹中的 IDLE(Python 3.14 64-bit)项(如图 1-14)，即可启动 Python 3.14 64-bit，打开“IDLE Shell 3.14.0”窗口。

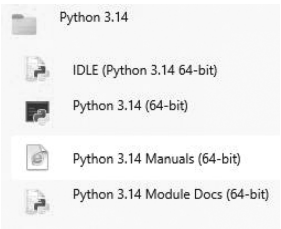


图 1-14 选择启动项

从图 1-15 可以看出，IDLE Shell 3.14.0 的工作界面由标题栏、菜单栏、工作区、状态栏等组成。

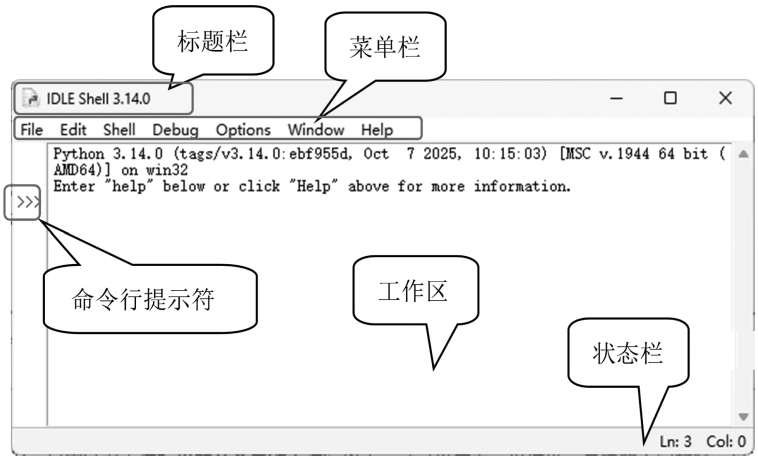


图 1-15 IDLE Shell 3.14.0 工作界面

提示

IDLE 是 Python 自带的集成开发环境，在上面章节安装步骤中勾选了 IDLE，安装后在电脑开始菜单中生成 IDLE 快捷方式。使用 IDLE 工具进行 Python 开发比较适合运行和调试一些简单的小程序，不适合做项目开发。

在 IDLE Shell 3.14.0 工作界面中可以进行各种计算操作，也可以使用命令打开各种工具，还可以查看各种命令的帮助说明等。其中，“>>>”为命令提示符，表示 Python 处于准备就绪状态。如在提示符后输入一条命令或一段程序后按 Enter 键，Python 将给出相应的结果，然后再次显示一个运算提示符“>>>”，如图 1-16 所示。

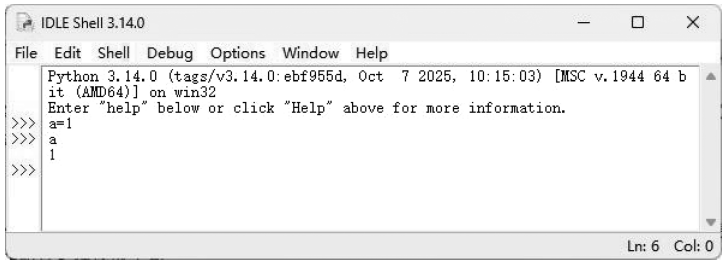


图 1-16 命令行窗口

二、Jupyter Notebook

Jupyter Notebook 是基于网页的、用于交互计算的应用程序，可被应用于全过程计算、开发、文档编写、代码运行和结果展示。Jupyter Notebook 是文学式开发工具，以网页的形式打开，用户可以在网页中直接编写和运行代码，运行结果也会直接在代码块下方显示。

Jupyter Notebook 的主要特点如下。

- 编程时具有语法高亮、缩进、Tab 补全的功能。
- 可直接通过浏览器运行代码，同时在代码块下方显示运行结果。
- 以富媒体格式展示计算结果。富媒体格式包括 HTML、LaTeX、PNG、SVG 等。
- 为代码编写说明文档或语句时，支持 Markdown 的语法。
- 支持使用 LaTeX 编写数学性说明。

安装 Jupyter Notebook 最简单的方法是使用 pip 命令，在命令行提示符中输入以下命令：

```
pip install jupyter
```

即可显示 Jupyter Notebook 的过程，如图 1-17 所示。在终端输入以下命令：

```
jupyter notebook
```



图 1-17　Jupyter Notebook 安装

提示

/path/to/pycharm/python -m jupyter notebook

上面的命令，启动一个使用 PyCharm 环境的 Jupyter Notebook。Jupyter Notebook 的运行依赖于特定的 Python 内核(Kernel)，而这个内核绑定到一个 Python 解释器。如果在 PyCharm 或其他地方安装了第三方库，Jupyter Notebook 即可使用该环境中已安装的库，无须额外安装。

运行 Jupyter Notebook 启动过程，之后自动打开 Jupyter Notebook 界面，如图 1-18 所示。

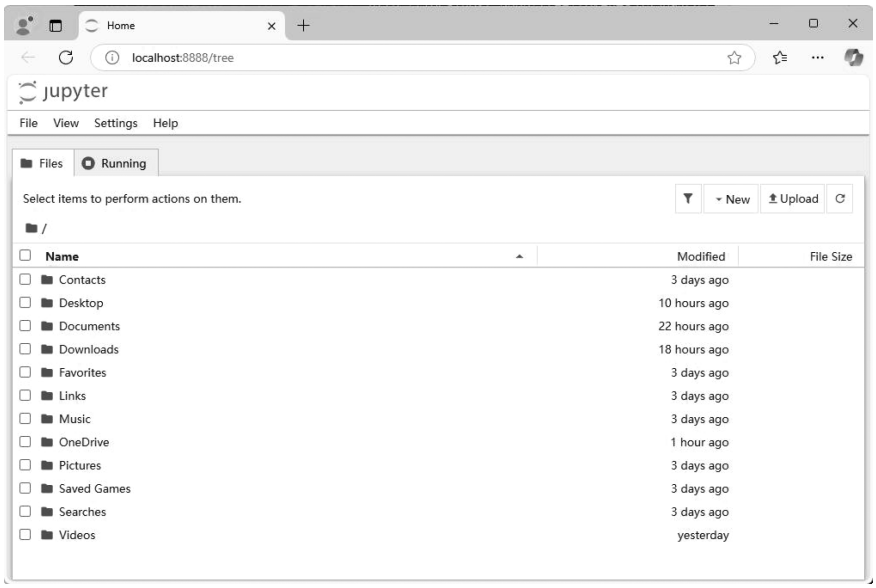


图 1-18 Jupyter Notebook 界面

实例 1.2 ——Jupyter 零售门店周营业额指标分析

某连锁便利店为优化门店运营策略，需快速获取单周销售核心指标。本实例基于 Python 原生数据结构(列表)存储周一至周日的营业额原始数据(单位：元)，使用 Jupyter 调用内置数学函数计算日均营业额、单日最高/最低销售额等关键指标。

【操作步骤】

(1)打开命令行提示符，输入下面的命令：

```
jupyter notebook
```

(2)运行该命令后，启动 Jupyter Notebook 并打开浏览器界面。在界面中，单击“Files”选项，在右上角的“New”下拉菜单中选择“Python 3(ipykernel)”选项，如图 1-19 所示。

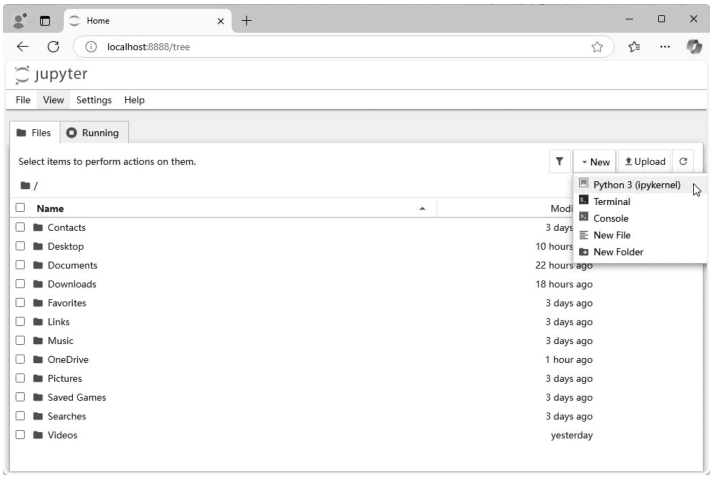


图 1-19 新建 Python 记事本文件命令

(3)此时，创建一个新的 Python 笔记本文件，并自动进入笔记本的编辑环境，如图 1-20 所示，在该界面即可开始编写和运行代码。

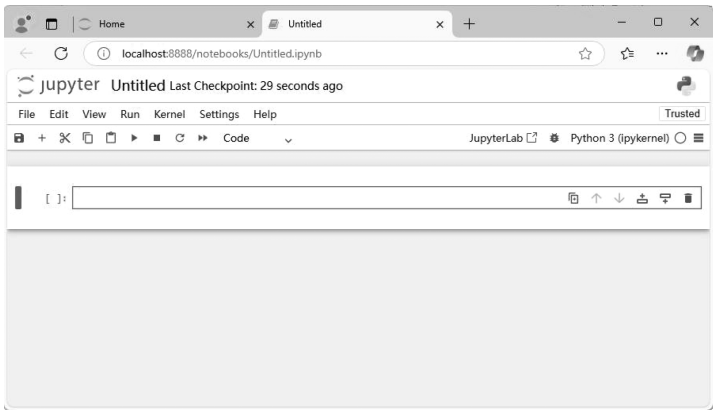


图 1-20　记事本编辑环境

(4)在文本框中输入如下程序(Py02.py)：

```
# 数据输入模块
days = ["Mon", "Tue", "Wed", "Thu", "Fri", "Sat", "Sun"]
sales_data = [1505, 1260, 1647, 1750, 1860, 1632, 1664]
# 统计计算模块
average = sum(sales_data) / len(sales_data)
max_sale = max(sales_data)
min_sale = min(sales_data)
# 结果输出模块
print("\n统计结果:")
print(f"平均营业额:{average:.2f}")
print(f"最高单日营业额:{max_sale:.2f}")
print(f"最低单日营业额:{min_sale:.2f}")
```

(5)单击“运行”按钮▶，或按快捷键 Ctrl+Enter，输出分析结果，如图 1-21 所示。

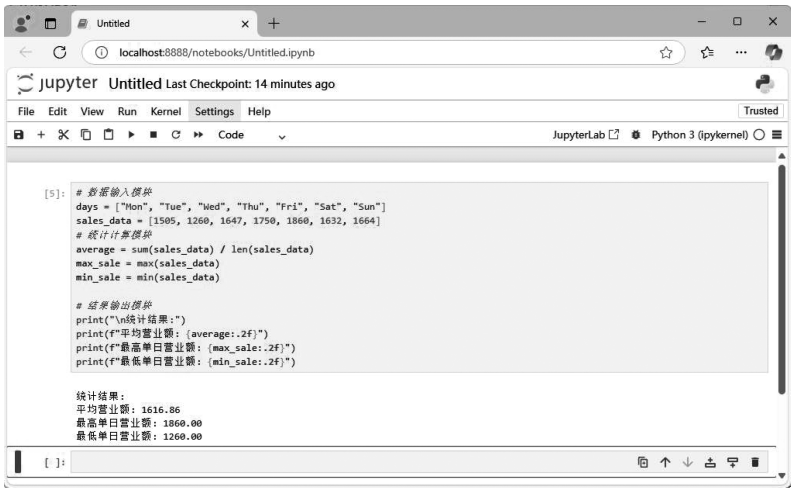


图 1-21　运行结果

运行结果：

```
统计结果：
平均营业额：1616.86
最高单日营业额：1860.00
最低单日营业额：1260.00
```

Jupyter Notebook 文件的保存类型有很多种。下面介绍如何保存为 Jupyter Notebook 专属格式和 Python 格式。

①保存为 Jupyter Notebook 专属格式。选择菜单栏中的保存命令，弹出如图 1-22 所示的“Save File As(另存为)”对话框，存储在默认保存位置，指定文件的名称为“Revenue Indicators.ipynb”，单击“Save”按钮，保存文件。返回“Home”主界面，可以发现，保存的 ipynb 文件，如图 1-23 所示。选中该文件，单击“Download”按钮，将该文件下载到计算机中。

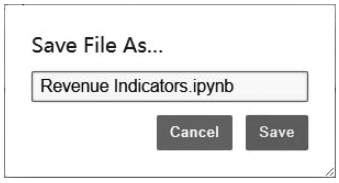


图 1-22 “Save As(另存为)”对话框

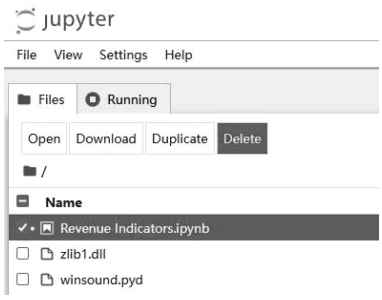


图 1-23 保存 ipynb 文件

②保存为 Python 格式。选择菜单栏中的“File”→“Save and Export Notebook As”命令，显示图 1-24 所示的文件类型子菜单。导出功能支持代码版本控制与跨环境迁移，导出的脚本文件将保留 Notebook 中的代码逻辑，但不会包含 Markdown 文本和输出结果。

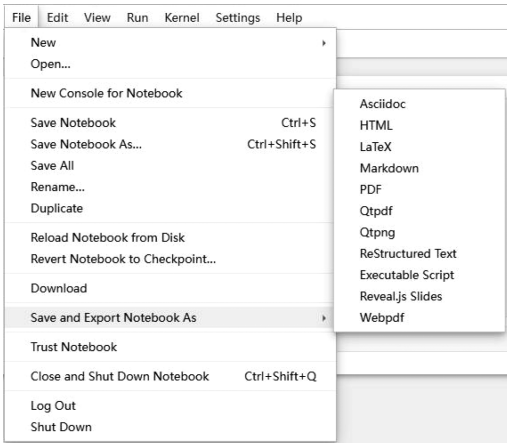


图 1-24 下载文件类型菜单

三、PyCharm

PyCharm 是由 JetBrains 打造的一款 Python IDLE。PyCharm 具备一般 Python IDLE 的功能，如调试、语法高亮、项目管理、代码跳转、智能提示、自动完成、单元测试、版本控制等。另外，PyCharm 还提供了一些很好的功能用于 Django 的开发，同时支持 Google App Engine 和 IronPython。

1. 下载软件

访问 JetBrains 的 PyCharm 官方网站进行下载，官网下载地址为：https://www.jetbrains.com/PyCharm/download/。在官网页面选择操作系统(Windows，macOS 或 Linux)并下载最新的 PyCharm 社区版，如图 1-25 所示。值得注意的是，根据 JetBrains 的计划，2025.2 版本将是最后一个独立的 PyCharm 社区版。从 2025.3 版本开始，社区版的功能将整合进一个统一的 PyCharm 产品中，但核心的 Python 开发功能将继续免费提供。因此，现在下载的版本即可使用到稳定且功能强大的免费 Python IDE。

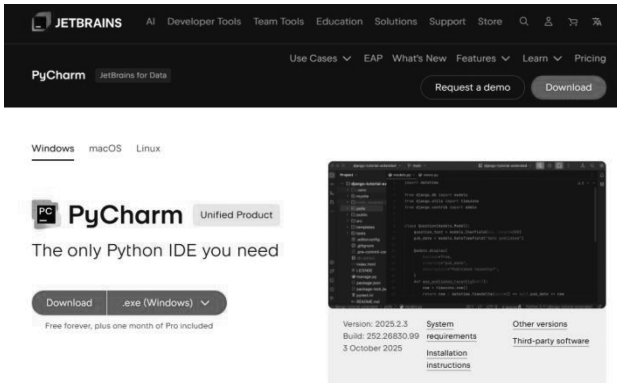


图 1-25　PyCharm 官网

2. 安装 PyCharm 的具体过程

双击 pycharm-community-2025.2.3.exe 文件，弹出安装界面，如图 1-26 所示。



图 1-26　安装界面

单击“下一步”按钮，进入下一个界面。在该界面中，用户需要选择 PyCharm 的安装路径，可以通过单击“浏览”按钮，自定义其安装路径，如图 1-27 所示。

单击“下一步”按钮，出现安装选项设置界面，需要进行一些设置，如图 1-28 所示。

- PyCharm：勾选该复选框，在桌面上创建一个名为“PyCharm”的快捷方式，方便快速启动程序。
- 添加"bin"文件夹到 PATH：勾选该复选框，将 PyCharm 的启动目录添加到环境变量中，执行该操作后，需要重启计算机。
- 添加“将文件夹作为项目打开”：若勾选该复选框，添加鼠标右键菜单，使用打开项目的方式可以打开此文件夹。
- .py：勾选该复选框后，以后在打开 .py 文件时就会用 PyCharm 打开，此时 PyCharm 每次打开的速度都会比较慢。

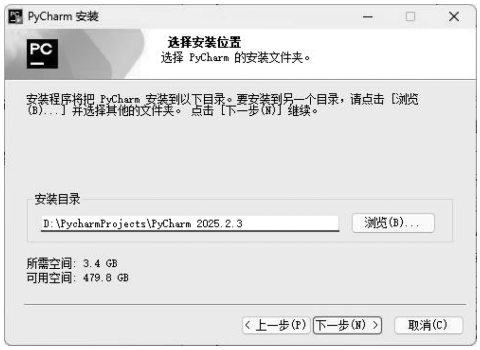


图 1-27　选择安装路径



图 1-28　安装选项设置界面

单击“下一步”按钮，默认选择“JetBrains”选项，如图 1-29 所示，单击“安装”按钮，此时会显示安装进度，如图 1-30 所示，需要等待安装完毕。

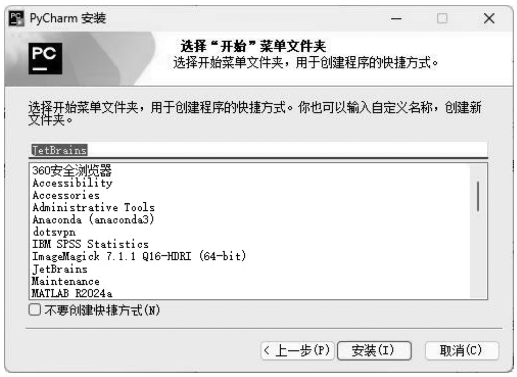


图 1-29　选择安装菜单文件



图 1-30　安装进度

安装结束后如图 1-31 所示。单击“完成”按钮即可完成 PyCharm 的安装工作。



图 1-31　安装结束后

3. PyCharm 的配置

双击运行桌面上的 PyCharm 图标，激活 PyCharm 启动界面，然后跳转到“语言和地区”对话框，选择“Chinese(Simplified)中文语言包”“中国大陆”选项，如图 1-32 所示。单击“下一个”按钮，进入“用户协议”对话框，勾选“我确认我已阅读并接受此《用户协议》的条款”复选框。



图 1-32　“语言和地区”设置界面

单击“继续”按钮，跳转到 PyCharm 欢迎界面 1。该界面包括项目、自定义、插件、学习四个选项卡。值得注意的是，此时界面的显示语言为中文，如图 1-33 所示。单击“自定义”选项卡，设置编辑区颜色参数。在“主题”选项下拉列表中显示 5 个主题，根据需要选择“Light with Light Header”(白色高亮主题)，自动更新界面颜色，如图 1-34 所示。

- Dark：提供黑色界面，默认选择该项。

- Light：白色主题。
- Light with Light Header：白色高亮主题。本书后续的编辑器界面将采用白色主题进行讲解。
- High contrast：黑色高对比度主题。
- Darcula：黑色经典主题，比 Dark(黑色主题)颜色要浅。



图 1-33　PyCharm 欢迎界面 1



图 1-34　自定义选项卡

此时，单击“项目”选项卡，进入 PyCharm 欢迎界面 2，如图 1-35 所示。



图 1-35　PyCharm 欢迎界面 2

四、新建 PyCharm 项目文件

在“项目”选项卡中，单击“新建项目”按钮，在弹出的“新建项目”对话框中，选择“纯 Python”选项卡，设置项目路径，并将项目名称文件夹命名为 Python _ Analysis，如图 1-36 所示。

(1)创建 Git 仓库：勾选此项，PyCharm 会在项目目录下创建一个 Git 仓库，用于进行版本控制。

(2)创建欢迎脚本：勾选此项，PyCharm 会在项目目录下创建一个名为 main. py 的示例 Python 文件。

(3)解释器类型。

①项目 venv：选择使用 venv 模块来创建项目的虚拟环境。venv 是 Python 自带的虚拟环境工具(Python 3. 3 及以上版本)，它轻量且易于使用，这种设计使项目依赖的第三方库仅安装在虚拟环境中，避免不同项目间的库版本冲突。

②基础 conda：专门用于管理 Conda 环境。

③自定义环境：对虚拟环境进行更详细的配置。

- 生成新的：表示要创建一个全新的虚拟环境。这是大多数情况下的首选。
- 选择现有：表示可以选择一个已经存在的虚拟环境作为当前项目的解释器。
- 类型：指定用于创建虚拟环境的工具。
- 从基础解释器继承软件包：不建议勾选此项。如果勾选此项，新创建的虚拟环境将能够访问系统全局安装的 Python 包。

• 可用于所有项目：如果勾选此项，新创建的虚拟环境将被添加到 PyCharm 的全局虚拟环境列表中。

在“解释器类型”选项中选择“项目 venv”，创建虚拟环境。在“基础 Python”下拉列表中，选择解释器版本 Python 3.14.0。

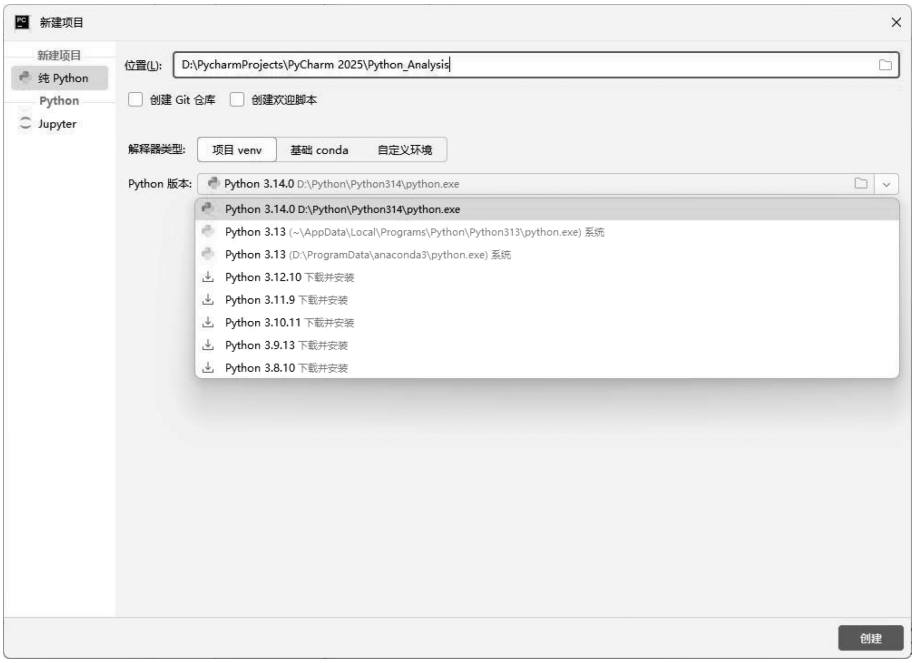


图 1-36 “新建项目”对话框

单击“创建”按钮后，PyCharm 将自动打开编辑器界面。此时不仅完成了项目目录的创建，同时也完成了开发环境的配置。在左侧的“项目”导航器中，可以看到新建的项目，如图 1-37 所示。在项目的源文件目录下，系统会自动生成名为 Python _ Analysis 的文件夹，其中包含 .idea 和 .venv 两个文件夹，分别用于存储项目配置信息和虚拟环境文件，如图 1-38 所示。

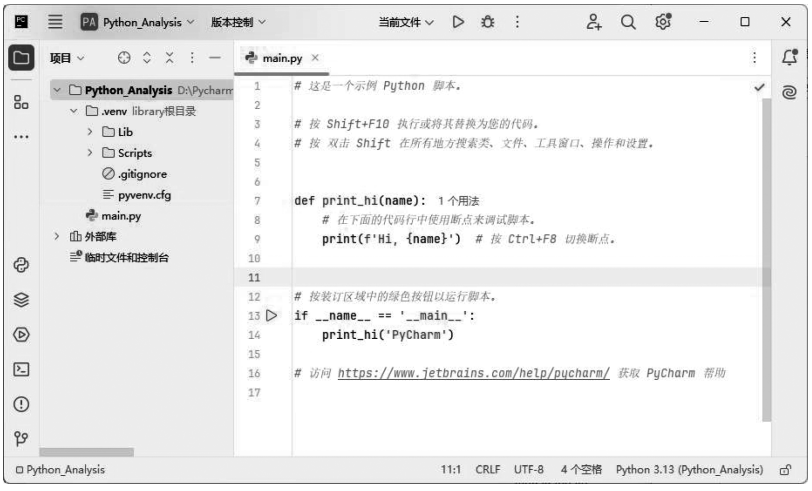


图 1-37 PyCharm 编辑器界面

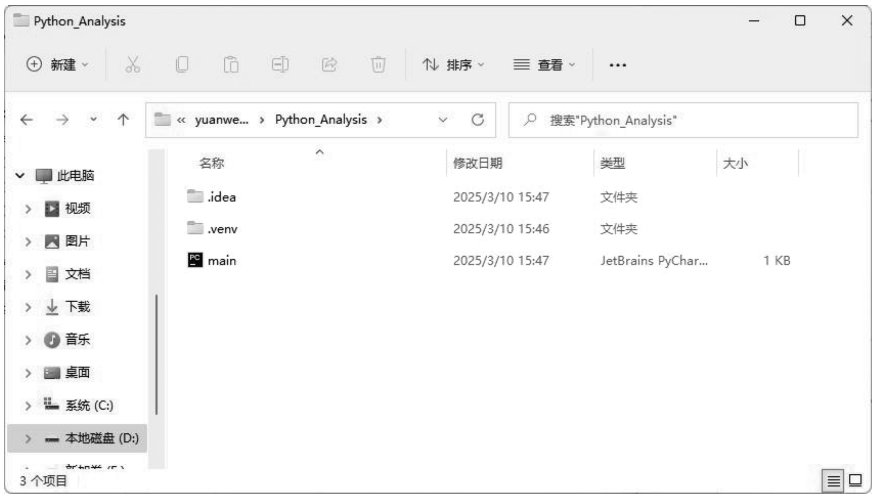


图 1-38　Python _ Analysis 文件夹

五、新建 PyCharm 文件

进入 PyCharm 项目文件后，需要创建 Python 文件，只有这样才可以进入 PyCharm 与 Python 交互代码编译界面。Python 文件的创建包含以下两种方式。

1. 通过菜单命令创建

在 PyCharm 中创建一个新文件最简便的方法是在“文件”→“新建”子菜单中选择一种适当的文件类型。选择菜单栏中的“文件”→“新建”命令，打开如图 1-39 所示的子菜单。选择“Python 文件”命令，弹出“新建 Python 文件”对话框，Python 文件包含 3 种类型：Python 文件、Python 单元测试、Python 存根。选择“Python 文件”选项，输入文件名称 python _ file _ 01，如图 1-40 所示，按 Enter 键，进入新建的文件，即 python _ file _ 01. py 的编辑界面。

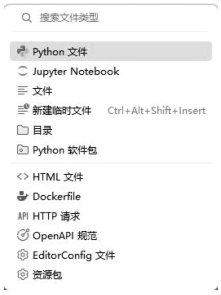


图 1-39　新建文件类型

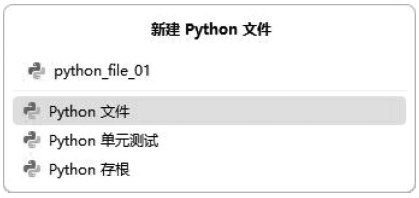


图 1-40　“新建 Python 文件”对话框

2. 利用右键快捷命令

在项目文件上单击鼠标右键，选择“新建”→“Python 文件”命令。

六、PyCharm 的编辑环境

PyCharm 的编辑环境如图 1-41 所示，其工作界面由菜单栏、工具栏、工作区、项目面板、运行面板、状态栏组成。



图 1-41　PyCharm 的编辑环境

PyCharm 的程序运行包含以下 3 种方式。

- 在命令行窗口中，在需要运行的代码上单击鼠标右键，选择“运行”命令。
- 单击菜单栏下方工具栏中的“运行”按钮▷。
- 快捷键 Ctrl＋Shift＋F10。

提示

对于同一个脚本，第一次运行使用右键快捷命令“运行”。

实例 1.3——程序运行演示

本实例简单输入一段程序，演示 PyCharm 程序的运行过程。

【操作步骤】

(1)打开 PyCharm，在 Py03.py 的命令行窗口中输入下面的程序：

```
print('Python 数据分析')
```

(2)按下快捷键 Ctrl＋Shift＋F10，“运行”面板中显示程序运行结果，如图 1-42 所示。

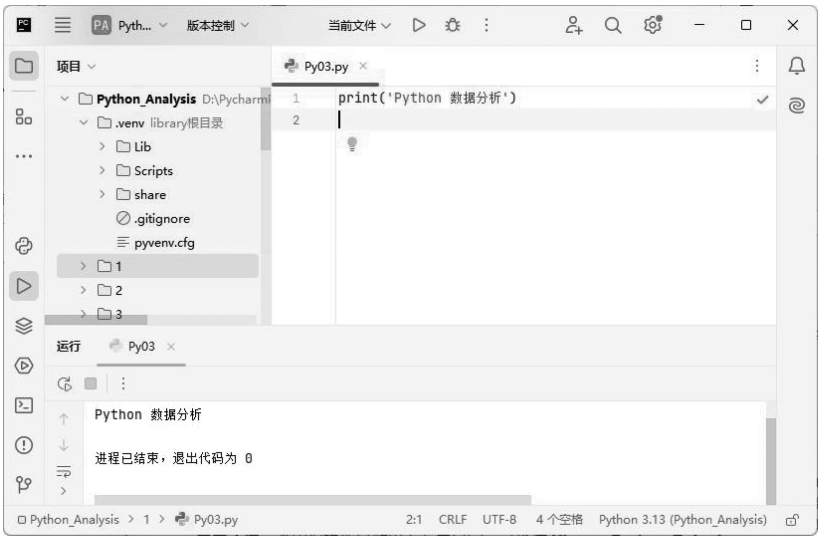


图 1-42　程序运行

(3)单击左上角≡按钮，打开菜单栏。选择菜单栏中的“工具”→“Python 或调试控制台”命令，在 PyCharm 中打开“Python 控制台”窗口，在“运行”面板中显示命令行提示符“>>>”，如图 1-43 所示。

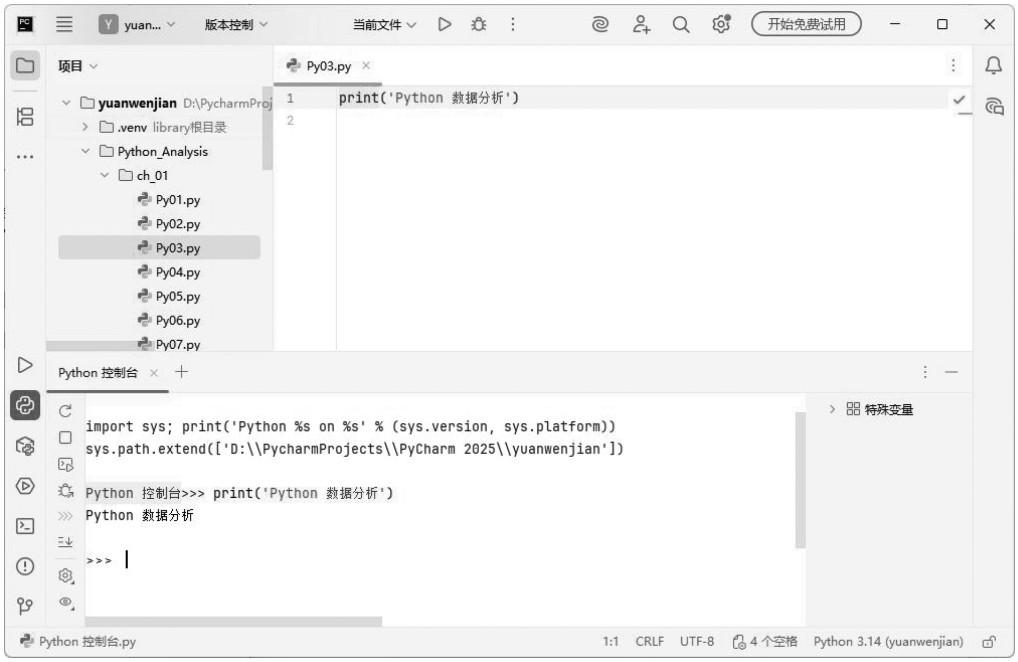


图 1-43　“Python 控制台”窗口

(4)输入下面的程序：

```
>>>print('Python 数据分析')
```

运行结果如下：

```
Python 数据分析
```

PyCharm 中的部分快捷键如下。

- Ctrl+Enter：在下方新建行但不移动光标。
- Shift+Enter：在下方新建行并将光标移到新行行首。
- Ctrl+/：注释(取消注释)选择的行。
- Ctrl+D：对光标所在行的代码进行复制。

任务三 Python 库管理

任务引入

在数据处理项目中，小李发现不同模块间的依赖冲突严重影响了开发进度。经过调研，他意识到规范的库管理是解决问题的核心。Python 生态中，PyPI 托管了超过 70 万个第三方库，如何精准安装所需版本？如何通过虚拟环境隔离项目依赖？建立标准化的依赖管理流程，将成为提升协作效率的关键。

知识准备

Python 库管理是开发中高效组织代码与复用功能的关键环节，其核心围绕库的分类、安装及模块化设计展开。



Python 模块体系

一、Python 模块体系

从系统设计层面来看，Python 模块体系可分为三个层级：标准库(内置模块)、第三方模块(通过 pip 或 pip3 等安装)和应用程序自定义模块。这种模块化不仅能组织代码结构、实现代码复用，还能避免不同模块间函数和变量的命名冲突。

注意

虽然标准库已存在于系统中，但它们并不会随解释器的启动而自动加载(除了少数极核心的内置函数)。要使用这些功能，必须先通过 import 语句显式导入。

1. Python 标准库

Python 标准库相当于解释器的外部扩展，它并不会随着解释器的启动而启动，要想使用这些外部扩展，必须提前导入。Python 内置的数据库很庞大，可以帮助用户处理各种工作，包括正则表达式、文档生成、单元测试、线程、数据库、网页浏览器、CGI、FTP、电子邮件、XML、XML-RPC、HTML、WAV 文件、密码系统、GUI(图形用户界面)、Tk 和其他

与系统有关的操作。下面按照模块分类介绍 Python 标准库。

(1)数学运算与数字处理。

- math：提供实数范围内的数学函数(三角函数、对数等)。
- decimal：提供高精度十进制运算，适用于对精度要求极高的金融场景。
- random：用于生成各种分布的伪随机数。
- statistics：提供基础统计功能(均值、中位数、方差等)。
- numbers：数字抽象基类，用于类型检查和层次结构。
- cmath：复数的数学函数。
- fractions：有理数(分数)运算。

(2)数据类型。

- datetime/calendar：处理日期、时间格式转换及日历功能。
- collections：提供高性能的容器包装器。
- copy：支持对象的浅拷贝与深拷贝。
- enum：支持枚举类型的定义。
- array/heapq/bisect：分别提供高效数值数组、堆队列算法及二分查找优化。

(3)文件与目录操作。

- pathlib：提供面向对象的文件系统路径操作方案。
- os. path：传统的路径字符串处理函数集。
- shutil：高级文件操作工具，支持文件递归复制、移动及目录树删除。
- tempfile/glob：用于生成临时文件及 Unix 风格的路径模式匹配。

(4)数据持久化与格式处理。

- json/csv：处理通用的 JSON 和 CSV 数据格式。
- sqlite3：内置的轻量级关系型数据库接口。
- pickle/shelve：Python 对象序列化与轻量级持久化存储。
- configparser：解析类似 Windows INI 格式的配置文件。

(5)系统与网络通信。

- os：提供通用的操作系统服务接口(环境变量、进程管理等)。
- sys：访问与解释器紧密相关的参数和函数(如 sys. path，sys. argv)。
- logging：官方标准日志记录工具，支持多级别日志输出。
- argparse：专业的命令行参数解析器。
- socket/ssl：底层网络通信与安全加密协议支持。
- http/urllib：处理 HTTP 协议及 URL 请求的基础库。

2. Python 三方库

下面介绍 Python 常用的第三方库，所有第三方库均需要下载、安装、导入后才可以应用。第三方库下载安装后，可以导入使用。Python 入的三方库中模块函数比 Python 内置的函数多，功能强大。在后面模块中介绍添加三方库过后的 Python 中的数据与函数。

(1)数据科学与人工智能。

- NumPy/SciPy：科学计算的基础，支持多维数组运算与高级算法。
- Pandas：数据分析的核心，提供强大的 DataFrame 结构处理表格数据。
- Matplotlib/Seaborn：数据可视化，生成专业统计图表。
- Scikit-learn：机器学习算法库，涵盖分类、回归与聚类。
- PyTorch/TensorFlow：深度学习与神经网络构建的行业标准。
- Transformers：自然语言处理(NLP)领域的主流库，提供了加载和使用 GPT、BERT 等多种预训练模型的统一接口。

(2)Web 开发与网络服务。

- Django：全能型 Web 框架，适合快速构建大型企业级应用。
- Flask/FastAPI：轻量级 Web 框架，FastAPI 以高性能和自动生成文档著称。
- Requests/Httpx：目前最流行的 HTTP 客户端库，简单易用。
- BeautifulSoup/Scrapy：网页解析与大规模分布式爬虫框架。

(3)自动化、测试与系统工具。

- Selenium/Playwright：浏览器自动化测试与动态网页抓取。
- Pytest：现代化的 Python 单元测试框架，简洁且插件丰富。
- Loguru：极简的新型日志库，旨在替代传统的 logging。
- Rich/Click：用于创建美观的终端界面(CLI)和彩色命令行输出。

(4)图像处理与多媒体。

- OpenCV：计算机视觉领域的权威库，支持实时人脸识别与图像处理。
- Pillow：常用的基础图像处理库(缩放、裁剪、格式转换)。
- Pygame：专为 2D 游戏开发设计的引擎。

(5)数据库与工程自动化。

- SQLAlchemy：最强大的 ORM 库，将数据库表映射为 Python 对象。
- pywin32：在 Windows 平台上调用系统 API、控制 Excel/Word 等 Office 组件。
- jira/python-jenkins：用于 DevOps 流程中的项目管理与持续集成自动化。

二、安装 Python 库

Python 的一大特色便是其丰富的模块库，尤其在数据分析领域，像 NumPy、Pandas、Matplotlib 和 SciPy 等库提供了强大的功能，能够帮助读者进行高效的数据处理、分析和可视化。为了使用这些第三方库，需要首先进行安装。

Python 包管理工具 pip(在 python3 中，pip 与 pip3 一致)作为生态体系的核心组件，能够有效实现第三方库的检索、安装、版本控制及依赖管理。该工具默认随 Python 3.4 及以上版本自动部署，开发者可通过命令行界面快速执行各类操作。

1. 环境维护与工具更新

建议定期执行以下命令，升级 pip 以获取最新功能与安全修复：

```
python -m pip install --upgrade pip
```

在 Windows 系统中需以管理员权限运行终端，升级操作可避免因版本滞后引发的依赖冲突或兼容性问题。

2. PyCharm 对话框安装第三方库

NumPy(numerical Python)是 Python 的一种开源的数值计算扩展库，用来存储和处理大型矩阵，比 Python 自身的嵌套列表结构要高效得多，支持大量的维度数组与矩阵运算，还针对数组运算提供了大量的数学函数库。

在“开始”下面的搜索栏中输入“cmd”，打开“命令提示符”界面，显示下面的用户名(不同的用户显示的用户名不同)：

```
C:\Users\yan>
```

在窗口中输入：

```
pip3 install numpy
```

出现 NumPy 的安装过程信息，显示程序如下：

```
C:\Users\yan>pip3 install numpy
Collecting numpy
Downloading numpy-2.2.1-cp313-cp313-win_amd64.whl (14.0 MB)
   | ████████████████████████████████ | 14.0 MB 218 kB/s
```

安装进度结束后，显示安装成功信息：

```
Installing collected packages: numpy
Successfully installed numpy-2.2.1
```

上面的命令将自动从 PyPI 仓库获取最新稳定版本。针对网络受限环境，可通过“-i”参数指定国内镜像源加速下载：

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

提示

需要注意的是，Python 中的库是经常更新的，因此当读者使用时，库的版本可能已经发生变化。如果不指定版本，pip 默认会安装最新版本的库。然而，随着库的更新，某些函数和功能可能会发生变化，甚至会被弃用或替换。因此，使用不同版本的库时，函数的行为和接口可能会有所不同。

如果希望获得与书中示例一致的功能和输出，可以通过指定库的版本来确保一致性。例如，可以使用以下命令安装特定版本的 NumPy：

```
pip install numpy==2.2.1
```

通过指定版本，可以确保安装的库版本与书中的版本一致，从而避免因版本不同而导致

的差异。

3.“可用软件包”对话框安装模块

如果选择在 PyCharm 集成开发环境中进行数据分析工作，仅仅通过 pip3 安装这些模块可能还不够。为了让 PyCharm 能够正确识别和使用这些模块，通常需要在 PyCharm 项目中进行额外的配置。

打开 PyCharm，选择菜单栏中的“文件”→“设置”命令，打开“设置”对话框，打开“Python”→“解释器”对话框，单击“安装”按钮+，弹出“可用软件包”对话框。

在搜索框中输入需要安装的数组计算库 Numpy，在列表框中选择“numpy”选项，如图 1-44 所示。单击“安装软件包”按钮，即可安装该模块库。

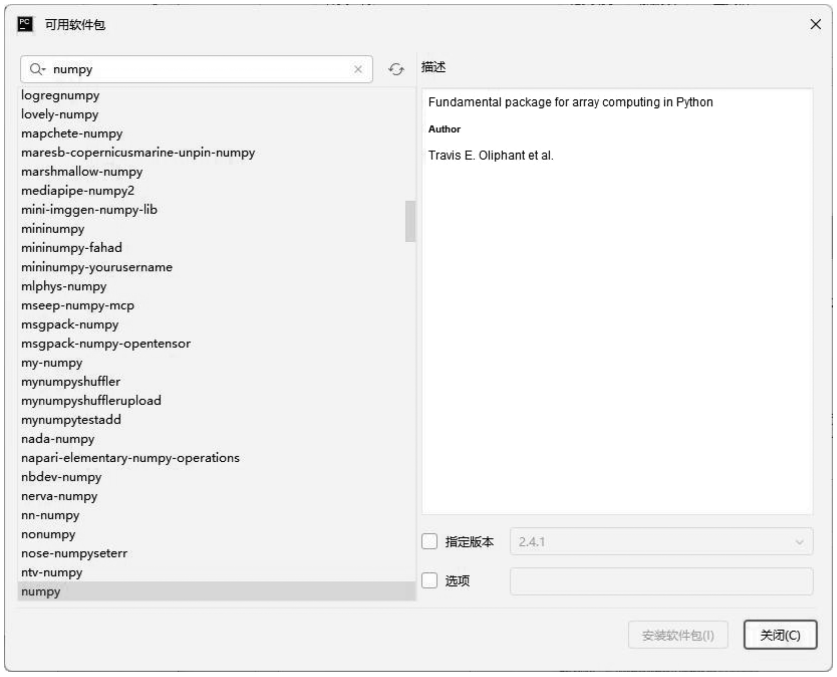


图 1-44 “可用软件包”对话框 1

在搜索框中输入需要安装的数据分析库 Pandas，在列表框中选择“pandas”选项，如图 1-45 所示。单击“安装软件包”按钮，即可安装该模块库。

在搜索框中输入需要安装的数据可视化库 Matplotlib，在列表框中选择“matplotlib”选项，如图 1-46 所示。单击“安装软件包”按钮，即可安装该模块库。

在搜索框中输入需要安装的数据分析库 SciPy，在列表框中选择“scipy”选项，如图 1-47 所示。单击“安装软件包”按钮，即可安装该模块库。

单击“确定”按钮，关闭该对话框，返回“可用软件包”对话框。显示安装的所有库。

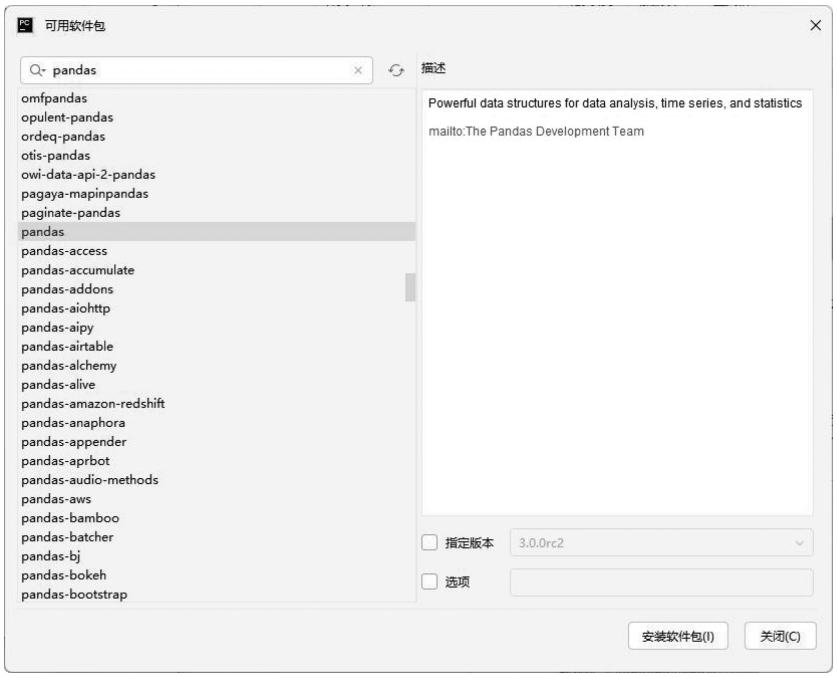


图 1-45 “可用软件包”对话框 2

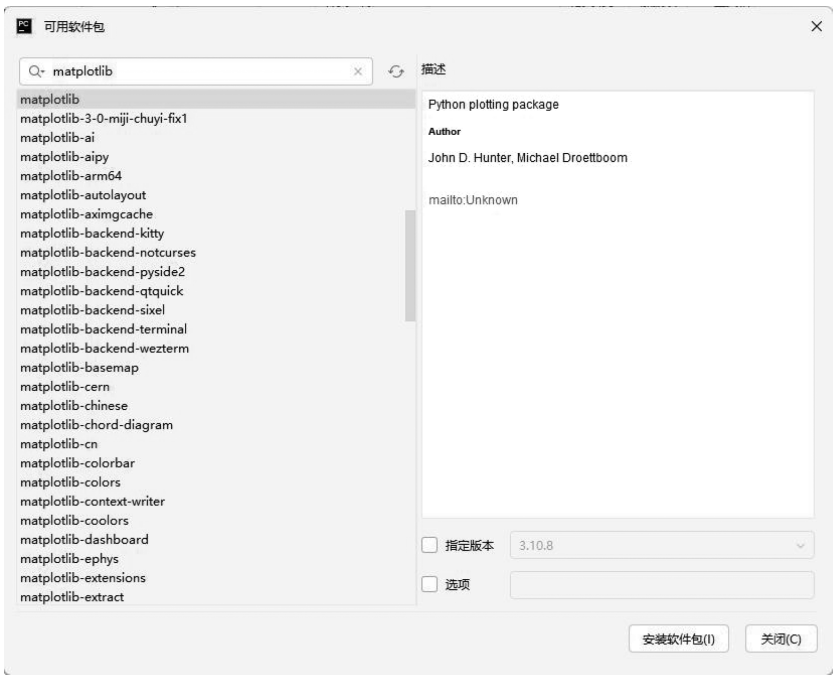


图 1-46 “可用软件包”对话框 3