第5章

数据可视化基础与应用

本章学习目标

- 了解大数据可视化的定义。
- 了解大数据的特征及技术框架。
- 了解不同的大数据图表分类。
- 掌握 numpy 数据可视化原理。
- 掌握 matplotlib 数据可视化基础。
- 掌握 pyecharts 数据可视化基础。
- 掌握大数据可视化应用。

本章先向读者介绍大数据可视化的概念，再介绍大数据可视化的图表，接着介绍大数据可视化的实现基础，最后介绍各种大数据可视化的应用。

5.1 数据可视化

5.1.1 数据可视化概述

1. 数据可视化简介

数据永远是枯燥的，而图形图像却具有生动性。数据可视化是关于数据视觉表现形式的科学技术研究，它让大数据更有意义，更贴近大多数人，因此大数据可视化是艺术与技术的结合。它将各种数据用图形化的方式展示给人们，从根本上讲，数据可视化系统并不是为了展示用户已知的数据之间的规律，而是为了帮助用户通过认知数据有新的发现，

发现这些数据所反映的实质。

数据可视化是一个相对的概念，它通过将数据转换为标识为人们提供帮助与指导，并最终成为通过数据分析传递信息的一种重要工具。与传统的立体建模之类的特殊技术方法相比，数据可视化所涵盖的技术方法要广泛得多，它是利用计算机图形学及图像处理技术将数据转换为图形或图像形式显示到屏幕上，并进行交互处理的理论、方法和技术。它涉及计算机视觉、图像处理、计算机辅助设计和计算机图形学等多个领域，并逐渐成为一项研究数据表示、数据综合处理、决策分析等问题的综合技术。

2. 数据可视化历史介绍

1）数据可视化的起源

数据可视化作品的出现最早可追溯到10世纪。当时一位不知名的天文学家绘制了一幅作品，其中包含了很多现代统计图形元素，例如坐标轴、网格和时间序列，如图5-1所示。

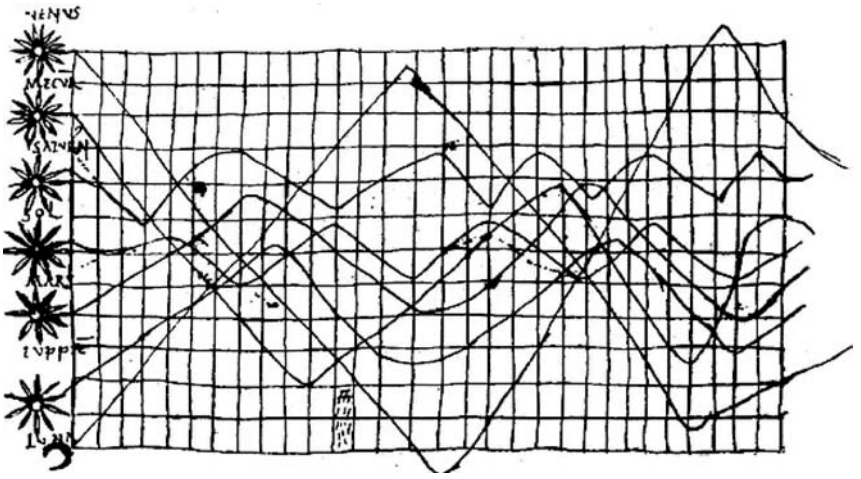

图5-1　数据可视化的起源

2）17世纪，最早的地图和图表

到了17世纪，随着社会的进一步发展与文字的广泛应用，微积分、物理、化学和数学等都开始蓬勃发展，统计学也开始出现了萌芽。数据的价值开始被人们重视起来，人口、商业、农业等经验数据开始被系统地收集整理，记录下来，于是各种图表和图形也开始诞生。

值得一提的是苏格兰工程师William Playfair（1759—1823），他创造了今天人们习以为常的几种基本数据可视化图形，即折线图、饼图（见图5-2）和条形图（见图5-3）。

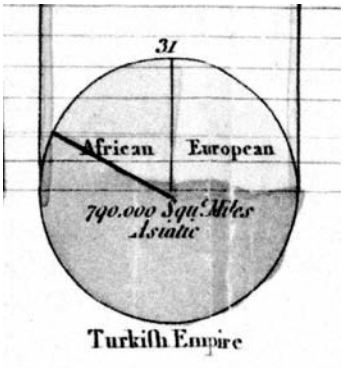

图5-2　William Playfair绘制的饼图

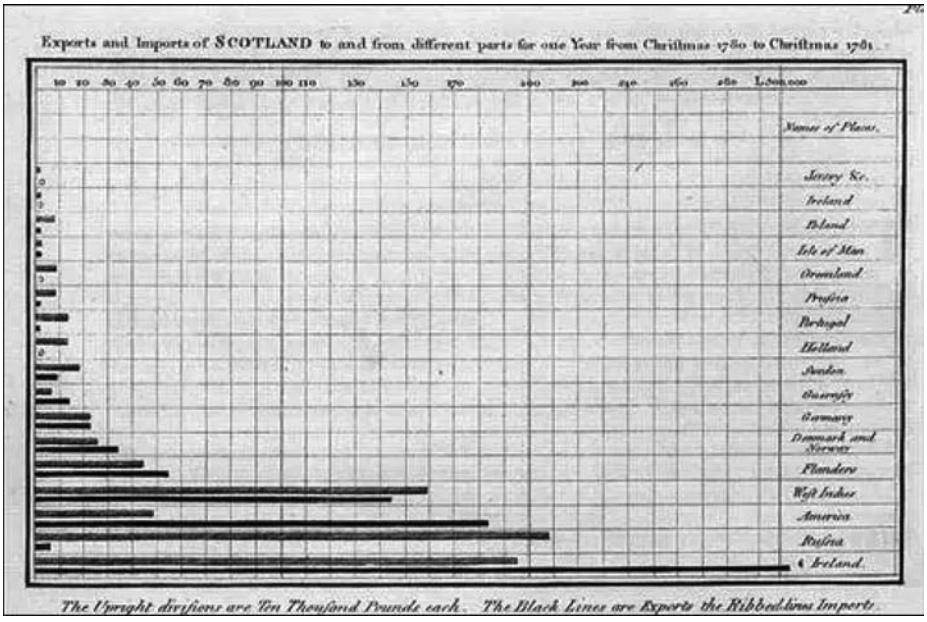

图 5-3　William Playfair 绘制的条形图

3）19 世纪，数据绘图的广泛应用

进入 19 世纪以后，随着科技迅速发展，工业革命从英国扩散到欧洲大陆和北美。社会对数据的积累和应用的需求与日俱增，现代的数据可视化慢慢开始成熟，统计图形和主题图的主要表达方式在随后的几十年间逐渐都出现了。在 19 世纪，数据可视化的重要发展包括：统计图形方面，散点图、直方图、极坐标图形和时间序列图等当代统计图形的常用形式都已出现；主题图方面，主题地图和地图集成为这个年代展示数据信息的一种常用方式，应用领域涵盖社会、经济、疾病、自然等各个主题。图 5-4 所示的是英国著名的护

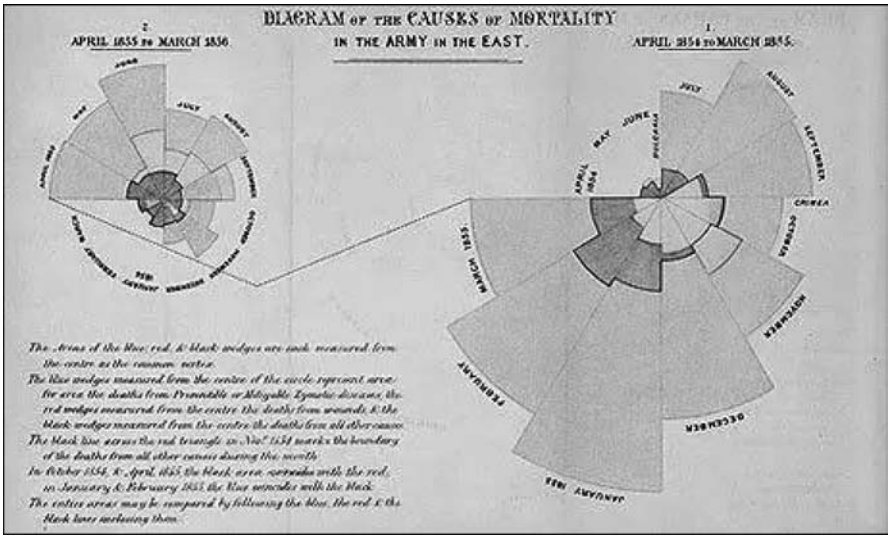

图 5-4　南丁格尔绘制的可视化图形

士和统计学家南丁格尔绘制的统计英军伤亡人数的可视化图形。

19世纪，在可视化图形的绘制中，统计图表成为主流，与此同时，有很多相关的教科书也都对数据图表的绘制进行了详细的描述。

4）20世纪，大数据绘图的低谷

在19世纪结束后，数据可视化的发展也随之进入了一个低谷，原因主要在于第一次世界大战和第二次世界大战的爆发对经济产生了深远的影响。此外，数理统计的诞生也使众多科学家们将数学基础作为首要目标，而图形作为一个辅助工具被搁置起来。直到20世纪下半叶，随着计算机技术的兴起，以及在世界大战后的工业和科学快速发展阶段，都把统计和数据问题放在重要的位置，在各行业的实际应用中，图形表达又重新占据了重要的地位。

5）21世纪，大数据可视化的日新月异

进入21世纪以来，计算机技术获得了长足的进展，随着数据规模呈指数级的增长，数据的内容和类型也比以前要丰富得多，这些都极大地改变了人们分析和研究世界的方式，也给人们提供了新的可视化素材，推动了数据可视化领域的发展。数据可视化依附计算机科学与技术拥有了新的生命力，并进入了一个新的黄金时代。

大数据可视化已经注定成为可视化历史中新的里程碑，VR、AR、MR、全息投影……这些当下火热的数据可视化技术已经被应用到游戏、房地产、教育等各行各业。因此，人们应该深刻地认识到数据可视化的重要性，更加注重交叉学科的发展，并利用商业、科学等领域的需求来进一步推动大数据可视化的健康发展。

3. 数据可视化的方法与组成

1）数据可视化的方法

数据可视化技术包含以下几个基本概念。

（1）数据空间：由n维属性和m个元素组成的数据集所构成的多维信息空间。

（2）数据开发：指利用一定的算法和工具对数据进行定量的推演和计算。

（3）数据分析：指对多维数据进行切片、块、旋转等动作剖析数据，从而能多角度、多侧面观察数据。

（4）数据可视化：指将大型数据集中的数据以图形图像的形式表示，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。

目前，大数据可视化的方法根据不同原理可以划分为基于几何的技术、面向像素的技术、基于图表的技术、基于层次的技术、基于图像的技术和分布式技术等。

大数据可视化的实施是一系列数据的转换过程，它从最初的原始数据到最终的图像经历了多个步骤，如图5-5所示。

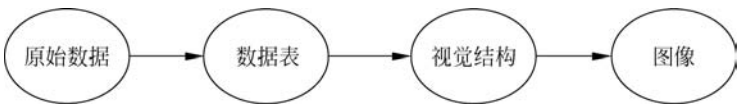

图5-5　大数据可视化的实施过程

从图 5-5 可以看出，人们首先通过对原始数据进行标准化、结构化的处理，把它们整理成数据表；再将数据表中的各种数值转换成视觉结构（包括形状、位置、尺寸、值、方向、色彩、纹理等），通过视觉的方式把它表现出来；最后将视觉结构进行组合，把它转换成图形传递给用户，用户通过人机交互的方式进行反向转换，去更好地了解数据背后隐藏的问题和包含的规律。

从技术角度上看，大数据可视化的实现是对海量数据进行分析处理的成果，它为用户创造出了既有意义又具人性化的可视化信息，使得枯燥的数据能够被人们所理解和接受。图 5-6 和图 5-7 所示的是大数据可视化的图像应用。

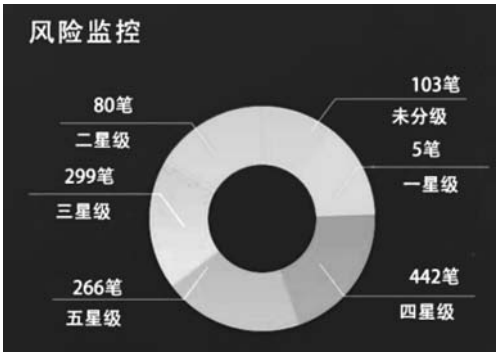

图 5-6　大数据可视化风险监控图像

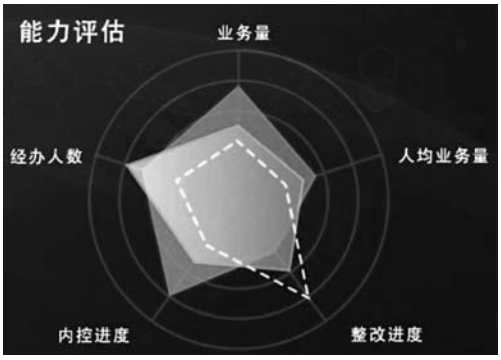

图 5-7　大数据可视化能力评估图像

2）数据可视化的组成

数据可视化技术一般由下列三方面组成。

（1）科学可视化。科学可视化主要关注三维现象的可视化，包含气象学、生物学、物理学、农学等，重点在于对客观事物的体、面及光源等的逼真渲染。

（2）信息可视化。信息可视化将数据信息和知识转换为一种视觉形式，在信息可视化中充分利用了人们对可视模式快速识别的自然能力。

（3）可视化分析。可视化分析是科学可视化与信息可视化领域发展的产物，侧重于借助交互式的用户界面对数据的分析与推理。

图 5-8 所示的是数据可视化的组成。

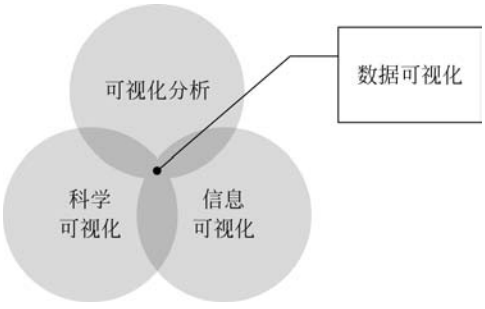

图 5-8　数据可视化的组成

4. 数据可视化的标准

数据可视化的标准通常有以下几点。

1）实用性

衡量数据实用性的主要参照是要满足使用者的需求，需要清楚地知道这些数据是不是人们想要了解的、与他们切身相关的信息。例如将气象数据可视化就是一个与人们切身相关的事情，因此实用性是一个较为重要的评价标准，它是一个主观的指标，也是评价体系中不可忽略的一环。

2）完整性

衡量数据完整性的重要指标是该可视化的数据应当能够帮助使用者全面而完整地理解数据的信息。其中包含要呈现的是什么样的数据、该数据有何背景、该数据来自何处、这些数据是被谁使用的、需要起到什么样的作用和效果、想要看到什么样的结果、是针对一个活动的分析还是针对一个发展阶段的分析、是研究用户还是研究销量等。

3）真实性

可视化的真实性考量的是信息的准确度和是否有据可依。如果信息是能让人信服的、精确的，那么它的准确度就达标了，否则该数据的可视化工作就不会令人信服，因此在实际的使用中应当确保数据的真实性。

4）艺术性

艺术性是指数据的可视化呈现应当具有艺术性，符合审美规则。不美观的数据图无法吸引读者的注意力，美观的数据图则可能会进一步引起读者的兴趣，提供良好的阅读体验。有一些信息容易让读者遗漏或者遗忘，通过美好的创意设计，可视化能够给读者更强的视觉刺激，从而有助于信息的提取。例如，在一个做对比的可视化中，让读者比较形状、大小或者颜色深浅都是不明智的设计，相比之下，位置的远近和长度一目了然。

5）交互性

交互性是指实现用户与数据的交互，方便用户控制数据。在数据可视化的实现中应多采用常规图表，并站在普通用户的角度，在系统中加入符合用户思考方式的交互操作，让大众用户也可真正地和数据对话，探寻数据对业务的价值。

5. 数据可视化的应用

目前大数据的可视化应用十分广泛，从政府机构到金融机构，从医学到工业，以及电子商务行业中都有数据可视化的身影。

1）政府机构

我国“十三五”规划纲要中明确指出，要实施国家大数据战略，这充分体现了政府对大数据的重视。该战略不仅推动了大数据的发展，而且有利于推动政府治理的创新，而数据可视化的应用可以让政府实现科学决策和高效治理。通过应用数据可视化，政府能够借助数据在短时间内制定及时、高效、准确的治理手段和决策管理，进行多方位布局和监管。不仅如此，数据可视化还可以帮助政府预测社会问题，制定相关的发展政策。

2）金融业

在当今互联网金融激烈的竞争下，市场形势瞬息万变，金融行业面临诸多挑战。通过引入数据可视化可以对企业各地的日常业务动态实时掌控，对客户数量和借贷金额等数据进行有效监管，帮助企业实现数据实时监控，加强对市场的监督和管理；通过对核心数据多维度地分析和对比，指导公司科学调整运营策略，制定发展方向，不断提高公司风控管理能力和竞争力。

3）医学

数据可视化可以帮助医院把之前分散、凌乱的数据加以整合，构建全新的医疗管理体系模型，帮助医院领导快速解决关注的问题，例如一些门诊数据、用药数据、疾病数据等。此外，大数据可视化还可以应用于诊断医学及一些外科手术中的精确建模，通过三维图像的建立可以帮助医生确定是否进行外科手术或者进行何种治疗。不仅如此，数据可视化还可以加快临床上对疾病预防、流行疾病防控等疾病的预测和分析能力。

4）工业生产

数据可视化在工业生产中有着重要的应用，例如可视化智能硬件的生产与使用。可视化智能硬件通过软/硬件结合的方式让设备拥有智能化的功能，并对硬件采集的数据进行可视化的呈现。因此在智能化之后，硬件就具备了大数据等附加价值。随着可视化技术的不断发展，今后智能硬件从可穿戴设备延伸到智能电视、智能家居、智能汽车、医疗健康、智能玩具、智能机器人、智能交通、智能教育等各个不同的领域。

5）电子商务

大数据可视化技术在电子商务中有着极其重要的作用。对于电商企业而言，针对商品展开数字化的分析运营是企业的日常必要工作。通过可视化的展示可以为企业销售策略的实施提供可靠的保证。现如今采用数据可视化方法进行营销可以帮助电商企业跨数据源整合数据，极大地提高了数据分析能力。通过快速进行数据整合，成功定位忠诚度高的顾客，从而制定精准化营销策略；通过挖掘数据，预测分析客户的购物习惯，获悉市场变化，提高竞争力，打造电商航母。例如，在商业模式中可建立消费者个性偏好与调查邮件之间的可视化数据表示。

图 5-9 所示的是大数据可视化在城市公共卫生服务平台上的应用，在该平台上通过引入数据可视化将枯燥的数字转换为生动而引人注意的图像，从而为政府决策起到了积极的引导作用。

6. 数据可视化面临的挑战

随着大数据技术的日益成熟，数据可视化也得到了迅猛的发展。但与此同时，数据可视化存在着许多问题，面临着巨大的挑战。数据可视化面临的挑战主要指可视化分析过程中数据的呈现方式，包括可视化技术和信息可视化显示。目前，在数据可视化研究中，高清晰显示、大屏幕显示、高可扩展数据投影、维度降解等技术都试着从不同角度解决这个难题。

此外，可感知的交互的扩展性也是大数据可视化面临的挑战之一，如可视化每个数据点都可能导致过度绘制而降低用户的辨识能力，通过抽样或过滤数据可以删去离群值。

图 5-9　数据可视化的应用

从大规模数据库中查询数据可能导致高延迟，使交互率降低。由于当前大多数大数据可视化工具在扩展性、功能和响应时间上的表现非常糟糕，所以大规模数据和高维度数据会使数据可视化变得困难，从而带来数据分析中的不确定性。

因此，对大数据可视化的实施应当遵循以下几点。

（1）正确认识数据可视化的意义。要重视数据可视化的作用，但也不可太依赖数据可视化。在对数据的使用中，并不是所有的数据都需要用可视化的方法来表达它的消息。在实际应用中，应以使用者的需求为第一要素，而不是盲目地进行数据可视化。

（2）重视数据的质量。数据可视化呈现的数据应当是干净的、真实的，因此通过数据治理或信息管理确保干净的数据十分必要。要遵从数据可视化的设计原则，并确保数据来源的真实性和合理性。

（3）改善数据可视化的硬件条件。数据可视化对硬件平台要求较高，因此在实施中应极力改善硬件条件，如可以尝试增加内存和提高并行处理的能力。此外，在构建大数据平台时应当选择合适的架构，以便数据可视化的实现。

5.1.2　数据可视化工具

目前常用的数据可视化工具较多，本节列出了一些使用较频繁的数据可视化工具。

（1）Excel。Excel 作为一种简单、方便、覆盖面广的 Office 软件，无疑是数据可视化工具的典型。作为数据可视化的入门工具，Excel 是快速分析数据的理想工具，也能创建供企业内部使用的数据图。例如，微软公司曾经发布了一款名为 GeoFlow 的插件，它是结合 Excel 和 Bing 地图所开发出来的 3D 数据可视化工具。但是 Excel 在颜色、线条和样式上选择的范围有限，因此用 Excel 很难制作出符合专业出版物和网站需要的数据图。

(2) D3.js。D3(数据驱动文件)是一种支持 SVG 渲染的 JavaScript 库,也是目前最受欢迎的可视化数据库之一。它不仅可以创建简单的条形图和折线图,还可以完成更复杂的 Voronoi 图、树图、圆形集图和字符云。D3.js 允许绑定任意数据到 DOM,然后将数据驱动转换应用到 Document 中。用户可以使用它用一个数组创建基本的 HTML 表格,或是利用它的流体过渡和交互,用相似的数据创建惊人的 SVG 条形图。

(3) Flot。Flot 是一个基于 jQuery 的开源 JavaScript 库,它是一个优秀的线图和条形图创建工具,可以运用于支持 Canvas 的所有浏览器。用户通过使用 Flot 可以很轻松地对图像进行回调、风格和行为操作。比起其他制图工具,Flot 能够给予用户更多的灵活空间。

(4) Google Chart API。Google Chart API 提供了一种非常完美的方式来可视化数据,提供了大量现成的图表类型,从简单的线图表到复杂的分层树地图等。此外,它还内置了动画和用户交互控制。值得注意的是,这些图像都是在客户端上生成的,如果设备不支持 JavaScript、非联网状态使用或者用不同格式保存,都会引发问题。

(5) Visual.ly。Visual.ly 是一个全新的可视化信息图形平台,它为用户提供即时数据可视化的功能,并提供了大量信息图模板,被誉为"信息图设计师的在线集市"。使用 Visual.ly 可以制作各种信息图,而不仅仅是数据可视化,因此它受到广大数据分析师和媒体从业人员的青睐。

(6) Modest Maps。Modest Maps 是一个可视化的数据地图工具,大小只有 10KB 左右,是目前最小的可用地图库。它用于将数据与地理信息叠加生产出数据地图,是近年来数据新闻生产的一种重要方式。与此同时,Google Maps 的出现完全颠覆了过去人们对在线地图功能的认识,让人们生成数据可视化地图变得简单。

(7) Processing。Processing 是数据可视化的招牌工具,同时也是交互式可视化处理的模范工具。Processing 能够让程序员使用更简单的代码,再循序编译成 Java。因此 Processing 可以在几乎所有平台上运行。

(8) CartoDB。CartoDB 是一个在 Web 中用来存储和虚拟化地理数据的工具,它可以很轻易地把表格数据和地图关联起来,以便于浏览者开发互动地图。例如,浏览者可以输入 CSV 通信地址文件,CartoDB 能将地址字符串自动转化成经度/维度数据并在地图上标记出来。目前 CartoDB 支持免费生成 5 张地图数据表,更多使用需要支付月费。

(9) R 语言。R 语言是目前非常流行、免费且开源的统计分析软件,是统计学家、数据科学家和业务分析师使用最广泛的一种编程语言,同时也是非常复杂的语言。它主要用于分析大型数据集的统计数据包,并拥有强大的社区和库。一般而言,开发者需要花一定的时间才能完全掌握它。

(10) Weka。Weka 是一款免费的、非商业化的、基于 Java 环境下开源的机器学习及数据挖掘软件,同时也是数据分析师喜爱的工具。它是一款能根据属性分类和集群大量数据的优秀工具,并且还能生成一些简单的图表,用于数据可视化中。

(11) iCharts。iCharts 提供了一个用于创建并呈现引人注目的图表的托管解决方案。有许多不同种类的图表可供选择,每种类型都完全可定制,以适合网站的主题。iCharts 有交互元素,可以从 Google Doc、Excel 表单和其他来源中获取数据。

5.1.3 数据可视化图表

1. 图表的分类

按照数据的作用和功能可以把图表分为比较类、分布类、流程类、地图类、占比类、区间类、关联类、时间类和趋势类等，其中在每一种类型的图表中都可包含不同的数据可视化图形，例如柱状图、K线图、散点图、气泡图、热力图、饼图、折线图、面积图、趋势图、直方图、雷达图、色块图、漏斗图、和弦图、仪表盘、环图和词云等。

2. 数据可视化图表介绍

1）柱状图

基础柱状图，使用垂直或水平的柱子显示类别之间的数值比较。其中一个轴表示需要对比的分类维度，另一个轴代表相应的数值。柱状图又可分为纵向柱状图和横向柱状图（条形图）。图 5-10 所示的是柱形图。

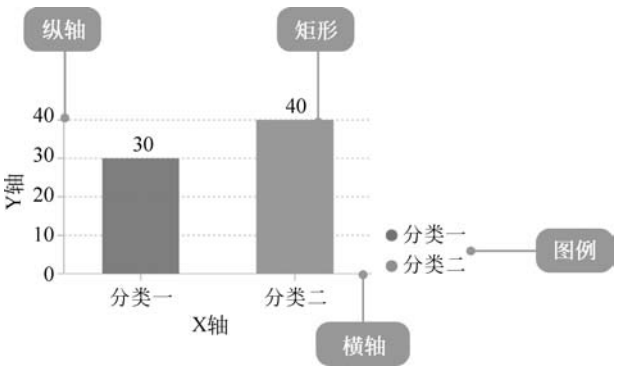

图 5-10 柱状图

2）K线图

K线图又称阴阳图、棒线、红黑线或蜡烛线，常用于展示股票交易数据。K线就是指将各种股票每日、每周、每月的开盘价、收盘价、最高价、最低价等涨跌变化状况用图形的方式表现出来。图 5-11 所示的是K线图。

图 5-11 K线图

3）散点图

散点图是指在回归分析中数据点在直角坐标系平面上的分布图，散点图表示因变量随自变量变化的大致趋势，据此可以选择合适的函数对数据点进行拟合。图 5-12 所示的是散点图。

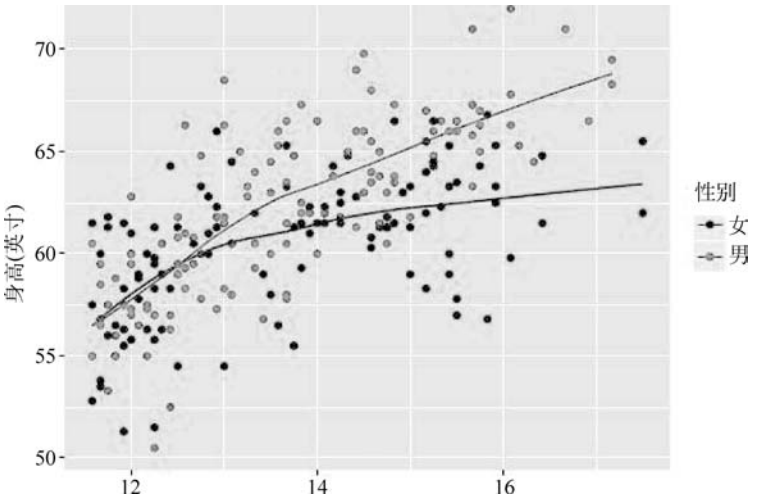

图 5-12　散点图

4）气泡图

气泡图是一种多变量图表，是散点图的变体，也可以认为是散点图和百分比区域图的组合。图 5-13 所示的是气泡图。

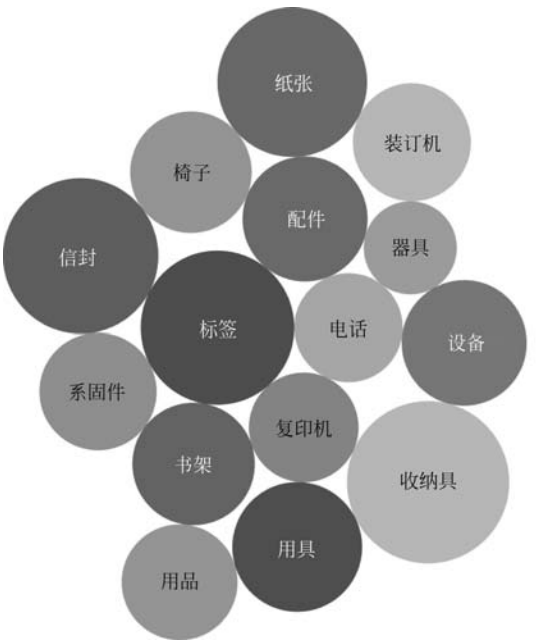

图 5-13　气泡图

5）热力图

热力图是以特殊高亮的形式显示访客热衷的页面区域和访客所在的地理区域的图示。热力图可以显示不可点击区域发生的事情。图 5-14 所示的是热力图。

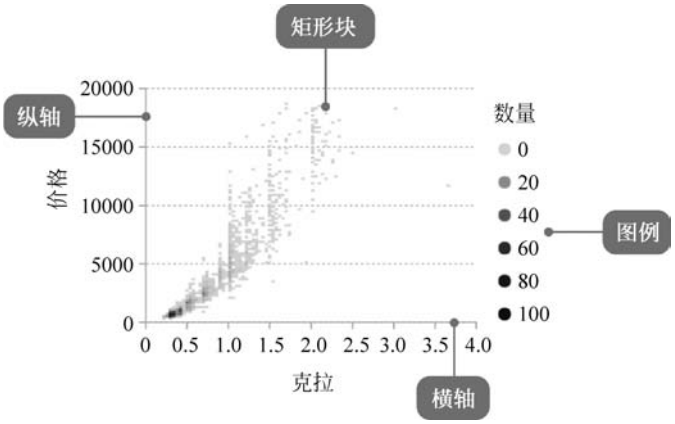

图 5-14　热力图

6）饼图

饼图用于表示不同分类的占比情况，通过弧度大小来对比各种分类。饼图通过将一个圆饼按照分类的占比划分成多个区块，整个圆饼代表数据的总量，每个区块（圆弧）表示该分类占总体的比例大小，所有区块（扇区）的总和等于 100%。图 5-15 所示的是饼图。

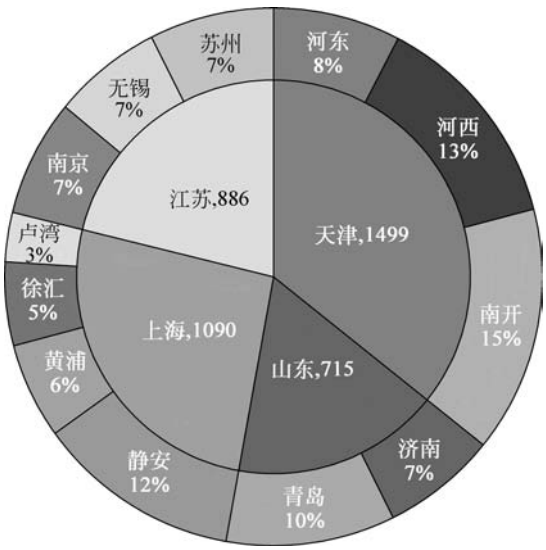

图 5-15　饼图

7）折线图

折线图用于显示数据在一个连续的时间间隔或者时间跨度上的变化，它的特点是反映事物随时间或有序类别变化的趋势。在折线图中，数据是递增还是递减、增减的速率、

增减的规律(周期性、螺旋性等)、峰值等特征都可以清晰地反映出来。图 5-16 所示的是折线图。

8) 面积图

面积图又称为区域图,它是在折线图的基础上形成的,它将折线图中折线与自变量坐标轴之间的区域使用颜色或纹理填充,这样一个填充区域称为面积,颜色的填充可以更好地突出趋势信息。图 5-17 所示的是面积图。

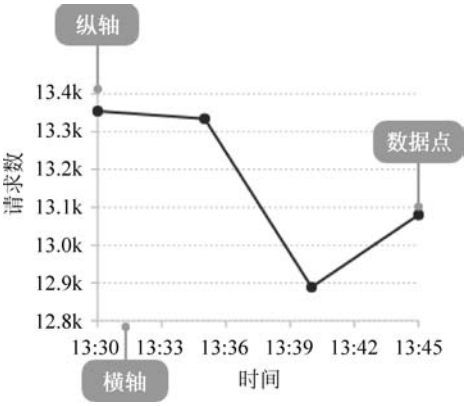

图 5-16　折线图

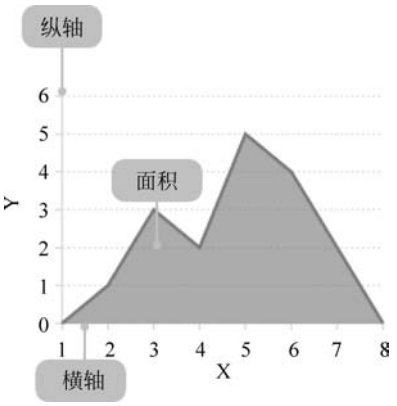

图 5-17　面积图

9) 漏斗图

漏斗图适用于业务流程比较规范、周期长、环节多的单流程单向分析,通过漏斗各环节业务数据的比较能够直观地发现和说明问题所在的环节,进而做出决策。漏斗图从上到下有逻辑上的顺序关系,表现了随着业务流程的推进业务目标完成的情况。图 5-18 所示的是漏斗图。

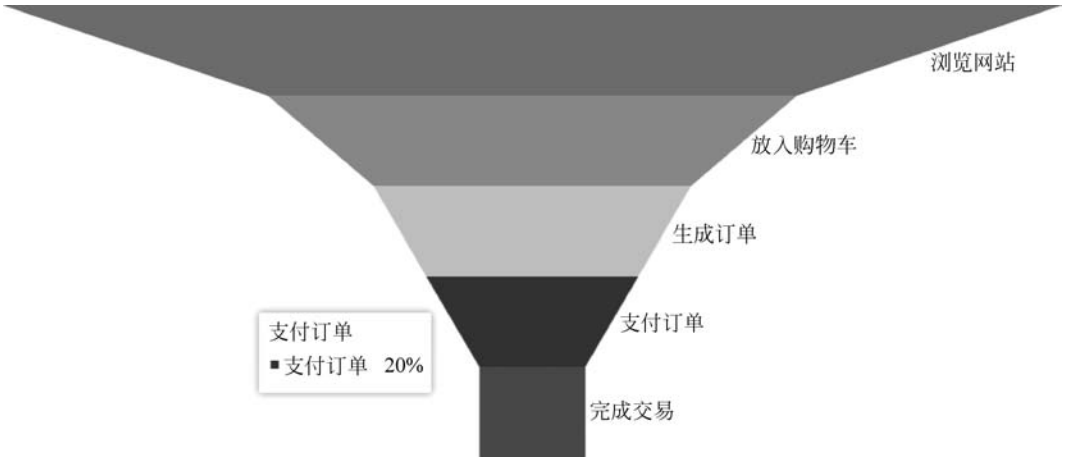

图 5-18　漏斗图

10) 雷达图

雷达图又称为戴布拉图、蜘蛛网图,传统的雷达图被认为是一种表现多维(四维以上)数据的图表。它将多个维度的数据量映射到坐标轴上,这些坐标轴起始于同一个圆心点,通常

结束于圆周边缘，将同一组的点使用线连接起来就成了雷达图。图 5-19 所示的是雷达图。

11）环图

环图又称为甜甜圈图，其本质是饼图将中间区域挖空。环图相对于饼图空间的利用率更高，可以使用它的空心区域显示文本信息，例如标题等。图 5-20 所示的是环图。

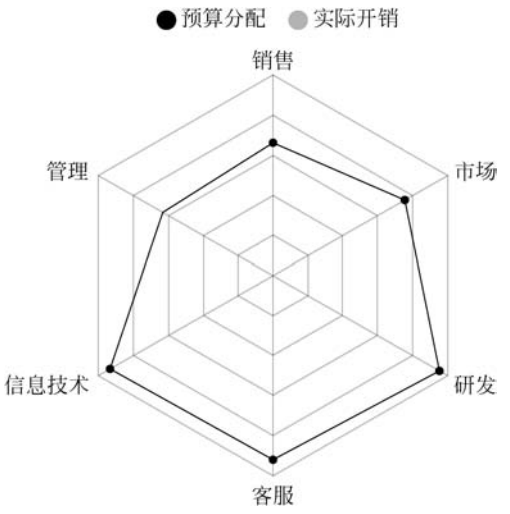

图 5-19　雷达图

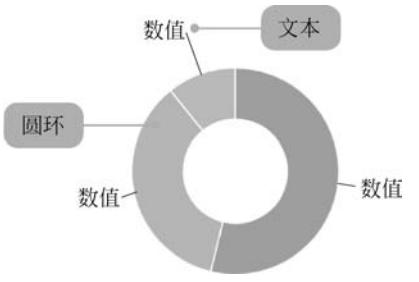

图 5-20　环图

12）直方图

直方图的形状类似于柱状图却有着与柱状图完全不同的含义。直方图涉及统计学的概念，首先要对数据进行分组，然后统计每个分组内数据元的数量。图 5-21 所示的是直方图。

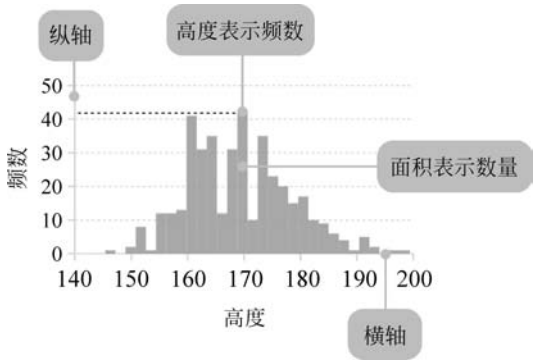

图 5-21　直方图

13）仪表盘

仪表盘是一种拟物化的图表，刻度表示量度，指针表示维度，指针角度表示数值。仪表盘图表就像汽车的速度表一样，有一个圆形的表盘及相应的刻度，有一个指针指向当前数值。目前很多的管理报表或报告上都使用这种图表，以直观地表现出某个指标的进度或实际情况。图 5-22 所示的是仪表盘。

14）词云

词云又称为文字云，由词汇组成类似云的彩色图形。词云一般用于展示大量文本数据。图 5-23 所示的是词云。

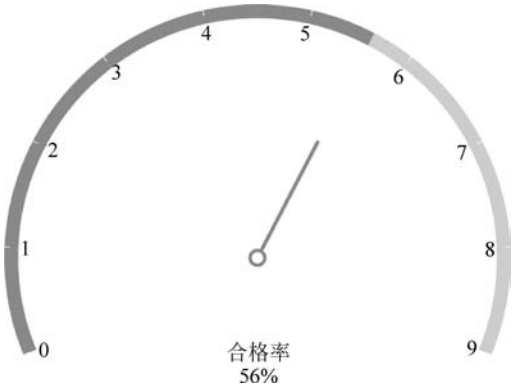

图 5-22　仪表盘

图 5-23　词云

5.2　matplotlib 可视化基础

视频讲解

5.2.1　numpy 库

1. numpy 库简介

numpy 是 Python 进行数据处理的底层库，是高性能科学计算和数据分析的基础，如著名的 Python 机器学习库 SKlearn 就需要 numpy 的支持。掌握 numpy 的基础数据处理能力是利用 Python 做数据运算及机器学习的基础。

同时，在数据可视化中也常需要用到 numpy 中的数组存储及矩阵运算等功能，因此掌握 numpy 库对学好数据可视化十分有帮助。

numpy 库具有以下特征。

（1）numpy 库中最核心的部分是 ndarray 对象。它封装了同构数据类型的 n 维数组，它的功能将通过演示代码的形式呈现。

（2）在数组中所有元素的类型必须一致，且在内存中占有相同的大小。

（3）数组元素可以使用索引来描述，索引的序号从 0 开始。

（4）numpy 数组的维数称为秩（rank），一维数组的秩为 1，二维数组的秩为 2，以此类推。在 numpy 中，每一个线性的数组称为一个轴（axes），秩其实是描述轴的数量。

值得注意的是，numpy 数组和标准 Python 序列之间有几个重要区别。

（1）numpy 数组在创建时就会有一个固定的尺寸，这一点和 Python 中的 list 数据类型是不同的。

（2）在数据量较大时，使用 numpy 进行高级数据运算和其他类型的操作是更为方便的。通常情况下，这样的操作比使用 Python 的内置序列更有效，执行代码更少。

2. numpy 库的安装与测试

本节以 Windows 7 为例，讲解 Python 3.7 中 numpy 库的安装过程。在 Windows 中进入 cmd 命令后，直接运行 pip install numpy 即可安装，安装完成后输入“import numpy”，如果没报错则表示成功，如图 5-24 所示。

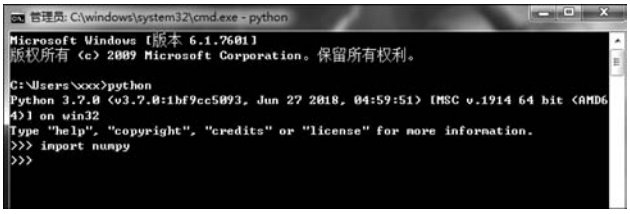

图 5-24　numpy 库成功安装

在实际运行中，建议在引用 numpy 库时输入以下代码。

```
import numpy as np
```

将 numpy 用 np 代替，以提高 Python 中代码的可读性和可用性。

3. numpy 库的使用

1）numpy 库数组的创建

在 numpy 库中创建数组可以使用如下语法。

```
numpy.array
```

该语句表示通过引入 numpy 库创建了一个 ndarray 对象。

【例 5-1】 创建数组对象。

代码如下。

```
import numpy as np
a = np.array([1,2,3])
print(a)
```

该例首先引入了 numpy 库，接着定义了一个一维数组 a，最后将数组输出显示。运行该程序，结果如图 5-25 所示。

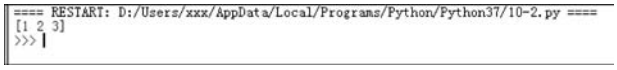

图 5-25　数组的定义与显示

2）numpy 数组的参数

在创建数组时可以加入以下参数。

```
numpy.array(object, dtype = None, copy = True, order = None, subok = False, ndmin = 0)
```

参数的含义如表 5-1 所示。

表 5-1　创建数组的参数

参 数 名 称	参 数 含 义
object	任何暴露数组接口方法的对象都会返回一个数组或任何(嵌套)序列
dtype	数组的所需数据类型，可选
copy	可选，默认为 True，对象是否被复制
order	C(按行)、F(按列)或 A(任意，默认)
subok	在默认情况下，返回的数组被强制为基类数组。如果为 True，则返回子类
ndmin	指定返回数组的最小维数

【例 5-2】 创建一个多维数组对象。

代码如下。

```
import numpy as np
a = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a)
```

该例定义并显示了一个多维数组，运行结果如图 5-26 所示。

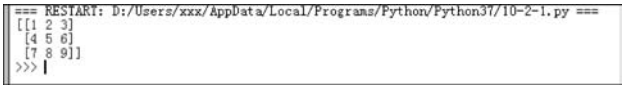

图 5-26　多维数组的定义与显示

【例 5-3】 显示多维数组的数据类型。

代码如下。

```
import numpy as np
a = np.array([[1,2,3],[4,5,6],[7,8,9]],dtype = complex)
print(a)
```

该例定义了一个多维数组，并显示其数据类型，运行结果如图 5-27 所示。

```
=== RESTART: D:/Users/xxx/AppData/Local/Programs/Python/Python37/10-2-2.py ===
[[1.+0.j 2.+0.j 3.+0.j]
 [4.+0.j 5.+0.j 6.+0.j]
 [7.+0.j 8.+0.j 9.+0.j]]
>>>
```

图 5-27　多维数组的数据类型的显示

在该例中 complex 类型由实部和虚部组成。表 5-2 显示了 numpy 中的常见数据类型。

表 5-2　numpy 中的常见数据类型

数 据 类 型	含　　义
bool	布尔类型
int	默认整数
int8	有符号的 8 位整型
int16	有符号的 16 位整型
int32	有符号的 32 位整型
int64	有符号的 64 位整型
uint8	无符号的 8 位整型
uint16	无符号的 16 位整型
uint32	无符号的 32 位整型
uint64	无符号的 64 位整型
float16	半精度浮点数
float32	单精度浮点数
float64	双精度浮点数
string	字符串类型
complex64	复数，分别用两个 32 位浮点数表示实部和虚部
complex128	复数，分别用两个 64 位浮点数表示实部和虚部

3）ndarray 对象的基本属性

在创建了一个数组以后，可以查看 ndarray 对象的基本属性，如表 5-3 所示。

表 5-3　ndarray 对象的基本属性

属性名称	属　性　值	属性名称	属　性　值
shape	数组中各维度的尺度	itemsize	数组中每个元素的字节大小
reshape	调整数组大小	nbetys	整个数组所占的存储空间
size	数组元素的总个数	flages	返回数组的当前值
data	数组中的元素在内存中所占的字节数		

【例 5-4】　显示多维数组的维度。

代码如下。

```
import numpy as np
a = np.array([[1,2,3],[4,5,6],[7,8,9]])
print(a.shape)
```

该例定义了一个多维数组，并显示其维度，运行结果如图 5-28 所示。

```
=== RESTART: D:/Users/xxx/AppData/Local/Programs/Python/Python37/10-2-3.py ===
(3, 3)
>>>
```

图 5-28　多维数组的维度显示

【例 5-5】 显示数组中每个元素的字节大小。

代码如下。

```
import numpy as np
a = np.array([1,2,3,4,5,6,7,8,9], dtype = np.int8)
print(a.itemsize)
```

该例定义了一个数组，并显示其元素的字节大小，运行结果如图 5-29 所示。

```
=== RESTART: D:/Users/xxx/AppData/Local/Programs/Python/Python37/10-2-4.py ===
1
>>>
```

图 5-29　数组元素的字节大小显示

4）ndarray 对象的切片和索引

ndarray 对象的内容可以通过索引或切片来访问和修改，ndarray 对象一般可由 arange()函数创建。代码为：

```
a = np.arange()
```

如果仅提取数组对象的一部分，则可以使用 slice()函数来构造，例如：

```
s = slice()
```

【例 5-6】 ndarray 对象的切片。

代码如下。

```
import numpy as np
a = np.arange(10)
s = slice(1,8,2)
print(a[s])
```

该例首先定义了一个数组，该数组对象由 arange()函数创建。然后分别用起始、终止和步长值 1、8 和 2 定义切片对象。当这个切片对象传递给 ndarray 时会对它的一部分进行切片，从索引 1 到 8，步长为 2。该例的运行结果如图 5-30 所示。

```
=== RESTART: D:/Users/xxx/AppData/Local/Programs/Python/Python37/10-2-5.py ===
[1 3 5 7]
>>>
```

图 5-30　ndarray 对象的切片

5）ndarray 对象的线性代数与三角函数

numpy 包含 numpy.linalg 模块，提供线性代数所需的所有功能。此模块中的一些重要功能如表 5-4 所示。此外，在 numpy 库中还可以计算三角函数，如表 5-5 所示。

表 5-4　numpy 中的线性代数模块

模块名称	模块功能	模块名称	模块功能
dot	计算两个数组的点积	determinant	计算数组的行列式
vdot	计算两个向量的点积	solve	计算线性矩阵方程
inner	计算两个数组的内积	inv	计算矩阵的乘法逆矩阵
matmul	计算两个数组的矩阵积		

表 5-5　numpy 中常见的三角函数

函数名称	函数作用	函数名称	函数作用
sin(x[, out])	正弦值	arcsin(x[, out])	反正弦
cos(x[, out])	余弦值	arccos(x[, out])	反余弦
tan(x[, out])	正切值	arctan(x[, out])	反正切

【例 5-7】 计算两个数组的点积，对于一维数组，它是向量的内积，对于二维数组，其等效于矩阵乘法。

代码如下。

```
import numpy.matlib
import numpy as np
a = np.array([[1,2],[3,4]])
b = np.array([[10,20],[30,40]])
np.dot(a,b)
print(np.dot(a,b))
```

其中 matlib 表示 numpy 中的矩阵库。该段程序定义了两个数组 a 和 b，并计算这两个数组的点积。其中点积的计算公式为：

$$[[1*10+2*30,1*20+2*40],[3*10+4*30,3*20+4*40]]$$

运行该程序，如图 5-31 所示。

```
=== RESTART: D:/Users/xxx/AppData/Local/Programs/Python/Python37/10.3-1.py ===
[[ 70 100]
 [150 220]]
>>>
```

图 5-31　numpy 中的线性代数

视频讲解

5.2.2　matplotlib 的认识与安装

matplotlib 是一个 Python 的 2D 绘图库，它以各种硬拷贝格式和跨平台的交互式环境生成出版质量级别的图形，在使用 matplotlib 之前，首先要将其安装在系统中。本节以 Windows 7 为例介绍其安装过程。

1）安装 Visual Studio

访问 http://dev.windows.com，单击 downloads，再查找一组 Visual Studio Community 免费的 Windows 开发工具。下载并运行该安装程序。

2）下载 matplotlib 安装程序

访问 https://pypi.python.org/pypi/matplotlib，查找与 Python 版本相配的 wheel 文件（扩展名为.whl）。

3）使用 pip 来安装 matplotlib

在 Windows 中进入 cmd 后，直接输入“python -m pip install --user matplotlib-2.1.0-cp36-cp36m-win32.whl”命令来执行 matplotlib 程序的安装。

4）安装完成

安装完成后，输入“pip list”命令查看已安装的 Python 库，如图 5-32 和图 5-33 所示。

```
C:\Users\xxx>pip list
Package                       Version
----------------------------- ---------
```

图 5-32　查看 pip 已安装的库

```
matplotlib                    2.2.3
```

图 5-33　显示已安装的 matplotlib 库

5.2.3　matplotlib 测试

1. matplotlib 库的测试

在安装完 matplotlib 库后，可在 Python 环境中测试。输入以下代码，如果不报错，则表示 matplotlib 库安装成功。

```
import matplotlib
import matplotlib.pyplot as plt
```

如图 5-34 所示，在 Python 环境中已正确安装了 matplotlib 库。

```
C:\Users\xxx>python
Python 3.7.0 (v3.7.0:1bf9cc5093, Jun 27 2018, 04:59:51) [MSC v.1914 64 bit (AMD6
4)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import matplotlib
>>> import matplotlib.pyplot as plt
>>>
```

图 5-34　在 Python 环境中已正确安装了 matplotlib 库

2. matplotlib 库的运行

【例 5-8】 在 Python 中输入代码，测试生成的 matplotlib 图形。

代码如下。

```
import matplotlib.pyplot as plt
plt.plot([1,2,3])
plt.ylabel('some numbers')
plt.show()
```

该程序绘制了一条直线，执行该程序，结果如图 5-35 所示。

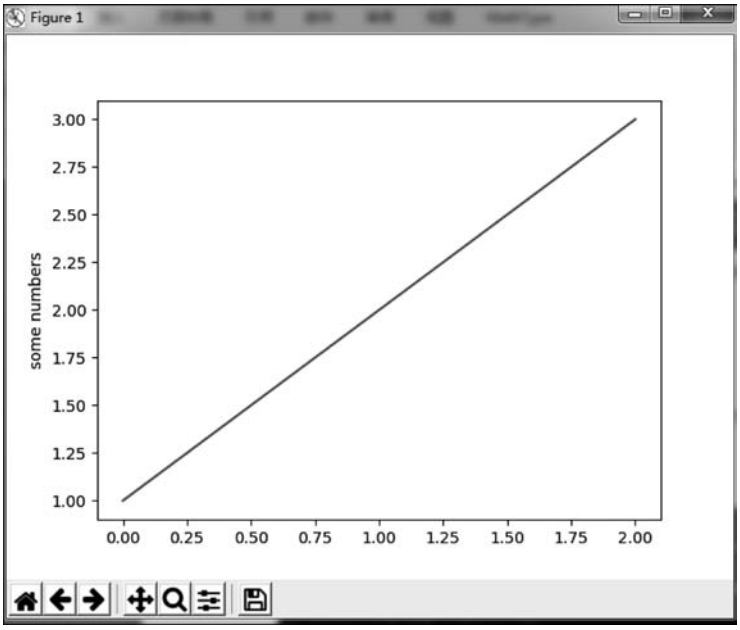

图 5-35　matplotlib 图形

5.2.4　matplotlib.pyplot 库

1. matplotlib.pyplot 库简介

matplotlib 是 Python 下著名的绘图库，为了方便快速绘图，matplotlib 通过 pyplot 模块提供了一套和 Matlab 类似的绘图 API，将众多绘图对象所构成的复杂结构隐藏在这套 API 内部。用户只需要调用 pyplot 模块所提供的函数就可以实现快速绘图以及设置图表的各种细节。matplotlib.pyplot 的引用方式如下。

```
import matplotlib.pyplot as plt
```

通过以上代码将 pyplot 模块重命名为 plt，有助于提高代码的可读性。简单地说，在后续的程序中，plt 代替了 matplotlib.pyplot。

2. matplotlib.pyplot 函数库简介

matplotlib.pyplot 是一个命令型函数集合，它可以让人们像使用 Matlab 一样使用 matplotlib。pyplot 中的每一个函数都会对画布图像做出相应的改变，例如创建画布、在

画布中创建一个绘图区、在绘图区上画几条线、给图像添加文字说明等。

1）plt.figure()

使用 plt.figure()函数创建一个全局绘图区域，其中可包含如下参数。

- num：设置图像编号。
- figsize：设置图像的宽度和高度，单位为英寸。
- facecolor：设置图像的背景颜色。
- dpi：设置绘图对象的分辨率。
- edgecolor：设置图像的边框颜色。

在创建了图像区域之后，再用 plt.show()函数显示。例如显示绘图区域的代码如下。

```
plt.figure(figsize=(6,4))
plt.show()
```

2）plt.subplot()

subplot()用于在全局绘图区域中创建自绘图区域，其中可包含如下参数。

- nrows：subplot 的行数。
- ncols：subplot 的列数。

使用 subplot 可以将 figure 划分为 n 个子图，但每条 subplot 命令只会创建一个子图。

【例 5-9】 用 subplot 划分子区域。

代码如下。

```
import matplotlib.pyplot as plt
plt.subplot(333)
plt.show()
```

该例使用语句 plt.subplot(333)将全局划分为了 3×3 的区域，其中横向为 3，纵向也为 3，并在第 3 个位置（右上方）生成了一个坐标系。运行该程序，结果如图 5-36 所示。

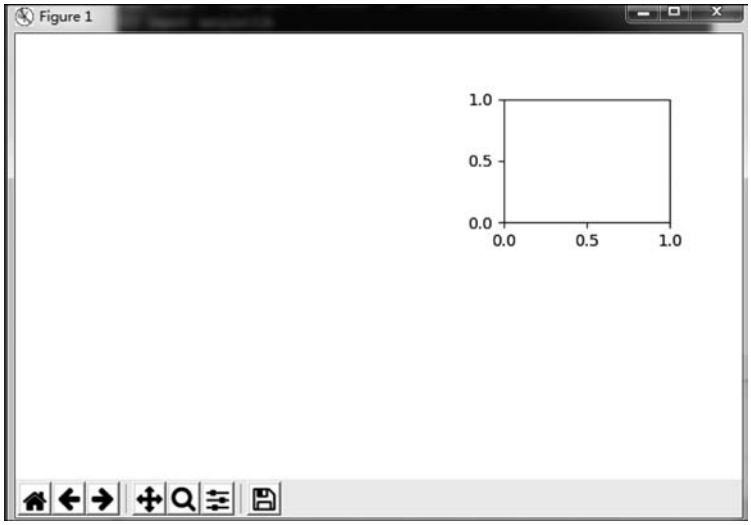

图 5-36　subplot 划分子区域

3）plt.axes()

plt.axes(rect,axisbg＝'w')创建一个坐标系风格的子绘图区域。默认创建一个subplot(111)坐标系，参数 rect＝[left,bottom,width,height]中 4 个变量的范围都是[0,1]，表示坐标系与全局绘图区域的关系；axisbg 表示背景色，默认为白色'white'。代码如下。

```
import matplotlib.pyplot as plt
plt.axes([0.1,0.1,0.7,0.3],axisbg = 'y')
plt.show()
```

4）plt.subplots_adjust()

plt.subplots_adjust()用于调整子绘图区域的布局。

3. matplotlib.pyplot 相关函数简介

plt 子库提供了 7 个用于读取和显示的函数、17 个用于绘制基础图表的函数、3 个区域填充函数、9 个坐标轴设置函数以及 11 个标签与文本设置函数，具体如表 5-6～表 5-10 所示。

表 5-6　plt 库的读取和显示函数

函 数 名 称	函数的作用	函 数 名 称	函数的作用
plt.legend()	在绘图区域放置绘图标签	plt.imsave()	保存数组为图像文件
plt.show()	显示绘制的图像	plt.savefig()	设置图像保存的格式
plt.matshow()	在窗口显示数组矩阵	plt.imread()	从图像文件中读取数组
plt.imshow()	在 axes 上显示图像		

表 5-7　plt 库的基础图表函数

函 数 名 称	函数的作用
plt.plot(x,y,label,color,width)	根据 x、y 数组绘制直线、曲线
plt.boxplot(data,notch,position)	绘制一个箱形图
plt.bar(left,height,width,bottom)	绘制一个条形图
plt.barh(bottom,width,height,left)	绘制一个横向条形图
plt.polar(theta,r)	绘制极坐标图
plt.pie(data,explode)	绘制饼图
plt.psd(x, NFFT＝256, pad_to, Fs)	绘制功率谱密度图
plt.specgram(x, NFFT＝256, pad_to, F)	绘制谱图
plt.cohere(x,y,NFFT＝256,Fs)	绘制 x－y 的相关性函数
plt.scatter()	绘制散点图
plt.step(x,y,where)	绘制步阶图
plt.hist(x,bins,normed)	绘制直方图
plt.contour(X,Y,Z,N)	绘制等值线
plt.clines()	绘制垂直线
plt.stem(x,y,linefmt, markerfmt, basefmt)	绘制曲线上每个点到水平轴线的垂线
plt.plot_date()	绘制日期数据
plt.plotfile()	绘制数据后写入文件

表 5-8　区域填充函数

函 数 名 称	函数的作用
fill(x,y,c,color)	填充多边形
fill_between(x,y1,y2,where,color)	填充曲线围成的多边形
fill_betweenx(y,x1,x2,where,hold)	填充水平线之间的区域

表 5-9　坐标轴设置函数

函 数 名 称	函数的作用	函 数 名 称	函数的作用
plt. axis()	获取设置轴属性的快捷方式	plt. autoscale()	自动缩放轴视图
plt. xlim()	设置 X 轴的取值范围	plt. text()	为 axes 图添加注释
plt. ylim()	设置 Y 轴的取值范围	plt. thetagrids()	设置极坐标网格
plt. xscale()	设置 X 轴的缩放	plt. grid()	打开或关闭极坐标
plt. yscale()	设置 Y 轴的缩放		

表 5-10　标签与文本设置函数

函 数 名 称	函数的作用	函 数 名 称	函数的作用
plt. figlegend()	为全局绘图区域放置图注	plt. get_figlabels()	返回当前绘图区域的标签列表
plt. xlabel()	设置当前 X 轴的文字	plt. figtext()	为全局绘图区域添加文本信息
plt. ylabel()	设置当前 Y 轴的文字	plt. title()	设置标题
plt. xticks()	设置当前 X 轴刻度位置的文字和值	plt. suptitle()	设置总图标题
plt. yticks()	设置当前 Y 轴刻度位置的文字和值	plt. annotate()	为文本添加注释
plt. clabel()	设置等高线数据		

4. numpy 和 matplotlib 绘图综合应用

【例 5-10】 用 numpy 库和 matplotlib 库绘制图形。

代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(10)
y = np.sin(x)
z = np.cos(x)
plt.plot(x, y, marker = " * ", linewidth = 3, linestyle = " -- ", color = "red")
#marker 设置数据点样式,linewidth 设置线宽,linestyle 设置线型样式,color 设置颜色
plt.plot(x, z)
plt.title("matplotlib")
plt.xlabel("x")
plt.ylabel("y")
plt.legend(["Y","Z"], loc = "upper right")        #设置图例
plt.grid(True)
plt.show()
```

该例绘制了两条折线，运行结果如图 5-37 所示。

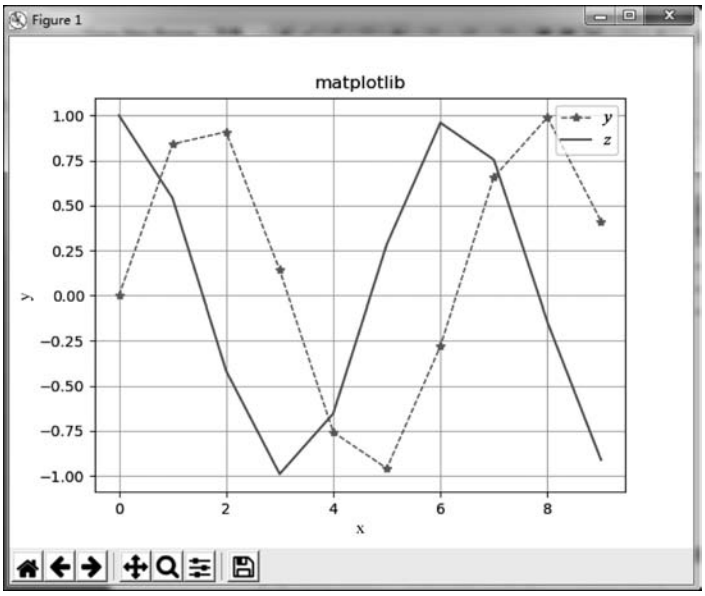

图 5-37　使用 numpy 库和 matplotlib 库绘制图形

值得注意的是，该例使用 numpy 库存储数组，使用 matplotlib 库将数组用图形输出到屏幕上，最终显示为两条颜色不同的折线 y 和 z，分别代表数学公式中的正弦函数 sin(x)和余弦函数 cos(x)。

如果想在此图中显示其他三角函数折线，例如 tan(x)，可加上如下代码。

```
w = np.tan(x)
plt.plot(x, w)
```

5.3　matplotlib 可视化绘图


视频讲解

5.3.1　绘制线性图形

使用 matplotlib 库可以绘制各种图形，其中最基本的图形是线性图形，主要由线条组成。

【例 5-11】　用 matplotlib 库绘制线性图形。

代码如下。

```
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
font_set = FontProperties(fname = r"c:\windows\fonts\simsun.ttc", size = 20) # 导入宋体字
                                                                              # 体文件
dataX = [1,2,3,4]
dataY = [2,4,4,2]
```

```
plt.plot(dataX,dataY)
plt.title("绘制直线",FontProperties = font_set);
plt.xlabel("x 轴",FontProperties = font_set);
plt.ylabel("y 轴",FontProperties = font_set);
plt.show()
```

该例绘制了一条直线，直线的形状由坐标值 x 和 y 决定，并引用了计算机中的中文字体来显示该图形的标题。运行该程序，结果如图 5-38 所示。

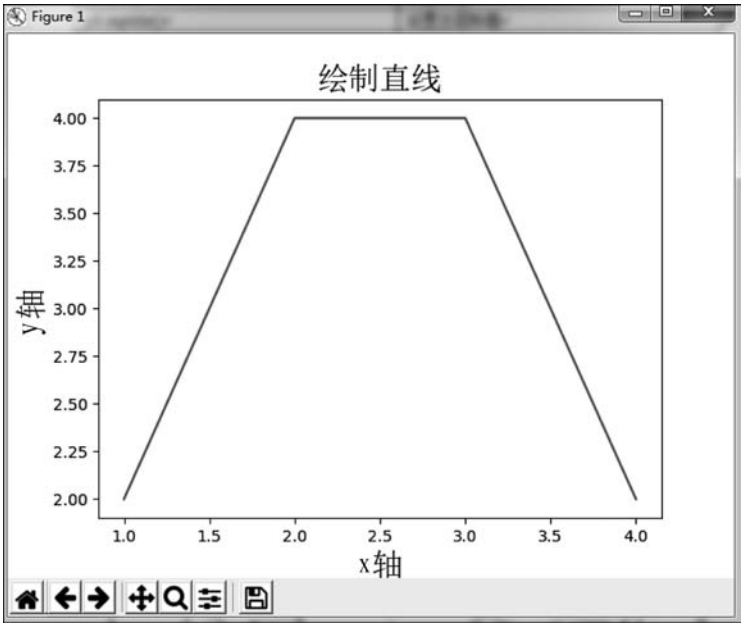

图 5-38　绘制直线

从图 5-38 可以看出，当 x 取值为 1 时，y 取值为 2；当 x 取值为 2 时，y 取值为 4。因此，最终在屏幕上显示一条未封闭的直线段。如果在 dataX 和 dataY 中设置多个参数，则可以显示其他的线性图形。

5.3.2　绘制柱状图

柱状图也称为条形图，是一种以长方形的长度为变量的表达图形的统计报告图，由一系列高度不等的纵向条纹表示数据分布的情况，用来比较两个或两个以上的数值。

【例 5-12】　用 matplotlib 库绘制柱状图。

代码如下。

```
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
font_set = FontProperties(fname = r"c:\windows\fonts\simsun.ttc", size = 15) #导入宋体字
                                                                              #体文件
x = [0,1,2,3,4,5]
y = [1,2,3,2,4,3]
```

```
plt.bar(x,y)                                                    #竖的条形图
plt.title("柱状图",FontProperties = font_set);                  #图标题
plt.xlabel("x 轴",FontProperties = font_set);
plt.ylabel("y 轴",FontProperties = font_set);
plt.show()
```

该例绘制了 6 个柱状形状，用 plt.bar()函数来实现，其中参数为 x、y，该程序的运行结果如图 5-39 所示。

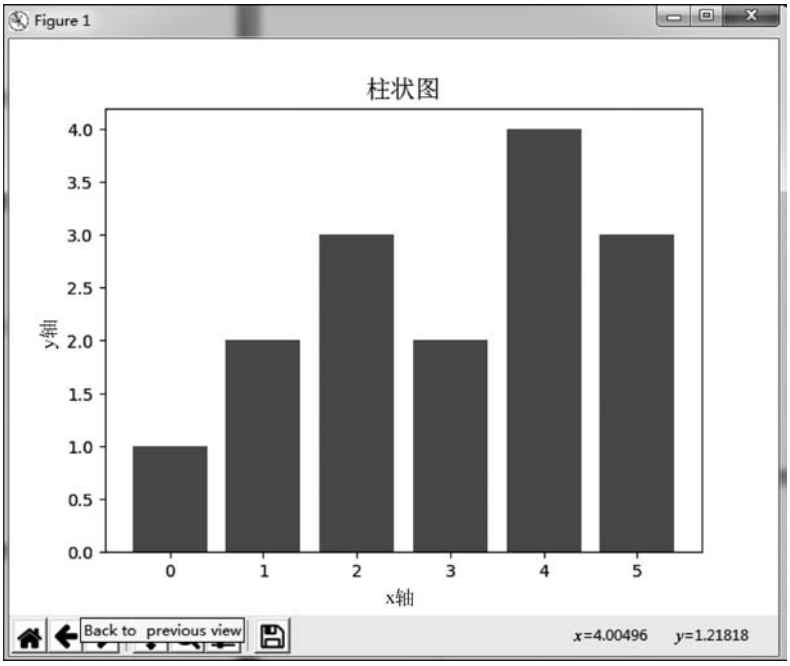

图 5-39 柱状图

在绘制柱状图时，也可以使用 numpy 来实现。

【例 5-13】 用 matplotlib 库和 numpy 库绘制随机出现的柱状图。

代码如下。

```
import matplotlib.pyplot as plt
from matplotlib.font_manager import FontProperties
import numpy as np
font_set = FontProperties(fname = r"c:\windows\fonts\simsun.ttc", size = 15) #导入宋体字
                                                                              #体文件
x = np.arange(10)
y = np.random.randint(0,20,10)
plt.bar(x, y)
plt.show()
```

该例使用 random()函数绘制了在区域中随机出现的柱状图。在语句 y = np.random.randint(0,20,10)中，参数 20 表示柱状图的高度，参数 10 表示柱状图的个数。

该程序的运行结果如图 5-40 所示。

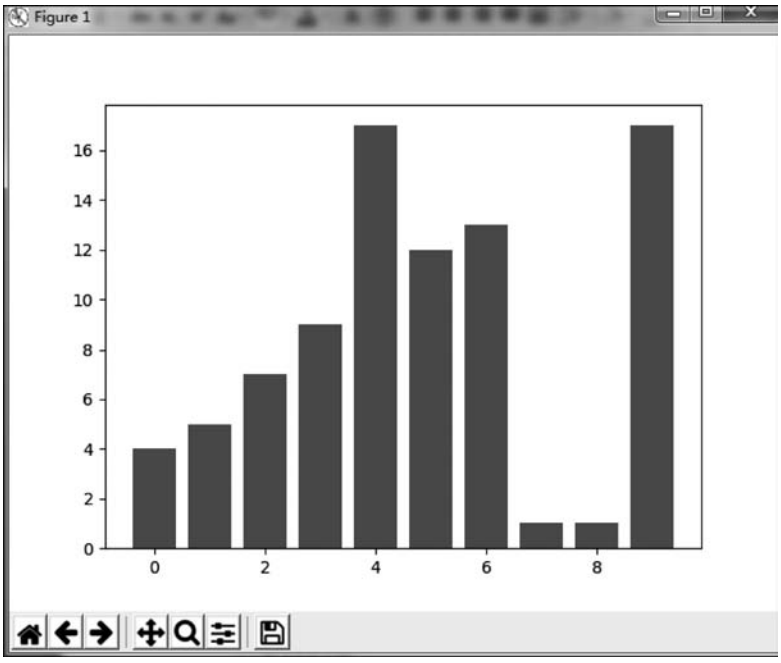

图 5-40　绘制随机的柱状图

5.3.3　绘制直方图

直方图又称为质量分布图，是一种统计报告图，由一系列高度不等的纵向条纹或线段表示数据分布的情况。直方图一般用横轴表示数据类型，用纵轴表示数据的分布情况。

【例 5-14】　用 matplotlib 库绘制直方图。

代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
mean, sigma = 0, 1
x = mean + sigma * np.random.randn(10000)
plt.hist(x,50,histtype = 'bar',facecolor = 'red',alpha = 0.75)
plt.show()
```

该例绘制了一个概率分布的直方图，用 plt.hist()函数来实现。其中参数 mean＝0 设置均值为 0，sigma＝1 设置标准差为 1，该程序的运行结果如图 5-41 所示。

5.3.4　绘制散点图

散点图在回归分析中使用较多，它将序列显示为一组点。值由点在图表中的位置表示，类别由图表中的不同标记表示，因此散点图通常用于比较跨类别的聚合数据。

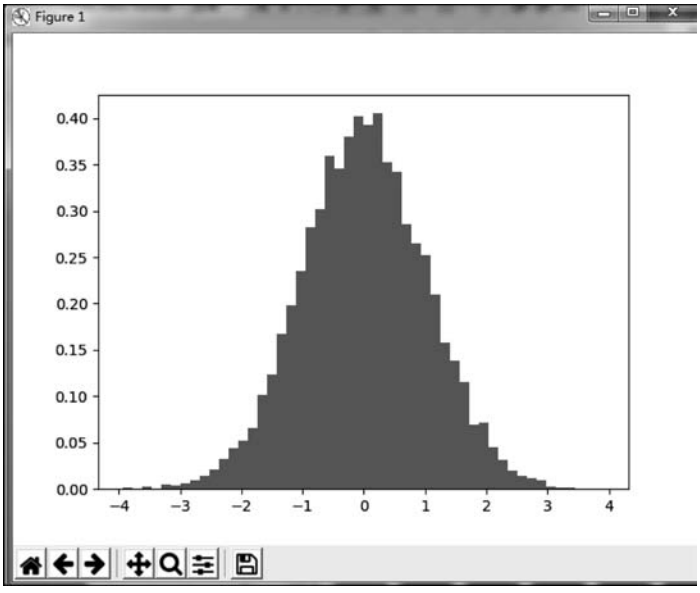

图 5-41　绘制随机的直方图

【例 5-15】 用 matplotlib 库绘制散点图。

代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
x = np.random.rand(100)
y = np.random.rand(100)
plt.scatter(x,y)
plt.show()
```

该例绘制了一个散点图，用 plt.scatter()函数来实现。其中语句 x = np.random.rand(100)和 y=np.random.rand(100)显示了在区域中随机出现的点的个数，该例共有 100 个点。该程序的运行结果如图 5-42 所示。

5.3.5　绘制极坐标图

极坐标图是指在平面内由极坐标系描述的曲线方程图。极坐标是指在平面内由极点、极轴和极径组成的坐标系。极坐标图用于对多维数组进行直接对比，多用在企业的可视化数据模型的对比与分析中。

【例 5-16】 用 matplotlib 库绘制极坐标图。

代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
theta = np.arange(0,2 * np.pi,0.02)
ax1 = plt.subplot(121, projection = 'polar')
ax1.plot(theta,theta/6,'--',lw = 2)
plt.show()
```

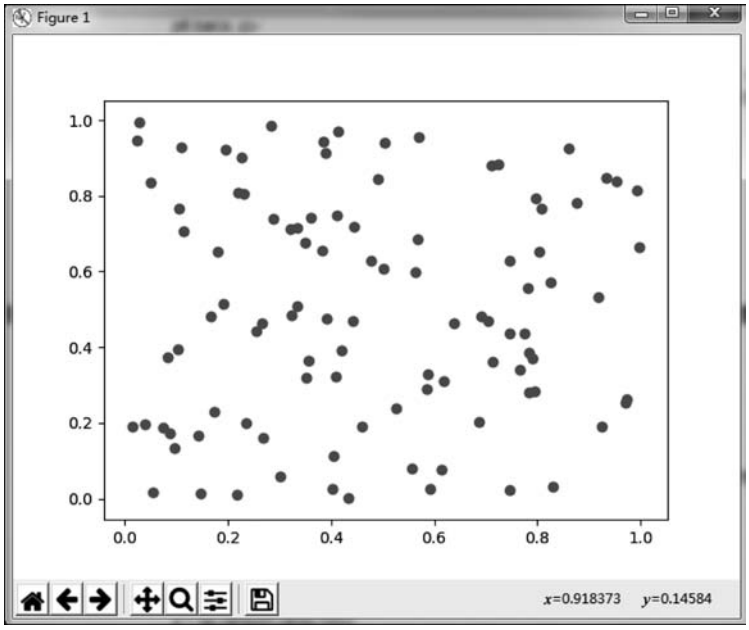

图 5-42　绘制散点图

该例绘制了一个极坐标图，用 plt.polar()函数来实现。在 matplotlib 库的 pyplot 子库中提供了绘制极坐标图的方法，在调用 subplot()创建子图时通过设置 projection＝'polar'即可创建一个极坐标子图，然后调用 plot()在极坐标子图中绘图，其中语句 theta 代表数学上的平面角度。该程序的运行结果如图 5-43 所示。

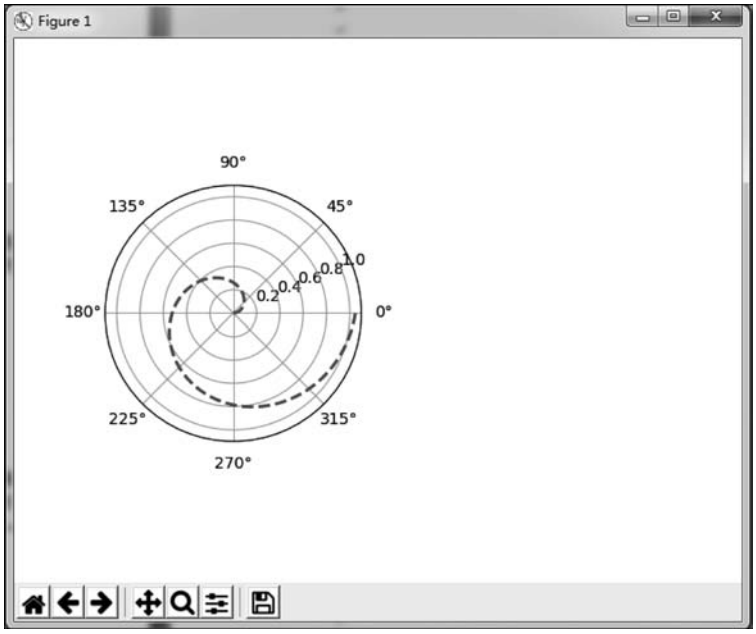

图 5-43　极坐标图

5.3.6 绘制饼图

饼图用于表示不同分类的占比情况，通过弧度大小来对比各种分类。饼图通过将一个圆饼按照分类的占比划分成多个区块，整个圆饼代表数据的总量，每个区块（圆弧）表示该分类占总体的比例大小。

【例 5-17】 用 matplotlib 库绘制饼图。

代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['font.sans-serif'] = ['SimHei']          #设置字体
plt.title("饼图");                                    #设置标题
labels = '计算机系','机械系','管理系','社科系'
sizes = [45,30,15,10]                                 #设置每部分的大小
explode = (0,0.0,0,0)                                 #设置每部分的凹凸
counterclock = False                                  #设置顺时针方向
plt.pie(sizes,explode = explode,labels = labels,autopct = '%1.1f%%',shadow = False,
startangle = 90)                    #设置饼图的起始位置,startangle = 90 表示开始角度为 90°
plt.show()
```

该例绘制了一个饼图，用 plt.pie()函数来实现。程序的运行结果如图 5-44 所示。

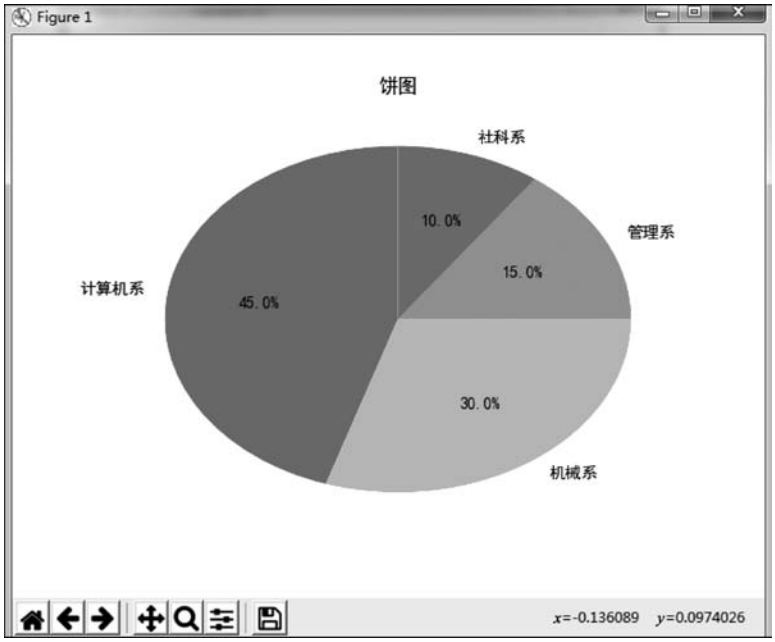

图 5-44 绘制饼图

在饼图的绘制中，如果想将某一部分凸显出来，在语句 explode=(0,0.0,0,0)中将 0 改为 0.1 即可实现。

将图 5-44 中的“机械系”区域凸显，如图 5-45 所示。代码为：

```
explode = (0,0.1,0,0)
```

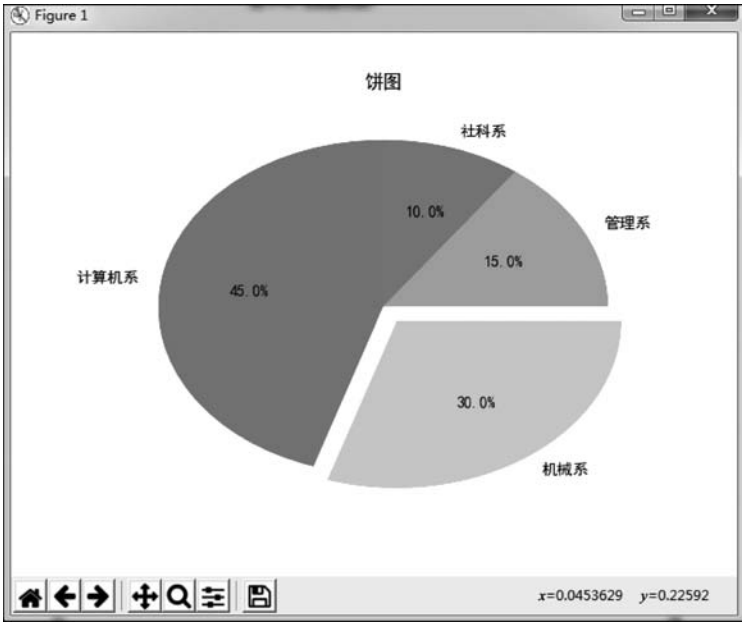

图 5-45　饼图的凸显


视频讲解

5.4　pyecharts 可视化应用

1. pyecharts 数据可视化介绍

pyecharts 是一个用于生成 Echarts 图表的类库，而 Echarts 是一个开源的数据可视化 JS 库，同时也是商业级数据图表，一个纯 JavaScript 的图表库可以流畅地运行在计算机和移动设备上。使用 pyecharts 可以让开发者轻松地实现大数据的可视化。

值得注意的是，目前 pyecharts 分为 v0 和 v1 两大版本，版本之间互不兼容。

2. pyecharts 的安装与使用

在使用 pyecharts 之前首先要安装它，使用以下命令来执行安装过程：

```
pip install pyecharts
```

执行后，可输入以下命令查看：

```
pip list
```

图 5-46 显示安装成功。

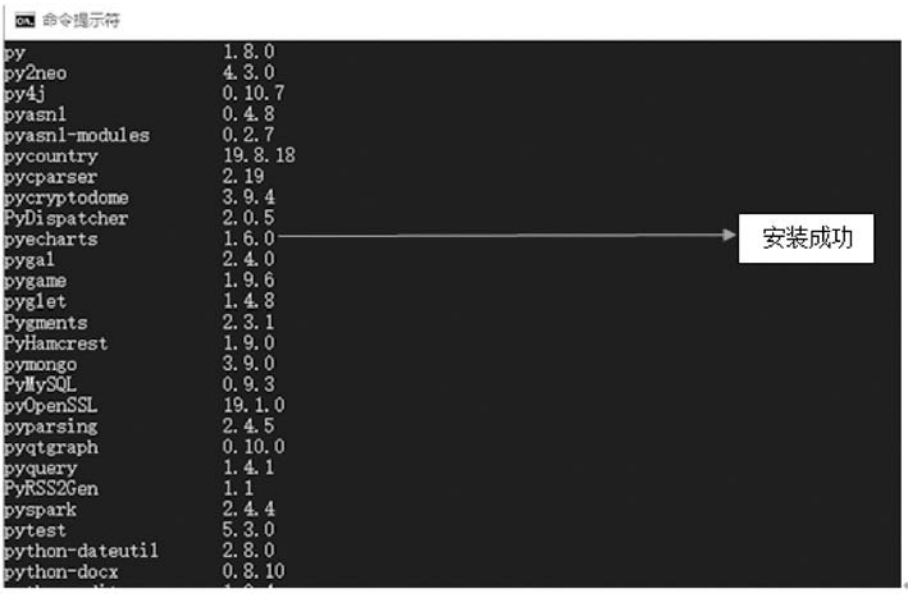

图 5-46　pyecharts 安装成功

从图 5-46 可以看出，pyecharts 安装的版本是 1.6，它与之前的 v0 版本相比发生了较大的变化。

如用户需要用到地图图表，可自行安装对应的地图文件包。命令如下：

```
pip install echarts-countries-pypkg,安装全球国家地图
pip install echarts-china-provinces-pypkg　安装中国省级地图
pip install echarts-china-cities-pypkg　安装中国市级地图
```

在安装完地图库以后，即可进行地图的数据可视化显示。

3. pyecharts 可视化绘图

使用 pyecharts 绘制图形主要有以下几步。

（1）导入库并定义图表的类型。

```
from pyecharts.charts import chart_name
```

（2）创建一个具体类型的实例对象。

```
chart_name = chart_name()
```

（3）添加图表的各项数据。

```
chart_name.add_xaxis; chart_name.add_yaxis
```

（4）添加其他配置。

```
.set_global_opts()
```

（5）生成 html 网页。

```
chart_name.render()
```

值得注意的是，在 v1 版本中是从 pyecharts.charts 中引入元件，而不是从 pyecharts 中引入。

在 pyecharts 中可以绘制多种图表，常见的基本图表类型见表 5-11。

表 5-11　pyecharts 中常见的图表名称以及含义

参 数 名 称	参 数 含 义
Bar	条形图/柱状图
Scatter	散点图
Funnel	漏斗图
Gauge	仪表盘
Line	折线图/面积图
Pie	饼图
Map	地图
Overlap	组合图
Line3D、Bar3D、Scatter3D	3D 折线图、3D 柱状图、3D 散点图
Liquid	水滴球图
Parallel	平行坐标图
Graph	关系图
Geo	地理坐标系
Boxplot	箱形图
EffectScatter	带有涟漪特效动画的散点图
Radar	雷达图
Polar	极坐标图
Sankey	桑基图
WordCloud	词云

这里列举了在 pyecharts 中常用的导入图表类型的方法：

```
from pyecharts.charts import Scatter          #导入散点图
from pyecharts.charts import Line             #导入折线图
from pyecharts.charts import Pie              #导入饼图
from pyecharts.charts import Geo              #导入地图
```

1）绘制柱状图

在 pyecharts 中绘制的柱状图通过柱子的高度和宽度来表现数据的大小。

【例 5-18】 用 pyecharts 库绘制柱状图。

代码如下。

```
from pyecharts.charts import Bar
bar = Bar()
bar.add_xaxis(["数学", "物理", "化学", "英语"])
bar.add_yaxis("成绩", [70, 85, 95, 64])
bar.render()
```

该例通过语句 from pyecharts.charts import Bar 引入了 pyecharts 库。语句 bar = Bar()创建实例，在 pyecharts 中每一个图形库都被封装成为一个类，这就是所谓的面向对象，在开发者使用这个类的时候需要实例化这个类。声明类之后，相当于初始化了一个画布，之后的绘图就是在这个画布上进行。语句 bar.add_xaxis(["数学", "物理", "化学", "英语"])设置了柱状图中 X 轴的数据，bar.add_yaxis("成绩", [70, 85, 95, 64])设置了图例以及 Y 轴的数据。在 pyecharts 中如果要绘制柱状图、散点图、折线图等二维数据图形，由于它既有 X 轴，又有 Y 轴，所以在代码书写中不仅要为 X 轴添加数据，还要为 Y 轴添加数据。最后通过 render()函数生成一个后缀名为 render 的网页，打开该网页即可查看数据可视化的结果。程序运行如图 5-47 所示。

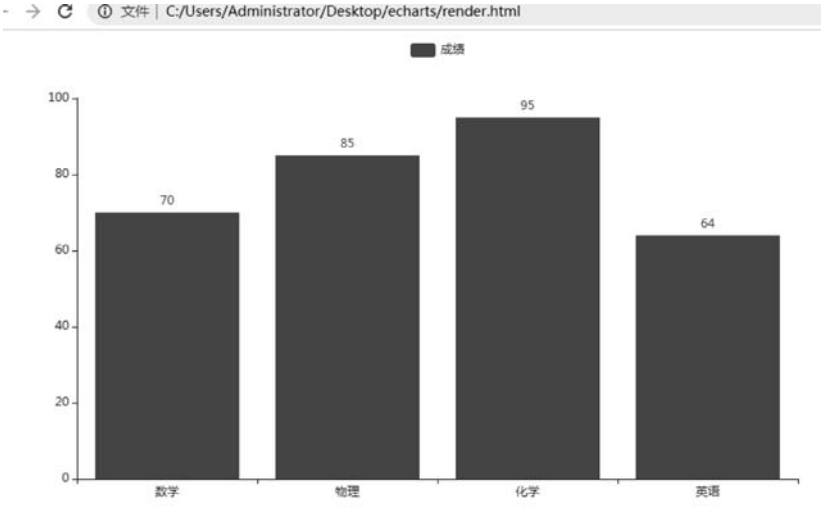

图 5-47　用 pyecharts 绘制柱状图

值得注意的是，pyecharts 在 v1.x 版本中所有方法均支持链式调用(一种设计模式)。因此，本例的代码也可以这样写：

```
from pyecharts.charts import Bar
bar = (
Bar()
.add_xaxis(["数学", "物理", "化学", "英语"])
.add_yaxis("成绩", [70, 85, 95, 64])
)
bar.render()
```

例 5-18 展示了一个图表最基本的信息，而在实际应用中，人们需要向图表中不断地添加信息来展示图表中数据的分布、特点以及做此图的目的等。因此，开发者可以使用 options 来配置各种图表参数。配置项有两种，即全局配置项和系列配置项，配置项越细越能画出更多细节，尤其是全局配置项，它可通过 set_global_options 方法来设置，其中主要的配置内容有 X、Y 坐标轴设置以及初始化配置、工具箱配置、标题配置、区域缩放配置、图例配置、提示框配置等。在 pyecharts 中引入 options 的代码如下：

```
from pyecharts import options as opts
```

本例配置 options 后代码如下：

```
from pyecharts.charts import Bar
from pyecharts import options as opts
bar = (
Bar()
.add_xaxis(["数学", "物理", "化学", "英语"])
.add_yaxis("成绩", [70, 85, 95, 64])
.set_global_opts(title_opts = opts.TitleOpts(title = "期末考试", subtitle = "小明"))
)
bar.render()
```

在这里通过 options 中的 TitleOpts 设置了主标题（title）为期末考试、副标题（subtitle）为小明，运行如图 5-48 所示。

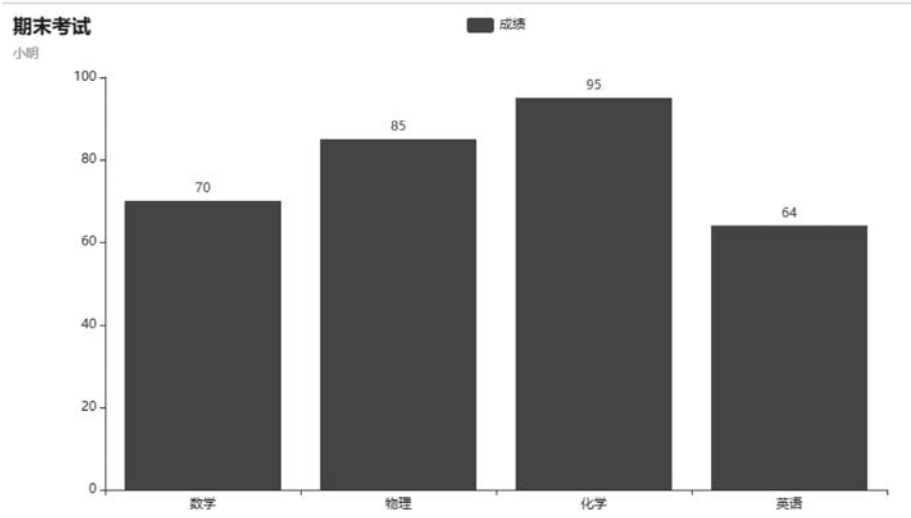

图 5-48　用 pyecharts 绘制柱状图并配置参数

2）绘制仪表盘图

在 pyecharts 中绘制的仪表盘图形是模仿汽车速度表的一种图表，常用来反映预算完成率、收入增长率等比例性指标。它简单、直观，人人会看，通过刻度来精确地展示数据。

【例 5-19】 用 pyecharts 库绘制仪表盘图。

代码如下。

```
from pyecharts import options as opts
from pyecharts.charts import Gauge
c = (
    Gauge()
    .add("", [("", 95)]
         )
    .set_global_opts(title_opts = opts.TitleOpts(title = "任务完成率"))
    .render("任务完成率.html")
)
```

该例通过语句 Gauge 来定义图表的类型为仪表盘。程序的运行显示如图 5-49 所示。

图 5-49 用 pyecharts 绘制仪表盘图

3）绘制饼图

在 pyecharts 中可以使用 pie 生成饼图。

【例 5-20】 用 pyecharts 库绘制饼图。

代码如下。

```
from pyecharts import options as opts
from pyecharts.charts import Page, Pie
import random
pie = (
    Pie()
    .add('鼠标选中分区后的 tip',
         [list(z) for z in zip(['20{}年第{}季'.format(year,season)
                                for year in [19, 20] # count 2
                                    for season in range(1,5)] # count 2
                ,[random.randint(2, 10) for _ in range(8)])]) # count 8
    .set_series_opts(label_opts = opts.LabelOpts(formatter = '{b}: {c}万套'))
    .set_global_opts(title_opts = opts.TitleOpts(title = '饼图实例 - 近两年季度销售'),
                     legend_opts = opts.LegendOpts(is_show = False))
)
pie.render('饼图.html')
```

该例使用语句 pie 来绘制饼图。程序的运行显示如图 5-50 所示。

4）绘制雷达图

在 pyecharts 中绘制的雷达图一般用来进行多指标体系比较分析。从雷达图中可以

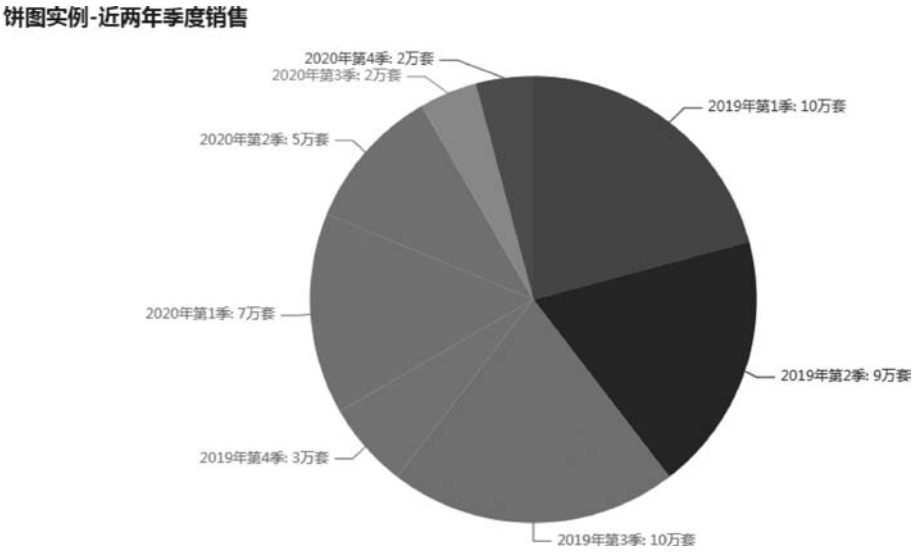

图 5-50　用 pyecharts 绘制饼图

看出指标的实际值与参照值的偏离程度，从而为分析者提供有益的信息。在实际应用中雷达图一般用于成绩展示、效果对比量化、多维数据对比等。

【例 5-21】 用 pyecharts 库绘制雷达图。

代码如下。

```
import random
from pyecharts import options as opts
from pyecharts.charts import Page, Radar
def radar_simple() -> Radar:
    c = (
        Radar()
        .add_schema(
            # 各项的 max_值可以不同
            schema = [
                opts.RadarIndicatorItem(name = '程序设计', max_ = 100),
                opts.RadarIndicatorItem(name = '动态规划', max_ = 100),
                opts.RadarIndicatorItem(name = '图论', max_ = 100),
                opts.RadarIndicatorItem(name = '搜索', max_ = 100),
                opts.RadarIndicatorItem(name = '模拟', max_ = 100),
                opts.RadarIndicatorItem(name = '数论', max_ = 100),
            ]
        )
        .add('小明', [[random.randint(10, 101) for _ in range(6)]],
             color = 'red',
             areastyle_opts = opts.AreaStyleOpts(      # 设置填充的属性
                 opacity = 0.5,
                 color = 'red'
         ),)
        .add('小红', [[random.randint(10, 101) for _ in range(6)]],
             color = 'blue',
             areastyle_opts = opts.AreaStyleOpts(
```

```
                    opacity = 0.5,        #透明度
                    color = 'blue'
            ),)
            .set_series_opts(label_opts = opts.LabelOpts(is_show = True))
            .set_global_opts(title_opts = opts.TitleOpts(title = '雷达图示例 - ACM 集训队队员
能力'))
        )
        return c
radar_simple().render('雷达图.html')
```

该例通过语句 Radar 来定义图表类型为雷达图，其中语句.add_schema 用于定义该图中两个系列的维度数据值以及雷达各维度的范围大小；语句.add 用于定义雷达图上的文字和颜色，以便于区分，其中小明用红线表示，小红用蓝线表示。程序的运行显示如图 5-51 所示。

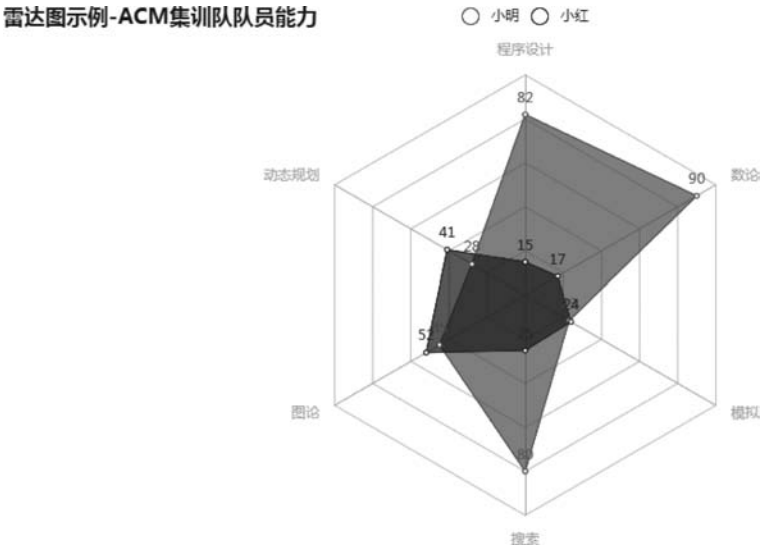

图 5-51　用 pyecharts 绘制雷达图

5）绘制折线图

【例 5-22】 在 pyecharts 中通过 Line 来绘制折线图。

代码如下。

```
from pyecharts.charts import Line
from pyecharts import options as opts
columns = ["Jan", "Feb", "Mar", "Apr", "May", "Jun", "Jul", "Aug", "Sep", "Oct", "Nov",
"Dec"]
#设置数据
data1 = [2.0, 4.9, 7.0, 23.2, 25.6, 76.7, 135.6, 162.2, 32.6, 20.0, 6.4, 3.3]
data2 = [2.6, 5.9, 9.0, 26.4, 28.7, 70.7, 175.6, 182.2, 48.7, 18.8, 6.0, 2.3]

line = (
#调用类
```

```
Line()
#添加 X 轴
 .add_xaxis(xaxis_data = columns)
#添加 Y 轴
 .add_yaxis(series_name = "北京",y_axis = data1)
 .add_yaxis(series_name = "上海",y_axis = data2)
.set_global_opts(title_opts = opts.TitleOpts(title = "降雨量", subtitle = "中国城市"))
)
line.render('zhexiantu.html')
```

运行如图 5-52 所示。

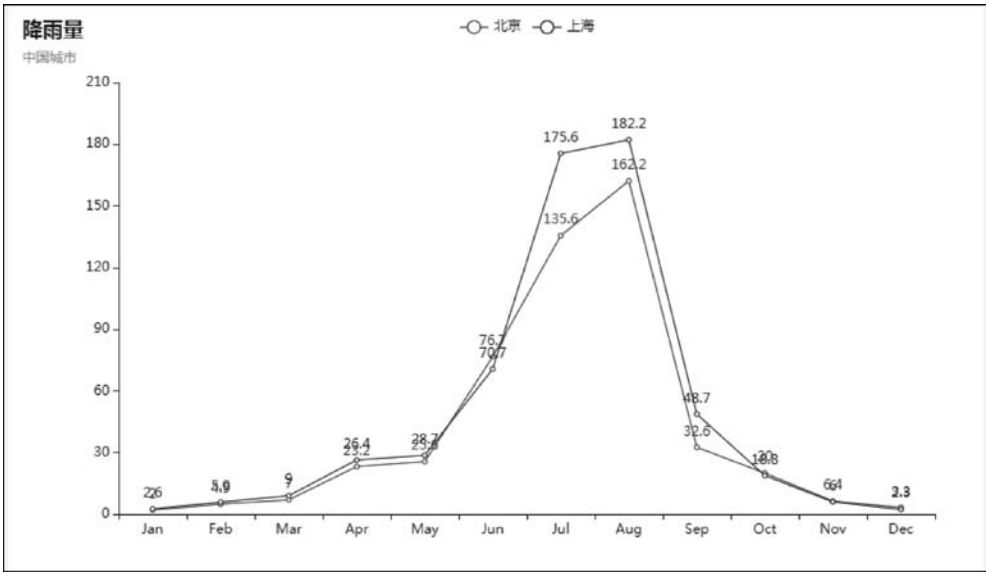

图 5-52　用 pyecharts 绘制折线图

5.5　本章小结

（1）数据可视化系统并不是为了展示用户已知的数据之间的规律，而是为了帮助用户通过认知数据有新的发现，发现这些数据所反映的实质。

（2）数据可视化技术一般由三方面组成，即科学可视化、信息可视化和可视化分析。

（3）按照数据的作用和功能可以把图表分为比较类、分布类、流程类、地图类、占比类、区间类、关联类、时间类和趋势类等。在每一种类型的图表中都可包含不同的数据可视化图形，例如柱状图、饼图、气泡图、热力图、趋势图、直方图、雷达图、色块图、漏斗图、和弦图、仪表盘、面积图、折线图、K 线图、环图、词云等。

（4）在数据可视化中经常需要用到 numpy 中的数组存储以及矩阵运算等功能，并通过调用 matplotlib 库来显示。

（5）pyecharts 是一个用于生成 Echarts 图表的类库，而 Echarts 是一个开源的数据

可视化JS库，同时也是商业级数据图表，一个纯JavaScript的图表库，使用pyecharts可以让开发者轻松地实现大数据的可视化。


视频讲解

5.6 实训

1. 实训目的

通过本章实训了解大数据可视化的特点，能进行简单的与大数据有关的可视化操作，能够绘制大数据可视化图形。

2. 实训内容

（1）绘制折线图，对四个学生的学习能力进行数据可视化的对比，代码如下。

```
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams[ 'font.sans-serif'] =[ 'Microsoft YaHei']        #设置字体
plt.title( '学生学习能力对比')                                  #标题
x = np.arange(1,11,1)
plt.plot(x,x * 1.2)
plt.plot(x,x * 2)
plt.plot(x,x * 3)
plt.plot(x,x * 4)
plt.legend(["黄亚兰","周兰","胡飞","赵云"])
plt.show()
```

该程序的运行结果如图5-53所示。

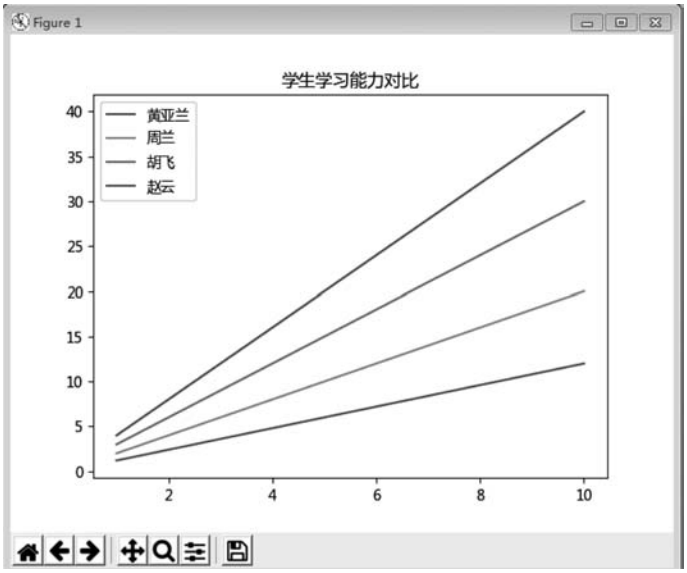

图5-53 学生学习能力对比图

(2) 绘制条形图,对 2021 年四个直辖市的 GDP 进行数据可视化的对比,代码如下。

```
import matplotlib.pyplot as plt
plt.rcParams[ 'font.sans-serif'] =[ 'Microsoft YaHei']                    #设置字体
GDP = [ 28000, 30133, 18590, 19500]                                       #设置 GDP 值
plt.bar(range( 4), GDP, align = 'center',color= 'blue', alpha = 1)        #绘图
plt.ylabel( 'GDP')                                                        #添加轴坐标
plt.title( '2021 年四个直辖市的 GDP 对比')                                  #标题
plt.xticks(range( 4),[ '北京市', '上海市', '天津市', '重庆市'])
plt.ylim([ 5000, 35000])                                                  #设置 y 轴范围值
plt.show()
```

运行该程序如图 5-54 所示。

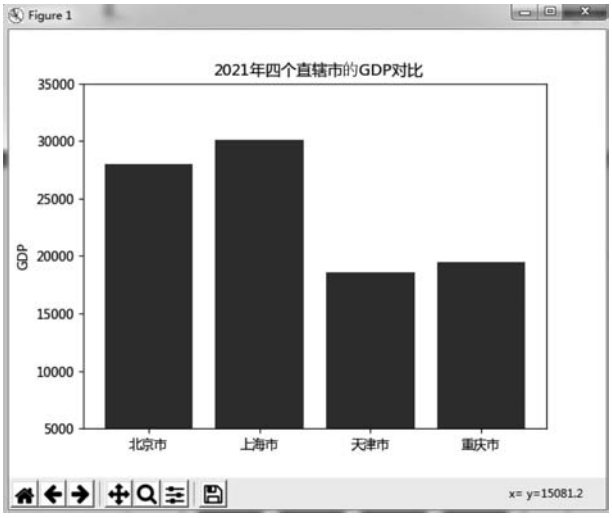

图 5-54　显示直辖市 GDP 数据可视化

(3) 使用子图来绘制曲线,代码如下。

```
import matplotlib.pyplot as plt
import numpy as np
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
x = np.arange(1,4 * np.pi,0.01)
y = np.cos(x)
z = np.sin(x)
plt.figure()
plt.subplot(121)
plt.title("余弦")
plt.plot(x,y,color = "r")
plt.subplot(122)
plt.title("正弦")
plt.plot(x,z,color = "c")
plt.show()
```

这里使用语句“plt. rcParams['font. sans-serif']=['SimHei']”设置字体，使用语句“plt. rcParams['axes. unicode_minus']=False”设置符号。该程序的运行结果如图 5-55 所示。

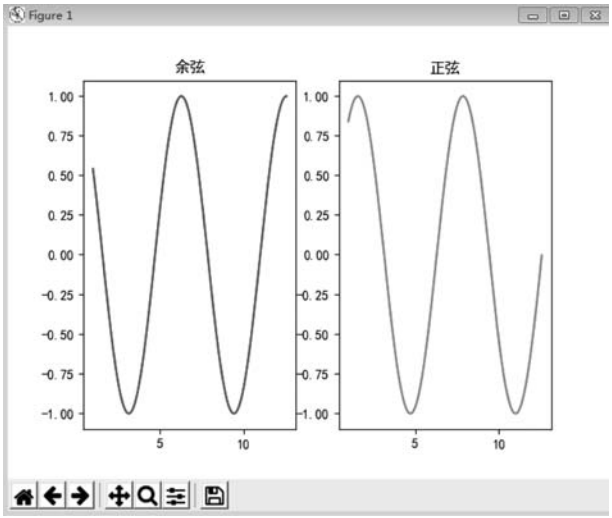

图 5-55　用子图绘制曲线

(4) 绘制多条折线图显示城市降雨量，代码如下。

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif'] = ['SimHei']
x = [1,2,3,4]
y = [6,7,8,2]
plt.plot(x,y,c = "green",marker = "o")
plt.title("降雨量");
plt.xlabel("月份");
plt.ylabel("降雨量");
a = [1,2,3,4]
b = [4,8,5,7]
plt.plot(a,b,c = "red",marker = "o")
z = [1,2,3,4]
w = [7,6,4,5]
plt.plot(z,w,c = "pink",marker = "o")
plt.legend(["重庆降雨量","成都降雨量","西安降雨量"])
labels = ['1月', '2月', '3月', '4月']
plt.xticks(x,labels)
plt.show()
```

该程序的运行结果如图 5-56 所示。

(5) 使用 pyecharts 绘制图书销售量对比图。

```
from pyecharts.charts import Bar
from pyecharts import options as opts
from pyecharts.globals import ThemeType
```

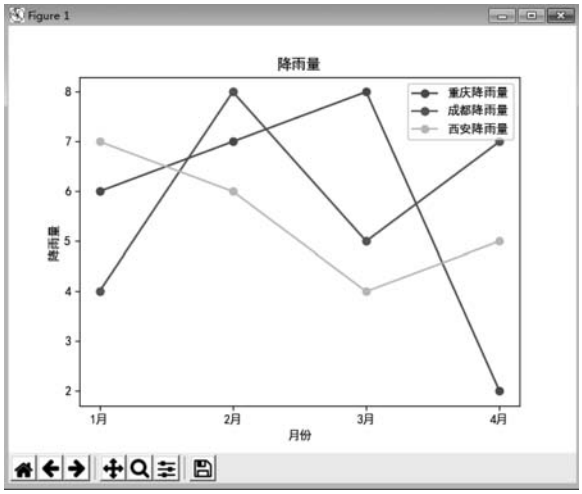

图 5-56　用折线图显示城市降雨量

```
bar = (
    Bar(init_opts = opts.InitOpts(theme = ThemeType.LIGHT))
    .add_xaxis(["哲学", "历史", "教育", "科技", "文学", "经济"])
    .add_yaxis("商家 A", [25, 20, 36, 40, 75, 90])
    .add_yaxis("商家 B", [35, 26, 45, 50, 35, 66])

    .set_global_opts(title_opts = opts.TitleOpts(title = "图书销售量", subtitle = "2020
年"))
)
bar.render('柱状图.html')
```

该程序的运行结果如图 5-57 所示。

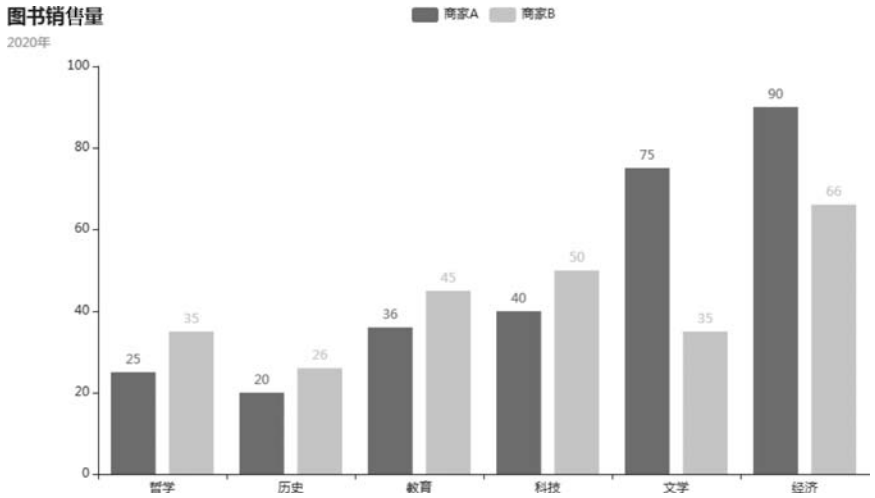

图 5-57　图书销售量对比图

习题

1. 什么是大数据可视化？
2. 大数据可视化对当今世界有哪些影响？
3. 大数据可视化面临哪些挑战？
4. 大数据可视化有哪些图表？
5. 在大数据可视化中 numpy 库有哪些作用？
6. pyecharts 库和 matplotlib 库有哪些区别？
7. 如何使用 matplotlib 库绘制条形图？
8. 如何使用 pyecharts 库绘制中国地图？
9. 如何使用 pyecharts 库绘制雷达图？