# 第 7 章 科学计算与数据分析基础

学习目标

- 学会使用 numpy 库和 pandas 库进行矩阵分析和数值运算
- 学会使用 matplotlib 库绘制图形

科学计算是为了解决科学和工程中数学问题而利用计算机进行的数值计算，它不仅可以帮助科学工作者通过计算发现自然规律，也是普通人提升专业化程度的必要手段，Python 语言的第三方库 Scipy（包含 numpy、pandas 以及 matplotlib 等）使用方便简单，使得非专业人士也可以轻松进行专业的科学计算。

## 7.1 numpy 库的使用

numpy 是高性能科学计算和数据分析的基础包，提供了重要的数据结构——N 维数组，即 ndarray，它是相同类型元素的集合，第 4 章中介绍的列表也可以表示数组，但是列表中的元素的类型可以互不相同。

numpy 是 Python 的第三方库，所以使用前需要先安装，安装后常使用 import numpy as np 来导入 numpy 库，np 是常用别名。

### 7.1.1 什么时候需要 numpy

科学计算中，经常会用到 n（n=1,2,3…）维矩阵，1 维矩阵可以理解为一行数据，2 维矩阵是有行和列的数据表格，矩阵的很多计算实现复杂，使用 numpy 可以大大简化计算的复杂性。

计算两个列表的乘积，不能直接使用乘法运算符，需要使用循环语句分别取出每一个元素来计算，如果元素数量较多，Python 的循环的低效性就将显现出来，但是，如果使用 numpy，不光可以直接计算，而且因为 numpy 是 C 语言编写的，运算效率也较高。

```
>>> a=[1,2,3]
>>> b=[2,4,6]
>>> c=a*b
Traceback (most recent call last):
  File "<pyshell#2>", line 1, in <module>
    c=a*b
TypeError: can't multiply sequence by non-int of type 'list'
>>> c=[]
>>> for i in range(len(a)):
    c.append(a[i]*b[i])
>>> c
[2, 8, 18]
>>> import numpy as np
>>> np.array(a)*np.array(b)
```

```
array([ 2,  8, 18])
```

### 7.1.2 创建 ndarray

创建 ndarray 的方法有很多，如表 7-1 所示。

表 7-1 创建 adarray 的常用函数表

| 分类 | 函数 | 举例 |
| --- | --- | --- |
| 从 Python 的类似 array 的数据类型转换而来 | array() | >>> np.array([1,2,3])<br>array([1, 2, 3])<br>>>> np.array((1,2,3))<br>array([1, 2, 3]) |
| numpy 用来创建 ndarray 的内部函数 | zeros() | >>> np.zeros((2,3))<br>array([[0., 0., 0.],<br>[0., 0., 0.]]) |
| | ones() | >>> np.ones((2,3))<br>array([[1., 1., 1.],<br>[1., 1., 1.]]) |
| | arange() | >>> np.arange(2,3,0.2)<br>array([2. , 2.2, 2.4, 2.6, 2.8]) |
| | linspace() | >>> np.linspace(2,3,5)<br>array([2. , 2.25, 2.5 , 2.75, 3. ]) |
| 特殊的库函数 | random() | >>> np.random.rand(2,3)<br>array([[0.49417606, 0.81456322, 0.28103888],<br>[0.01292281, 0.15490604, 0.05986464]]) |

表 7-1 中给出了创建 ndarray 的常用函数范例，更为详细的参数说明、返回值说明以及范例，可以使用 help()获得，help()是 Python 中获得帮助的通用函数，是学习的好帮手。下面是获得 zeros()函数帮助信息的语句，运行结果较长，没有列出。

```
>>> help(np.zeros)
```

### 7.1.3 ndarray 的基本特性

ndarray 的常用属性中，ndim 是 ndarray 的维度的数量，也称为秩，shape 是 ndarray 的每个维度大小组成的元组、size 是 ndarray 中的元素的总个数、dtype 是 ndarray 中的元素的数据类型，itemsize 是 ndarray 中的元素占用的字节数，下面例子中创建了一个 2 行 3 列的由随机数组成的 ndarray，并通过实例运行显示各参数的含义。

```
>>> import numpy as np
>>> x=np.random.rand(2,3)
>>> x
array([[0.59887136, 0.59165017, 0.92151827],
       [0.22801505, 0.12739391, 0.45758655]])
>>> x.ndim          # ndim 属性是 ndarray 的维度的数量，也称为秩
2
>>> x.shape         # shape 属性是 ndarray 的每个维度大小组成的元组
(2, 3)
>>> x.size          # size 属性是 ndarray 中的元素的总个数
6
>>> x.dtype         # dtype 属性是 ndarray 中的元素的数据类型
dtype('float64')
```

```
>>> x.itemsize      # itemsize 属性是 ndarray 中的元素占用的字节数
8
```

### 7.1.4 ndarray 的基本操作

ndarray 的基本操作包括加减乘除运算、索引和切片、修改维度以及常用的统计运算。

（1）加减乘除运算

```
>>> a=[1,2,3]
>>> b=[4,5,6]
>>> a+b                                  #列表的加法运算，不是相应元素的加法
[1, 2, 3, 4, 5, 6]
>>> import numpy as np
>>> np.array(a)+np.array(b)              #两个 ndarray 的加法运算
array([5, 7, 9])
>>> np.array(a)-np.array(b)              #两个 ndarray 的减法运算
array([-3, -3, -3])
>>> np.array(a)*np.array(b)              #两个 ndarray 的乘法运算
array([ 4, 10, 18])
>>> np.array(a)/np.array(b)              #两个 ndarray 的除法运算
array([0.25, 0.4 , 0.5 ])
>>> 5*np.array(a)                        #普通数值和 ndarray 的运算
array([ 5, 10, 15])
```

（2）索引和切片

ndarray 的索引和切片与序列的索引和切片类似。

```
>>> import numpy as np
>>> x=np.array([[[0,1,2],[3,4,5]],
                [[6,7,8],[9,10,11]]])
>>> x
array([[[ 0,  1,  2],
        [ 3,  4,  5]],

       [[ 6,  7,  8],
        [ 9, 10, 11]]])
>>> x[0]
array([[0, 1, 2],
       [3, 4, 5]])
>>> x[-1]
array([[ 6,  7,  8],
       [ 9, 10, 11]])
>>> x[0,1]                    等于x[0][1]
array([3, 4, 5])
>>> x[0,1,2]
5
>>> x[:,0,0]
```

```
array([0, 6])
```

（3）修改维度

可以使用 reshape()方法修改数组维度，同时保持原数组的值不变。

```
>>> import numpy as np
>>> x=np.ones((2,3))
>>> x
array([[1., 1., 1.],
       [1., 1., 1.]])
>>> y=x.reshape(3,2)
>>> y
array([[1., 1.],
       [1., 1.],
       [1., 1.]])
```

默认浮点型

（4）常用的统计运算

表 7-2 numpy 库中的常用统计函数

| 函数 | 功能描述 |
|---|---|
| sum() | 按指定轴返回数组元素的和 |
| mean() | 按指定轴返回数组元素的平均值 |
| max() | 按指定轴返回数组元素的最大值 |
| min() | 按指定轴返回数组元素的最小值 |
| var() | 按指定轴返回数组元素的方差 |
| std() | 按指定轴返回数组元素的标准差 |
| argmin() | 按指定轴返回数组元素的最小值的索引 |
| argmax() | 按指定轴返回数组元素的最大值的索引 |

举例如下。

```
>>> x=np.array([[1,2,3],[4,5,6]])
>>> x.sum()                              #求 ndarray 中所有元素的和
21
>>> x.sum(axis=0)                        #按行求和
array([5, 7, 9])
>>> x.sum(axis=1)                        #按列求和
array([ 6, 15])
>>> x.mean()
3.5
>>> x.min()
1
>>> x.max()
6
>>> x.var()
2.9166666666666665
>>> x.std()
1.707825127659933
>>> x.argmin()
0
```

```
>>> x.argmax()
5
```

## 7.2 pandas 库的使用

pandas 库是以 numpy 库为基础构建的，通常用来处理表格型（关系型）的数据集或与时间序列相关的数据集，这一点与 numpy 库不同，numpy 库的优势是矩阵运算，pandas 提供了高效地操作大型数据集所需的工具，提供了大量能快速便捷地处理数据的函数和方法，是使 Python 成为强大而高效的数据分析环境的重要因素之一。

最好的理解 pandas 库中数据结构的方法，就是把它看作一个更低维数据的灵活的容器，例如，DataFrame 是 Series 的容器，Series 是 scalar（标量，pure quantity）的容器。

安装好 pandas 库之后，常使用 import pandas as pd 导入 pandas 库，pd 是常用的别名。

### 7.2.1 Series

Series（变长字典）是一种有序的字典，由一组数据以及一组与之相关的数据标签（即索引）组成。仅有一组数据也可以产生简单的 Series 对象，需要注意的是 Series 中的索引值是可以重复的。

**（1）Series 的创建**

创建 Series 的基本方法是调用 Series 方法。

s = pd.Series(data, index=index)

参数 data 可以是很多不同的类型，例如字典、numpy 的 ndarray、标量数值（如数字 5），参数 index 是标签列表。如果 data 是 ndarray，那么给出的 index 必须和 data 长度相同，如果没有给出 index，则系统自动产生一个 index，即[0,...,len(data)-1]。

```
>>> import pandas as pd
>>> import numpy as np
>>> s=pd.Series(np.random.rand(3),index=['a','b','c'])
>>> s
a    0.931964
b    0.194951
c    0.766217
dtype: float64
>>> s=pd.Series(np.random.rand(3))
>>> s
0    0.213746
1    0.858296
2    0.222178
dtype: float64
```

也可以将字典、纯数据转换成 Series。

```
>>> d={'b':1,'a':0,'c':2}
>>> e=pd.Series(d)
>>> e
```

```
>>> d=pd.Series(5,index=['a','b','c'])
>>> d
a    5
```

```
b    1
a    0
c    2
dtype: int64
```

```
b    5
c    5
dtype: int64
```

**（2）Series 的操作**

Series 的有些操作和 ndarray 相似。注意，使用整数索引对 Series 进行切片时，右侧最大索引值是不包括的，类似于字符串的切片，但是使用标签索引对 Series 进行切片时，切片区间会包含右侧最大索引。

```
>>> d=pd.Series({'b':1,'a':0,'c':2})
>>> d
b    1
a    0
c    2
dtype: int64
>>> d[0:2]                          #使用整数索引对 Series 进行切片
b    1
a    0
dtype: int64
>>> d["b":"a"]                      #使用标签索引对 Series 进行切片
b    1
a    0
dtype: int64
>>> d[d>d.median()]
c    2
dtype: int64
>>> d[[0,2]]
b    1
c    2
dtype: int64
>>> d[[2,1,1]]
c    2
a    0
a    0
dtype: int64
>>> np.exp(d)
b    2.718282
a    1.000000
c    7.389056
dtype: float64
```

Series 的有些操作和字典类似。

```
>>> d=pd.Series({'b':1,'a':0,'c':2})
>>> d
b    1
a    0
```

```
c    2
dtype: int64
>>> d['a']
0
>>> 'd' in d
False
```

### 7.2.2 DataFrame

DataFrame 是二维的表格型数据结构，包含有一组有序的列，每列可以是不同的值类型（数值，字符串，布尔型等），DataFrame 既有行索引也有列索引，可以将 DataFrame 理解为 Series 的容器。

7.4 实例中的 df=ts.get_k_data('000651',start='2019-01-01')，df 就是一个 DataFrame，存储了格力空调自 2019 年开始的交易日 date、开盘价 open、收盘价 close、最高价 high、最低价 low、成交量 volume 以及股票代码 code。

**（1）DataFrame 的创建**

从 Series 的字典创建。

```
>>> d = {'one': pd.Series([1., 2., 3.], index=['a', 'b', 'c']),
     'two': pd.Series([1., 2., 3., 4.], index=['a', 'b', 'c', 'd'])}
>>> df=pd.DataFrame(d)
>>> df
   one  two
a  1.0  1.0
b  2.0  2.0
c  3.0  3.0
d  NaN  4.0
```

从 ndarray 或者列表的字典创建。

```
>>> d={'one':[1,2,3,4],'two':[4,3,2,1]}
>>> df=pd.DataFrame(d)
>>> df
   one  two
0    1    4
1    2    3
2    3    2
3    4    1
```

实际应用中，一些数据通常存在 Excel 文件和 CSV（Comma-Separated Values，逗号分隔值）文件中，CSV 文件一般采用.csv 作为扩展名，可以通过记事本或者 Excel 打开。

DataFrame 对象也可以很方便地从这两种文件中导入或者导出数据。导入 Excel 文件的 read_excel()方法语法规定如下：

pandas.read_excel(io, sheet_name = 0, header = 0)

其中参数 io 是文件的路径字符串，参数 sheet_name 是工作表名称字符串，header 参数不写或者取 0 表示第一行作为标题行，如果没有标题行，则 header 取 None。

假定 D 盘根目录下有数据文件 gj.xlsx，如图 7-1 所示，将其数据导入到 DataFrame 中，可以使用语句 df=pd.read_excel("d:\\gj.xlsx")。

| A | B | C | D |
|---|---|---|---|
| 股票名称 | 交易所 | 股价 | 成交量 |
| 格力电器 | 深圳 | 62.38 | 249793 |
| 中国平安 | 上海 | 81.03 | 491605 |
| 上海机场 | 上海 | 71.71 | 120156 |
| 长江电力 | 上海 | 17.89 | 237044 |
| 粤高速A | 深圳 | 7.66 | 38425 |

图 7-1 Excel 文件 gj.xlsx 内容图

```
>>> import pandas as pd
>>> df=pd.read_excel("d:\\gj.xlsx")
>>> print(df)
   股票名称 交易所   股价    成交量
0  格力电器   深圳    62.38   249793
1  中国平安   上海    81.03   491605
2  上海机场   上海    71.71   120156
3  长江电力   上海    17.89   237044
4  粤高速 A   深圳   429.00    38425
```

DataFrame 对象也可以导出为文件，如下三个语句分别导出为 Excel 文件、CSV 文件和 HTML 文件。

```
df.to_excel("result1.xlsx",sheet_name="Sheet1")
df.to_csv("result2.csv")
df.to_html("result3.html")
```

**（2）DataFrame 的操作**

从语义上，可以把 DataFrame 理解为 Series 的字典，列的选择、增加以及删除等操作都和字典的操作类似。

```
>>> df["股价"]                                    #选择列
0    62.38
1    81.03
2    71.71
3    17.89
4     7.66
Name: 股价, dtype: float64
>>> df[2:4]                                       #选择行
   股票名称 交易所 股价    成交量
2  上海机场   上海   71.71   120156
3  长江电力   上海   17.89   237044
>>> df[df["股价"]>50]                             #筛选出股价高于 50 的股票信息
   股票名称 交易所 股价    成交量
0  格力电器   深圳   62.38   249793
1  中国平安   上海   81.03   491605
2  上海机场   上海   71.71   120156
```

也可以使用 loc 方法设置筛选条件，如果有多个筛选条件，可以使用逻辑与运算符（&）和逻辑或运算符（|），例如筛选股价低于 30，并且成交量大于 200000 的股票。

```
>>> df.loc[(df["股价"]<30) & (df["成交量"]>200000)]
   股票名称 交易所   股价    成交量
```

```
3  长江电力  上海  17.89  237044
```

使用 groupby()方法可以进行分组，然后进行统计计算，DataFrame 常用统计函数功能描述如表 7-3 所示。

```
>>> d=df.groupby("交易所")
>>> a=d.mean()
>>> a
          股价       成交量
交易所
上海    56.876667  282935
深圳    35.020000  144109
```

表 7-3 DataFrame 常用统计函数

| 函数名 | 功能描述 |
|---|---|
| df.describe() | 查看数据列的汇总统计信息 |
| df.sum() | 返回所有列的总和 |
| df.mean() | 返回所有列的平均值 |
| df.count() | 返回每一列的非空值的个数 |
| df.max() | 返回每一列的最大值 |
| df.min() | 返回每一列的最小值 |
| df.median() | 返回每一列的中位数 |
| Df.std() | 返回每一列的标准差 |

## 7.3 matplotlib 库的使用

matplotlib 是 Python 的二维绘图库，可以生成出版质量的图形。pyplot 是常用的画图模块，功能非常强大，安装 matplotlib 后，常使用 import matplotlib.pyplot as plt 语句可以导入 pyplot 模块，plt 是 pyplot 的常用别名。

### 7.3.1 基本绘图函数 plot()

plot()是 pyplot 模块中的最基本的绘图函数，语法格式如下：

plot(x,y,s,linewidth)

函数中，x 表示横坐标的取值范围，可选参数，省略 x 时默认用 y 数据集的索引作 x；y 表示与 x 相对应的纵坐标的取值范围；s 是控制线型的格式字符串，可选参数，省略时绘制的线型采用默认格式，可以通过设定格式字符串来控制点线的颜色、风格样式，plot()函数常用颜色符号含义如表 7-4 所示，plot()函数常用线条符号含义如表 7-5 所示；linewidth 是线的宽度。

使用 plot()绘制的简单的变化关系图，比如表示[1,2,3]和[1,4,9]的变化关系，如图 7-2 所示，可以把绘制的折线理解为经过三个点(1,1)、(2,4)和(3,9)的线。

```
>>> import matplotlib.pyplot as plt
>>> plt.plot([1,2,3],[1,4,9])
>>> plt.show()
```

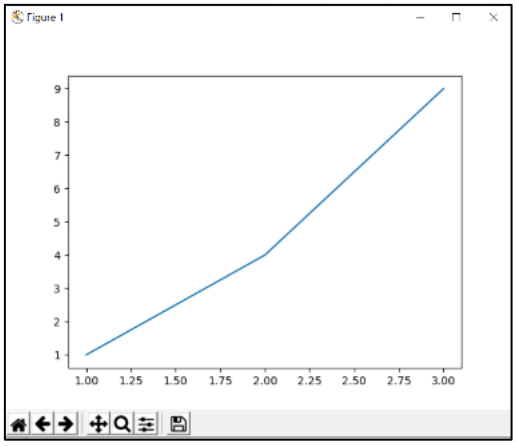



图 7-2 plot()绘制的简单的变化关系图

图 7-3 绘制的是简单的变化关系图，plt.plot()函数中还可以加入丰富的参数，包括点线的颜色、形状等。

表 7-4 plot()函数颜色符号含义表

| 颜色符号 | 含义 | 颜色符号 | 含义 |
|---|---|---|---|
| r | 红色 red | y | 黄色 yellow |
| g | 绿色 green | k | 黑色 black |
| b | 蓝色 blue | w | 白色 white |

表 7-5 plot()函数线条符号含义表

| 线条符号 | 含义 | 线条符号 | 含义 |
|---|---|---|---|
| - | 实线 | < | 左三角 |
| -- | 长虚线 | > | 右三角 |
| : | 短虚线 | ^ | 上三角 |
| -. | 点横线 | v | 倒三角 |
| . | 点 | s | 正方形 |
| , | 像素 | d | 菱形 |
| o | 圆形 | p | 正五边形 |
| * | 星形 | h | 正六边形 |
| \| | 竖线 | + | 十字 |
| x | 叉号 | None | 空 |

对于绘制好的图形，可以对于图形的坐标和标签进行个性化设置，常用设置函数如表 7-6 所示。

表 7-6 常用坐标轴及标签设置函数

| 函数 | 功能说明 |
|---|---|
| axis() | 设置坐标轴属性 |
| xlabel()、ylabel() | 设置 x 轴和 y 轴的标签 |
| title() | 设置绘图区的标题 |
| grid() | 设置网格线是否出现 |
| text() | 在绘图区中指定位置显示文字 |
| legend() | 设置绘图区的图例 |

【例 7-1】绘制 y=x、$y=x^2$ 和 $y=x^3$ 的函数图形。

【参考代码】

ex7-1.py

```
1    import numpy as np
2    import matplotlib.pyplot as plt
3    x=np.linspace(1,5,20)
4    plt.plot(x,x,"rh",x,x**2,"bH",x,x**3,"g^")
5    plt.axis([0,6,0,150])                    #4 个参数分别是[xmin,xmax,ymin,ymax]
6    plt.xlabel("The value of x")
7    plt.ylabel("The value of y")
8    plt.title("The example of pyplot",color="r",fontsize=20)
9    plt.grid(True)
10   plt.text(4.5,80,"y1=x**3")
11   plt.text(5.1,22,"y2=x**2")
12   plt.text(5.1,5,"y3=x")
13   plt.legend(["y1=x**3","y2=x**2","y3=x"])
14   plt.show()
```

【运行结果】

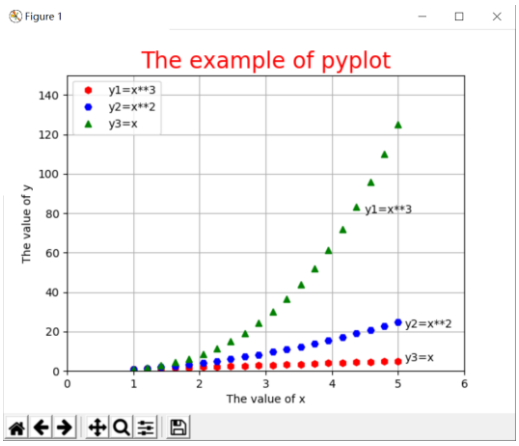



图 7-3 例 7-1 运行结果图

如果图中出现中文，需要语句 plt.rcParams['font.sans-serif']=['SimHei']，此语句可以将中文设置为黑体，否则中文将显示为乱码，需要楷体可以将 SimHei 替换为 Kaiti，微软雅黑可以替换为 Microsoft YaHei。

### 7.3.2 其他常用绘图函数

除了 plt.plot()，pyplot 模块还提供了绘制散点图、饼图、直方图、箱型图等多种图形的函数，如表 7-7 所示。

表 7-7 常用绘图函数表

| 函数 | 功能描述 |
|---|---|
| boxplot() | 绘制箱型图 |
| bar() | 绘制条形图 |
| barh() | 绘制横向条形图 |
| polar() | 绘制极坐标图 |
| pie() | 绘制饼图 |
| psd() | 绘制功率谱密度图 |

| | |
|---|---|
| scatter() | 绘制散点图 |
| step() | 绘制步阶图 |
| stem() | 绘制火柴杆图 |
| hist() | 绘制直方图 |
| contour() | 绘制等值线图 |
| vlines() | 绘制垂直线 |

### 7.3.3 绘制子图

如果需要在一个绘图区域中绘制多个不叠加的图形，以便于同时查看比较，需要用到 pyplot 模块中的 subplot()函数。subplot()函数的语法格式如下：

subplot(nrows,ncols,index)

函数中，nrows 表示将绘图区分割为 nrows 行，ncols 表示将绘图区分割为 ncols 列，index 表示当前子绘图区的索引。子绘图区索引按照行优先顺序，从 1 开始编排序号，依次增 1。2 行 2 列的子图绘制区域示意如图 7-4 所示。

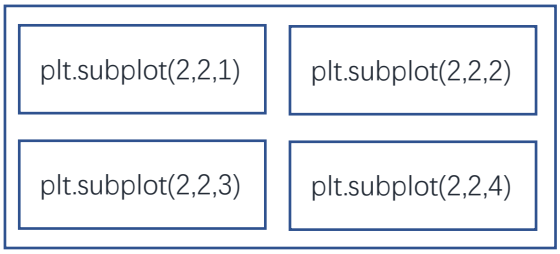



图 7-4 2 行 2 列的 4 个子图绘制区域示意图

【例 7-2】在一个绘图区的 2 行 2 列的 4 个子图绘制区中，分别绘制 1980 年-2018 年我国人均国民生产总值的 4 个变化图（条形图、散点图、饼图和火柴杆图）。

【参考代码 a】

ex7-2a.py

```
1    import matplotlib.pyplot as plt
2    x={1980:468,1990:1663,2000:7942,2010:30808,2018:64644}
3    plt.subplot(2,2,1)
4    plt.bar(x.keys(),x.values())
5    plt.subplot(2,2,2)
6    plt.scatter(x.keys(),x.values())
7    plt.subplot(2,2,3)
8    plt.pie(x.values(),labels=x.keys())
9    plt.subplot(2,2,4)
10   plt.stem(x.keys(),x.values(),use_line_collection=True)
11   plt.show()
```

【运行结果 a】

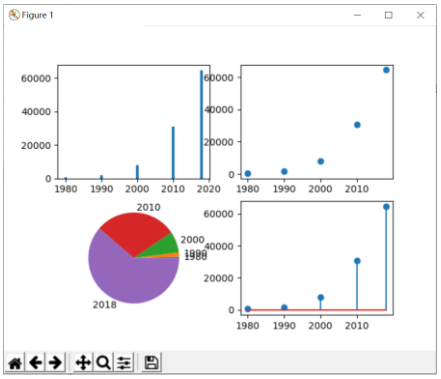



图 7-5 例 7-2 运行结果图 a

从图 7-5 中可以看出，图中出现了图和图之间的交叠，为了解决这个问题，可以使用参考代码 b 中的第 3 行语句 fig, axs = plt.subplots(2, 2, constrained_layout=True)，改进后的代码没有出现交叠问题，如图 7-6 所示。

【参考代码 b】

ex7-2b.py

```
1    import matplotlib.pyplot as plt
2    x={1980:468,1990:1663,2000:7942,2010:30808,2018:64644}
3    fig, axs = plt.subplots(2, 2, constrained_layout=True)
4    axs.flat[0].bar(x.keys(),x.values())
5    axs.flat[1].scatter(x.keys(),x.values())
6    axs.flat[2].pie(x.values(),labels=x.keys())
7    axs.flat[3].stem(x.keys(),x.values(),use_line_collection=True)
8    plt.show()
```

【运行结果 2】

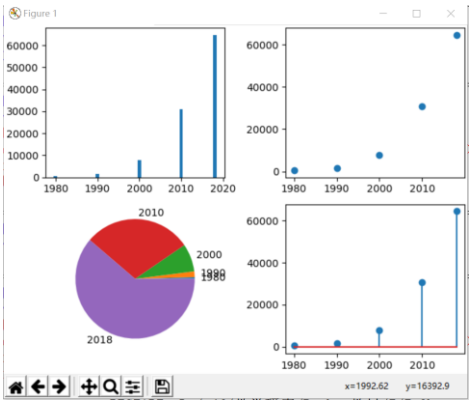



图 7-6 例 7-2 运行结果图 b

## 7.4 实例 财经数据分析

**【实例功能】**通过调用财经数据第三方库 Tushare，获得格力空调 2019 年以来的每日收盘价，并计算出 5 日均线和 30 日均线，通过绘图可视化，观察格力空调股价的走势。

**【实例代码】**

实例 7.py

```
1    import tushare as ts
```

```
2    import numpy as np
3    import matplotlib.pyplot as plt
4    plt.rcParams['font.sans-serif']=['SimHei']
5    df=ts.get_k_data('000651',start='2019-01-01')
6    df["ma5"]=np.NAN
7    df["ma30"]=np.NAN
8    df['ma5'] = df['close'].rolling(5).mean() # 窗口向下滚动 5 个
9    df['ma30'] = df['close'].rolling(30).mean() # 窗口向下滚动 30 个
10   df[['close','ma5','ma30']].plot()
11   plt.title('2019 年格力空调股价走势',fontsize=15)
12   plt.show()
```

**【实例运行】**

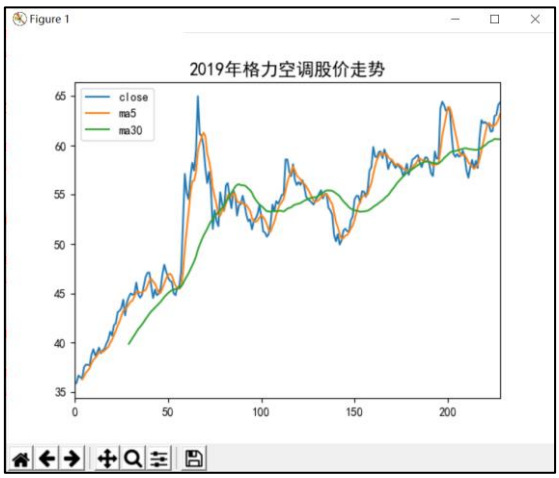



图 7-7 实例运行结果图

【解析】

Tushare 是一个免费、开源的 Python 财经数据接口包,官网 http://tushare.org/，主要实现对股票等金融数据从数据采集、清洗加工到数据存储的过程，能够为金融分析人员提供快速、整洁、和多样的便于分析的数据，为他们在数据获取方面极大地减轻工作量，使他们更加专注于策略和模型的研究与实现上。考虑到 pandas 库在金融量化分析中体现出的优势，Tushare 返回的绝大部分的数据格式都是 pandas 的 DataFrame 类型，非常便于用 pandas/numpy/matplotlib 进行数据分析和可视化。需要说明的是，现在 Tushare 接口不再维护，Tushare pro 新版发布，可以注册使用。

pandas、numpy 以及 matplotlib 都是 Scipy 中的核心库。在 Scipy 官网 www.scipy.org 中，有详细的用户手册可供参考。

第 1 行代码的功能是导入 Tushare 库，第 2 行代码的功能是导入 numpy 库，第 3 行代码的功能是导入 matplotlib 库,这三个库都需要安装后才能使用,安装方法请参考图 1-8。第 4 行代码的功能是解决图标中的中文显示问题，如果没有这句代码，图表中的中文将不能正常显示。

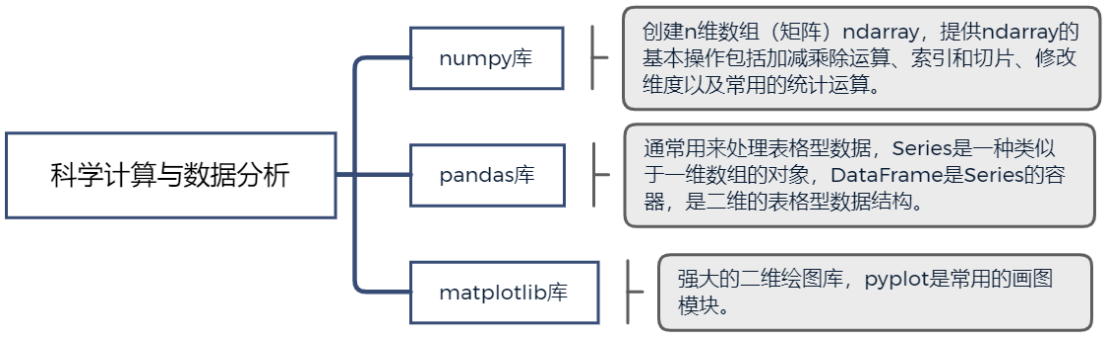
第 5 行代码 df=ts.get_k_data('000651',start='2019-01-01')中，“000651”是格力电器的股票代码，start 参数表示获取数据的开始时间，返回变量 df 是一个 240 行 7 列的 DataFrame，其中 240 行记录，每一行记录是一个交易日的数据，7 列分别为日期 date、开盘价 open、收盘价 close、最高价 high、最低价 low、成交量 volume 以及股票代码 code。下面是 df 的内容。

```
          date   open  close   high    low     volume    code
0   2019-01-02  36.45  35.80  36.45  35.70   424789.0  000651
1   2019-01-03  35.80  35.92  36.19  35.75   258798.0  000651
2   2019-01-04  35.72  36.65  36.70  35.56   489612.0  000651
3   2019-01-07  36.88  36.48  36.96  36.25   392690.0  000651
4   2019-01-08  36.41  36.34  36.42  36.03   193021.0  000651
..         ...    ...    ...    ...    ...        ...     ...
235 2019-12-27  65.15  64.53  65.76  64.40   249793.0  000651
236 2019-12-30  64.57  65.58  65.70  64.05   264579.0  000651
237 2019-12-31  65.35  65.58  65.86  64.80   209307.0  000651
238 2020-01-02  65.99  67.90  68.09  65.90   463614.0  000651
239 2020-01-03  68.38  67.10  68.70  66.90   368703.0  000651

[240 rows x 7 columns]
```

第 6 行和第 7 行是给 df 增加新的两列 5 日均线 ma5 和 30 日均线 ma30，初始值设置为 numpy 中的空值 NAN，第 8 行和第 9 行则使用 rolling()函数自动向下取数据以及 mean()函数求平均值来计算 ma5 和 ma30，第 10 行使用 plot()函数来绘制图表，第 11 行将图表的标题设置为“2019 年格力空调股价走势”，字体大小为 15，第 12 行 plt.show()的功能是显示图表。

## 7.5 本章小结




## 课后习题

一、选择题

1. 下列不属于 ndarray 属性的是_____。

   A. ndim　　B. shape　　C. size　　D. add

2. 创建一个 3 行 3 列的 ndarray 数组，下列代码中错误的是_____。

   A. np.arange(0,9).reshape(3,3)

   B. np.array([3,3])

   C. np.random.rand(3,3)

   D. np.zeros((3,3))

3. 下列参数中调整后显示中文的是_____。

A. lines.linestyle　B. lines.linewidth　C. font.sans-serif　D. axes.unicode_minus

4. 下列代码中绘制散点图的是_____。

A. plt.scatter(x,y)　B. plt.plot(x,y)　C. plt.legend('upper left')　D. plt.xlabel('散点图')

5. 下列字符表示 plot 线条颜色、点的形状和类型为 五角星点短虚线的是_____。

A. 'bs-'　B. 'go-'　C. 'r+-'　D. 'r*:'

二、填空题

1. 生成范围在 0~1、服从均匀分布的 10 行 5 列的数组，可以使用_____语句。
2. 需要在一个绘图区域创建 3 行 2 列的子图绘制区域的其中一个，可以使用 plt._____(3,2,4) 语句。