ipython与Jupyter notebook使用

一：ipython介绍

安装方法： pip install ipython

1. 科学计算标准工具集的组成部分

2. IPython是一个免费、开源的项目，支持Linux、Unix、Mac OS X和Windows平台，其官方网址是http://ipython.org/。
3. IPython中包括各种组件，其中的两个主要组件是：基于终端方式和基于Qt的交互式Python shell，支持多媒体和绘图功能的基于Web的notebook（版本号为0.12以上的IPython支持此功能）

二： Jupyter Notebook介绍

1. Jupyter Notebook（此前被称为 IPython notebook）是一个交互式笔记本，支持运行 40 多种编程语言。
2. Jupyter Notebook 的本质是一个 Web 应用程序，便于创建和共享文学化程序文档，支持实时代码，数学方程，可视化和 markdown。 已迅速成为处理数据的必备工具，用途包括：数据清理和转换，数值模拟，统计建模，机器学习等等
3. jupyter优势

   可选择语言：支持超过40种编程语言，包括Python、R、Java等。

   分享笔记本：可以使用电子邮件、GitHub和Jupyter Notebook Viewer与他人共享。

   交互式输出：代码可以生成丰富的交互式输出，包括HTML、图像、视频、LaTeX等等。

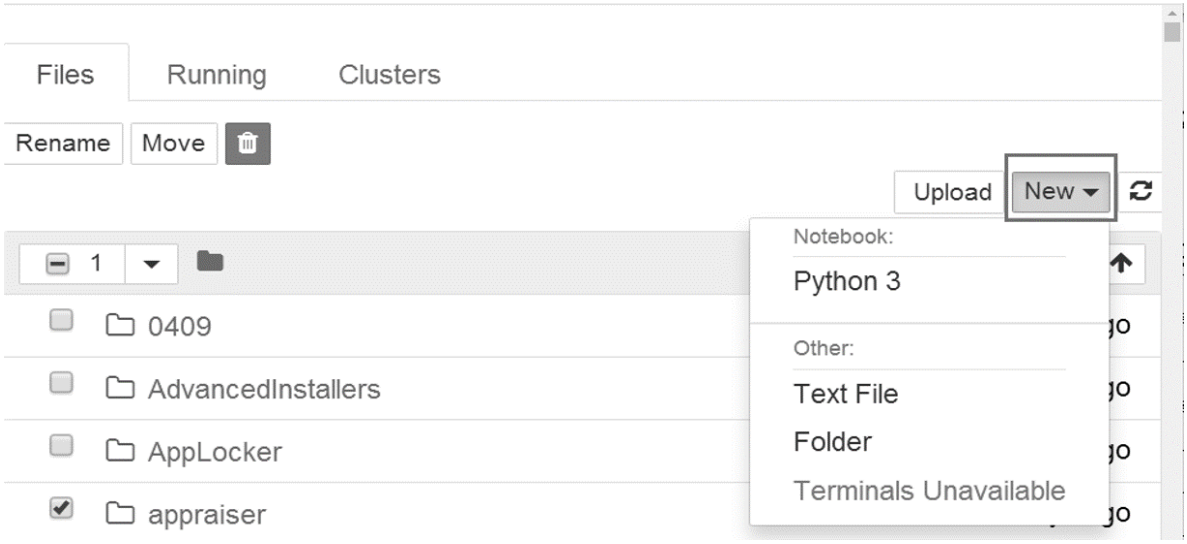
三：Jupyter Notebook使用

1) 打开并新建一个Notebook

“Text File”为纯文本型

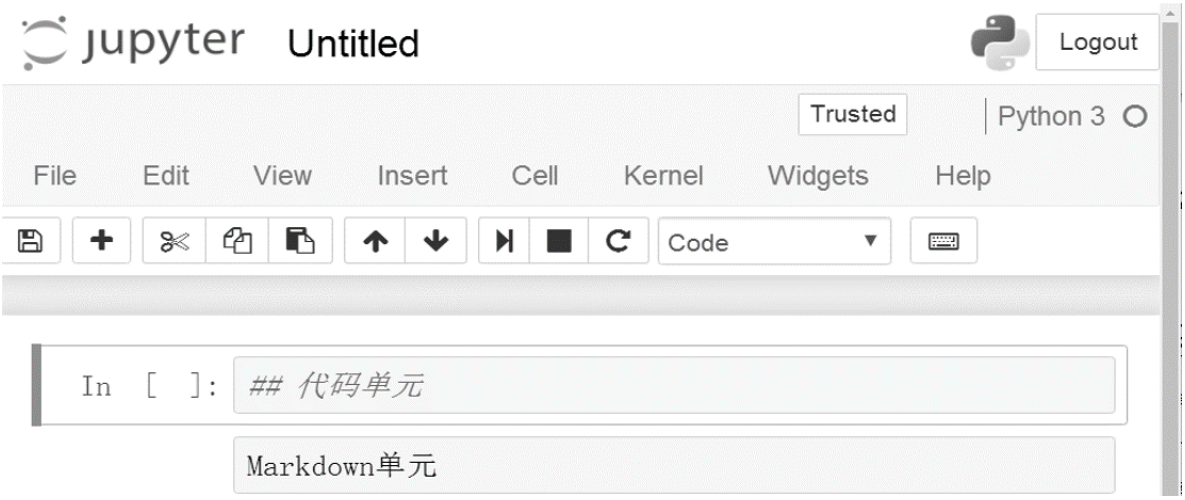
“Folder” 为文件夹

“Python 3” 表示 Python 运行脚本



2) Jupyter Notebook 的界面及其构成

1. 选择”Python 3”选项，进入 Python 脚本编辑界面，Notebook 文档由一系列单元（Cell）构成，主要有两种形式的单元
2. 代码单元。这里是读者编写代码的地方。
3. Markdown 单元。在这里对文本进行编辑。



3) jupyter notebook快捷键

“Esc”键：进入命令模式

“Y” 键：切换到代码单元

“M”键：切换到 Markdown 单元

“B”键：在本单元的下方增加一单元

“H”键：查看所有快捷命令

“Shift + Enter”组合键：运行代码

命令模式：用于执行键盘输入的快捷命令。

4) Markdown 使用

Markdown 是一种可以使用普通文本编辑器编写的标记语言，通过简单的标记语法，它可以使普通文本内容具有一定的格式

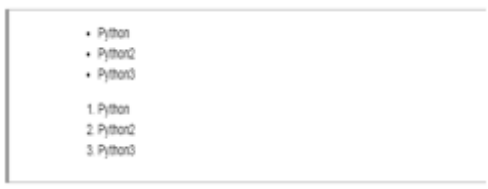
- 标题：标题是标明文章和作品等内容的简短语句。一个“#”字符代表一级标题，以此类推。

```
# 一级标题
## 二级标题
### 三级标题
#### 四级标题
##### 五级标题
###### 六级标题
```

- 列表：列表是一种由数据项构成的有限序列，即按照一定的线性顺序排列而成的数据项的集合。

  对于无序列表，使用星号、加号或者减号作为列表标记

  对于有序列表，则是使用数字“，”“（一个空格）”。

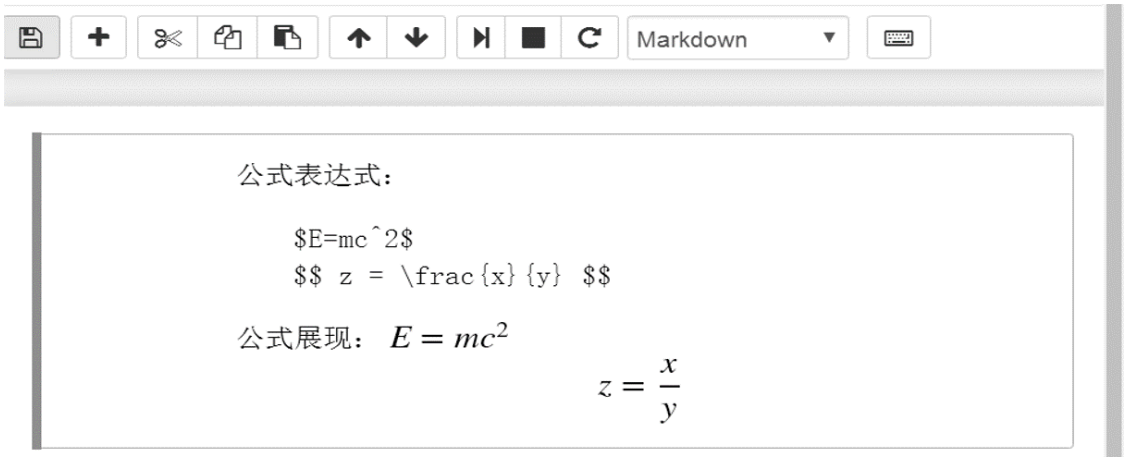


- 加粗 / 斜体：前后有两个星号或下划线表示加粗，前后有 3 个星号或下划线表示斜体。

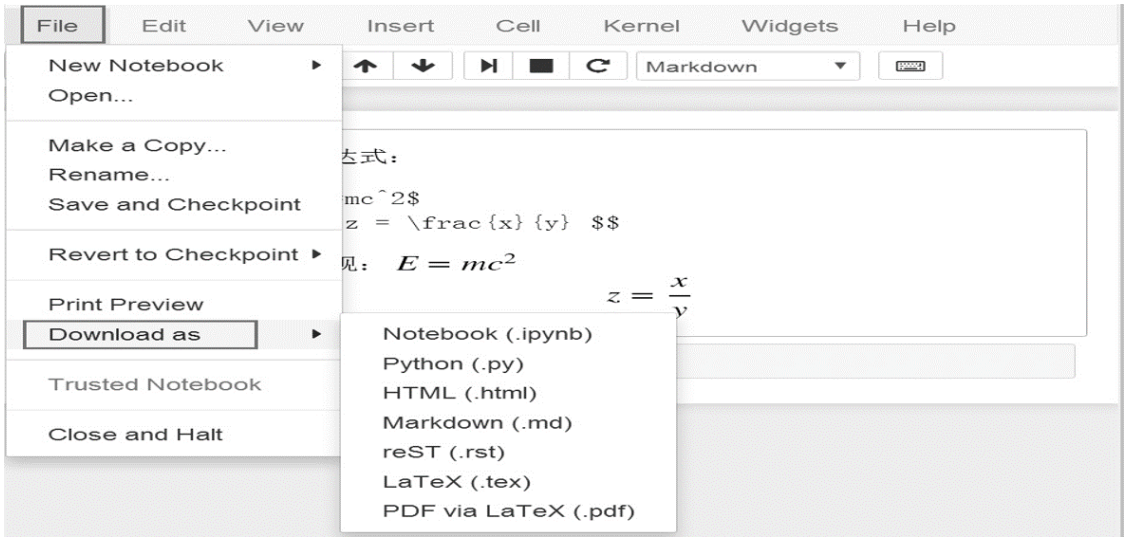
- 数学公式编辑：LaTeX 是写科研论文的必备工具， Markdown 单元中也可以使用 LaTeX 来插入数学公式。

  在文本行中插入数学公式，应在公式前后分别加上一个“$” 符号

  如果要插入一个数学区块，则在公式前后分别加上两个“$$”符号。



- 导出功能：Notebook 还有一个强大的特性，就是导出功能。可以将 Notebook 导出为多种格式，如HTML、 Markdown、reST、PDF (通过 LaTeX) 等格式。导出功能可通过选择“File→Download as



了解matplotlib基础语法及常见参数

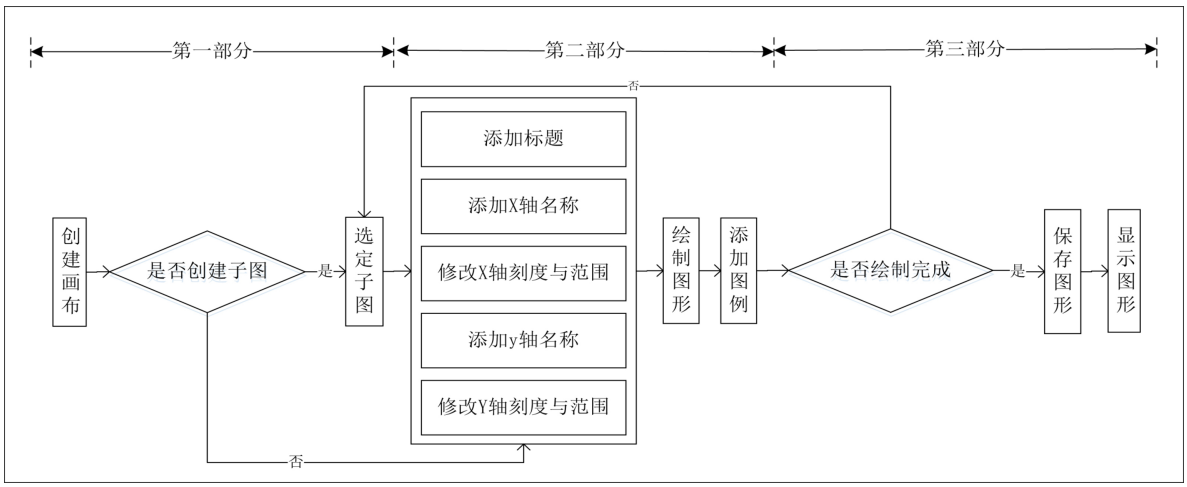
一：基本绘图流程

数据可视化有助于我们对数据的更深入直观的认识

Matplotlib 是一个 Python 的 绘图库，我们使用matplotlib库中的pyplot模块：

通常我们导入语句如下所示： import matplotlib.pyplot as plt

安装方法： pip install matplotlib



二：绘图步骤详情

1. 创建画布与创建子图

第一部分主要作用是构建出一张空白的画布：最简单的绘图可以省略第一部分，直接在默认的画布上进行图形绘制。

plt.figure(num=None, figsize=None, dpi=None, facecolor=None, edgecolor=None)

函数名称	函数作用
num	图像编号或名称，数字为编号 ，字符串为名称
figsize	指定figure的宽和高，单位为英寸,1英寸等于2.5cm
dpi	指定绘图对象的分辨率，即每英寸多少个像素
facecolor	背景颜色
edgecolor	边框颜色

第二部分主要作用是是否将整个画布划分为多个部分，方便在同一幅图上绘制多个图形的情况

figure.add_subplot(子图总行数, 子图总列数, 子图位置)

创建并选中子图，可以指定子图的行数，列数，与选中图片编号。

2. 添加画布内容

第二部分是绘图的主体部分。其中添加标题，坐标轴名称，绘制图形等步骤是并列的，没有先后顺序，可以先绘制图形，也可以先添加各类标签。添加图例一定要在绘制图形之后

函数名称	参数	函数作用
plt.title(s,fontsize,rotation)	标题名称，字体大小，旋转角度	设置标题
plt.xlabel(s,fontsize,rotation)	标签名称，字体大小，旋转角度	在当前图形中添加x轴名称
plt.ylabel(s,fontsize,rotation)	标签名称，字体大小，旋转角度	在当前图形中添加y轴名称
plt.xlim()	x轴范围最小值，x轴范围最大值	指定当前图形x轴的范围
plt.ylim()	y轴范围最小值，y轴范围最大值	指定当前图形y轴的范围
plt.xticks( )	x轴刻度值序列,x轴刻度标签文本序列 [可选]	指定x轴刻度的数目与取值
plt.yticks()	y轴刻度值序列,y轴刻度标签文本序列 [可选]	指定y轴刻度的数目与取值
plt.legend(labels，loc,fontsize)	图例的文本标签,位置，字体	指定当前图形的图例

显示图例的具体位置

```
#    ===============   =============
#    Location String   Location Code
#    ===============   =============
#    'best'            0
#    'upper right'     1
#    'upper left'      2
#    'lower left'      3
#    'lower right'     4
#    'right'           5
#    'center left'     6
#    'center right'    7
#    'lower center'    8
#    'upper center'    9
#    'center'          10
#    ===============   =============
```

3. 保存与展示图形

第三部分主要用于保存和显示图形

函数名称	函数作用
plt.savefig(fname)	保存绘制的图片,传入路径
plt.show()	在本机显示图形。

三：绘制折线图

代码位置：matplotlib_basis.ipynb

1. 折线图（line chart）

是一种将数据点按照顺序连接起来的图形。折线图的主要功能是查看因变量y随着自变量x改变的趋势，最适合用于显示随时间（根据常用比例设置）而变化的连续数据。同时还可以看出数量的差异，增长趋势的变化。

2. 折线图函数

matplotlib.pyplot.plot()

主要参数主要如下。

参数名称	说明
x，y	接收array。表示x轴和y轴对应的数据。无默认。
color	接收特定string。指定线条的颜色。默认为None。
linestyle	接收特定string。指定线条类型。默认为“-”。
marker	接收特定string。表示绘制的点的类型。默认为None。
alpha	接收0-1的小数。表示点的透明度。默认为None。

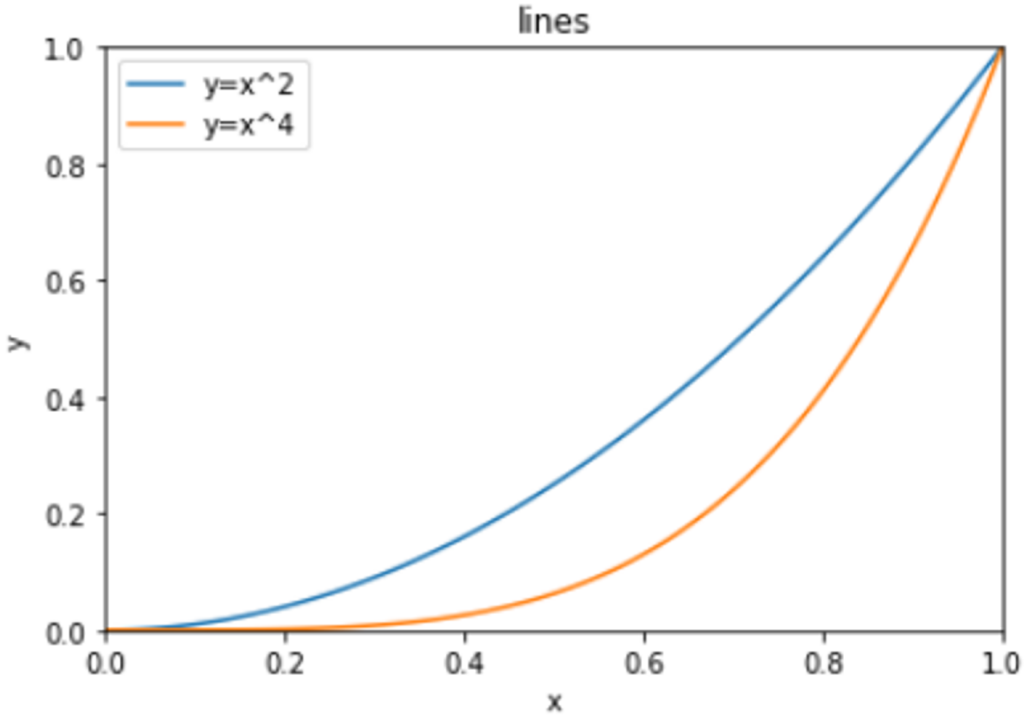
color参数的8种常用颜色的缩写。

颜色缩写	代表的颜色	颜色缩写	代表的颜色
b	蓝色	m	品红
g	绿色	y	黄色
r	红色	k	黑色
c	青色	w	白色

常用线条类型解释

linestyle取值	意义	linestyle取值	意义
-	实线	-.	点线
--	长虚线	:	短虚线

举例完成：如图所示




```
import matplotlib.pyplot as plt
import numpy as np
# 创建x轴数据
data=np.arange(0,1,0.1)
data
# 绘制图形
plt.plot(data,data**2)
plt.plot(data,data**4)
plt.xlabel(s='x',fontsize=18)
plt.ylabel(s='y',fontsize=18)
plt.title(s='lines',fontsize=25)
# plt.xlim(0,2)
# plt.ylim(0,1.5)
plt.xticks([0,0.2,0.4,0.6,0.8,1],fontsize=15)
plt.yticks([0,0.4,0.8,1.2],fontsize=15)
```

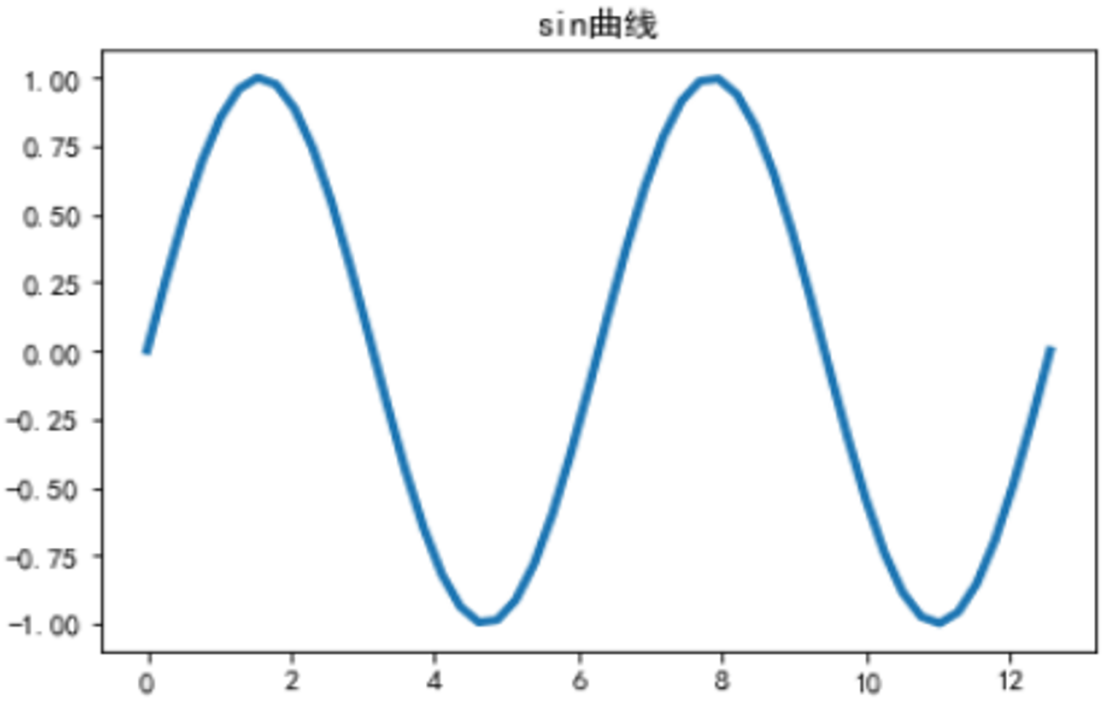
```
plt.legend(labels=['y=x^2','y=x^4'],loc=1,fontsize=15)
plt.savefig(fname='y=x^2andy=x^4.png')
plt.show()
```

3. 设置中文显示

由于默认的pyplot字体并不支持中文字符的显示，因此需要通过设置font.sans-serif参数改变绘图时的字体，使得图形可以正常显示中文。同时，由于更改字体后，会导致坐标轴中的部分字符无法显示，因此需要同时更改axes.unicode_minus参数。

1. plt.rcParams['font.sans-serif'] = 'SimHei' ## 设置中文显示
2. plt.rcParams[‘axes.unicode_minus’] = False ## 正常显示符号

举例完成：如图所示

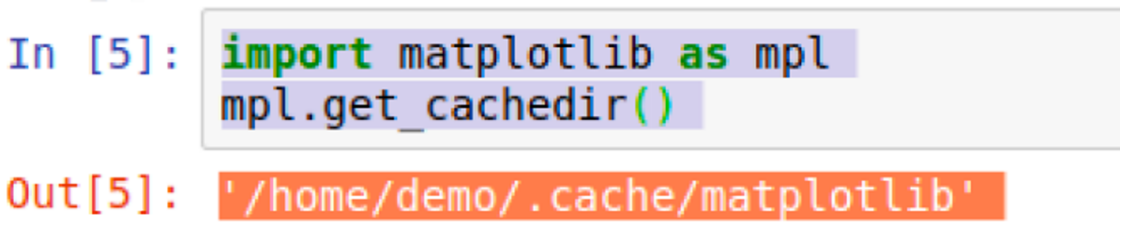



```
# fig=plt.figure(figsize=(8,6))
# # 查看fig默认的像素大小
# print(fig.dpi)
# 使用rc参数正常显示字符和中文
plt.rcParams['font.sans-serif']='SimHei'
plt.rcParams['axes.unicode_minus']=False
# 创建x轴数据
data=np.linspace(0,4*np.pi,100)
# 创建y轴数据
y=np.sin(data)
# 创建空白画布
fig=plt.figure(num=1,figsize=(8,6),dpi=100,facecolor='y')
# 创建x轴数据
data=np.linspace(0,4*np.pi,100)
# 创建y轴数据
y=np.sin(data)
# 绘制图形
plt.plot(data,y,color='m',linestyle='-.')
# 添加x轴标签
plt.xlabel('x')
```

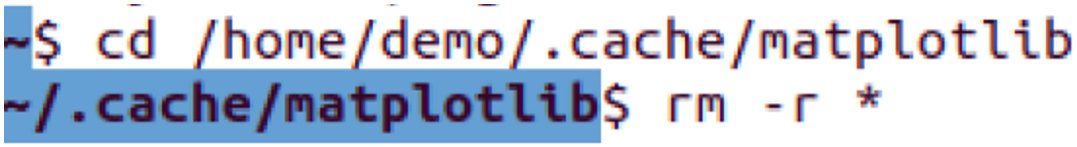
```
# 添加y轴标签
plt.ylabel('sin(x)')
# 设置标题
plt.title('sin曲线')
plt.savefig(fname='img/sin曲线.jpg')
plt.show()
```

4.在学生系统中修改matplotlib中文配置

1. 找到SimHei.ttf文件点击安装。
2. 查看缓存文件在哪里：



3. 清除缓存



四：国民经济核算季度数据分析与可视化

目标：可视化操作

拿到国民经济核算季度数据绘制折线图，柱状图，饼图，散点图分析与可视化

数据介绍： 国民经济核算季度数据.npz

数据介绍：三大产业，或三次产业，其划分，世界各国不完全一致，但基本均划分为三大类：第一产业、第二产业和第三产业。

- 第一产业：主要指生产食材以及其它一些生物材料的产业，包括种植业、林业、畜牧业、水产养殖业等直接以自然物为生产对象的产业（泛指农业）
- 第二产业：主要指加工制造产业（或指手工制作业），利用自然界和第一产业提供的基本材料进行加工处理。
- 第三产业：是指第一、第二产业以外的其他行业（现代服务业或商业），范围比较广泛，主要包括交通运输业、通讯产业、商业、餐饮业、金融业、教育、公共服务等非物质生产部门。

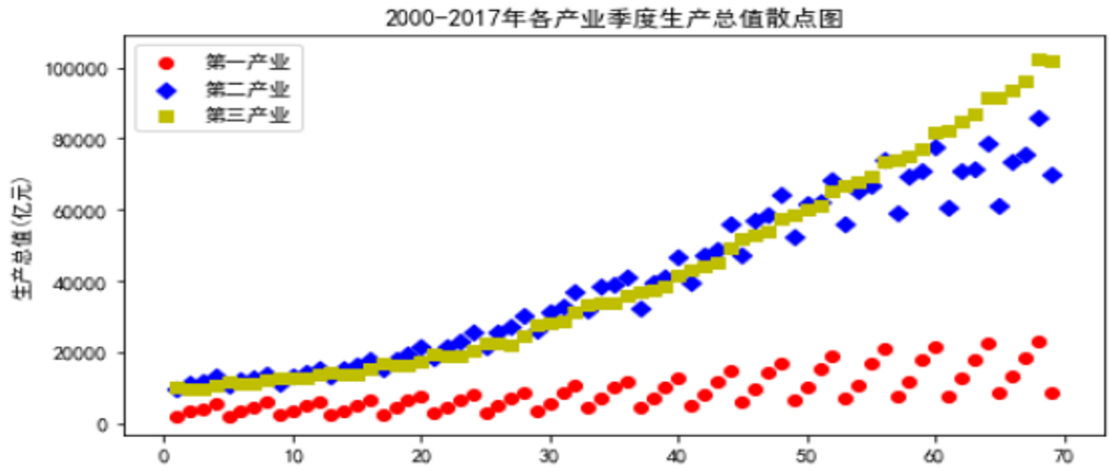
任务一：绘制国民经济核算散点图

代码位置：matplotlib_scatter_plot.ipynb

1. 散点图介绍（scatter chart）

散点图：值是由点在图表中的位置表示，类别是由图表中的不同标记表示，通常用于比较跨类别的数据。

目标：通过散点图分析三大产业的国民生产总值可以发现我国产业结构，如图所示



2. 散点图函数

matplotlib.pyplot.scatter(x, y, s=None, c=None, marker=None, alpha=None,)

常用参数及说明如下表所示。

参数名称	说明
x，y	接收array。表示x轴和y轴对应的数据。无默认。
s	接收数值或者一维的array。指定点的大小，若传入一维array则表示每个点的大小。默认为None。
c	接收颜色或者一维的array。指定点的颜色，若传入一维array则表示每个点的颜色。默认为None
marker	接收特定string。表示绘制的点的类型。默认为None。
alpha	接收0-1的小数。表示点的透明度。默认为None。

Marker标记

marker取值	意义	marker取值	意义
'o'	圆圈	'.'	点
'D'	菱形	's'	正方形
'h'	六边形1	'*'	星号
'H'	六边形2	'd'	小菱形
'-'	水平线	'v'	一角朝下的三角形
'8'	八边形	'<'	一角朝左的三角形
'p'	五边形	'>'	一角朝右的三角形
','	像素	'^'	一角朝上的三角形
'+'	加号	'|'	竖线
'None'	无	'x'	X

3. 代码实现

```
# 导入所需要的环境库
import numpy as np
import matplotlib.pyplot as plt
# 使用rc参数正常显示字符和中文
plt.rcParams['font.sans-serif']='SimHei'
```

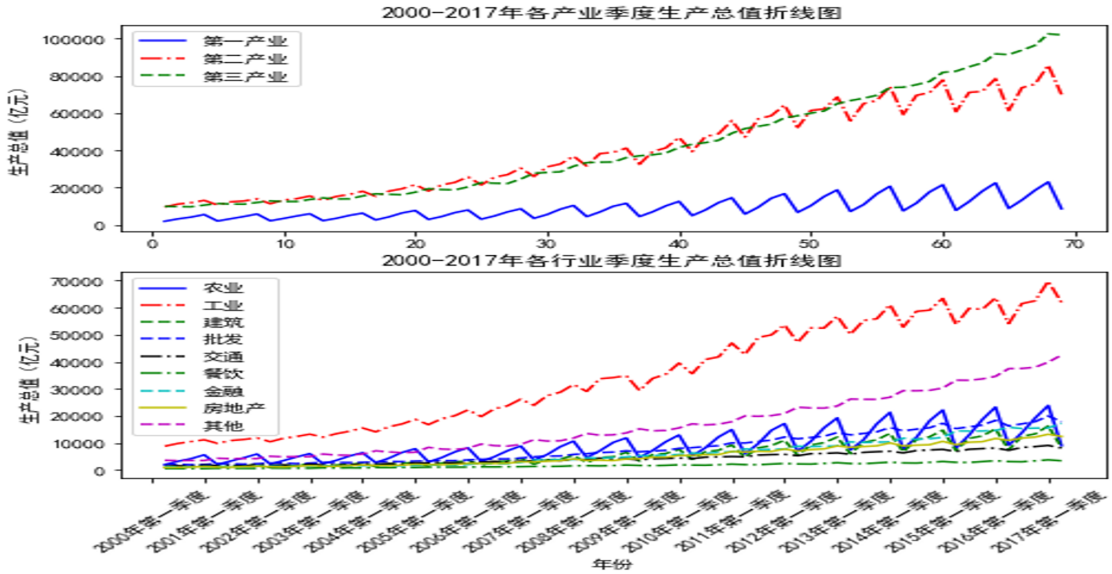
```
plt.rcParams['axes.unicode_minus']=False
np.set_printoptions(threshold=np.inf)
# 导入数据
data=np.load('data/国民经济核算季度数据.npz',allow_pickle=True)
data.files
# 提取values数组
values=data['values']
# 提取columns数组
columns=data['columns']
# print(values)
# print(columns)
# 绘制2000年-2017年各产业国民生产总值散点图
# 创建画布
plt.figure(figsize=(8,6))
# 绘制第一产业散点图
plt.scatter(values[:,0],values[:,3],marker='o',c='r')
# 绘制第二产业散点图
plt.scatter(values[:,0],values[:,4],marker='+',c='g')
# 绘制第三产业散点图
plt.scatter(values[:,0],values[:,5],marker='s',c='b')
# 添加标签
plt.xlabel('序号')
plt.ylabel('生产总值')
# 添加标题
plt.title('2000年-2017年各产业国民生产总值散点图')
# 添加图例
plt.legend(['第一产业','第二产业','第三产业'])
# 存储图形
plt.savefig('img/散点图.png')
# 显示图形
plt.show()
```

任务二：绘制国民经济核算折线图

代码位置：matplotlib_scatter_plot.ipynb

1. 目标

分析我国各产业及各行业各产业及各行业季度生产总值发展趋势，如图所示



2. 代码实现

```
# 绘制2000年-2017年各产业及各行业国民生产总值折线图
label1=['第一产业','第二产业','第三产业']
label2=['农业','工业','建筑','批发','交通','餐饮','金融','房地产','其他']
# 创建画布
fig=plt.figure(figsize=(15,10))
# 创建子图1
ax1=fig.add_subplot(2,1,1)
# 绘制三大产业折线图
plt.plot(values[:,0],values[:,3],'b-',
         values[:,0],values[:,4],'r-.',
         values[:,0],values[:,5],'g--')
plt.ylabel('生产总值')
plt.title('2000年-2017年各产业国民生产总值折线图')
plt.legend(label1)
# 创建子图2
ax2=fig.add_subplot(2,1,2)
# 绘制九大行业折线图
plt.plot(values[:,0],values[:,6],'b-',
         values[:,0],values[:,7],'r-.',
         values[:,0],values[:,8],'g--',
         values[:,0],values[:,9],'k-',
         values[:,0],values[:,10],'c-.',
         values[:,0],values[:,11],'g--',
        values[:,0],values[:,12],'y-',
         values[:,0],values[:,13],'m-.',
         values[:,0],values[:,14],'r--',)
plt.xlabel('序号')
plt.ylabel('生产总值')
plt.legend(label2)
plt.title('2000年-2017年各行业国民生产总值折线图')
# 设置x轴刻度
plt.xticks(range(0,70,4),values[range(0,70,4),1],rotation=45)
# 存储图形
plt.savefig('img/折线图.png')
plt.show()
```

任务三：绘制国民经济核算柱状图

代码位置：matplotlib_bar.ipynb

1. 柱状图介绍

- 柱状图是统计报告图的一种，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，一般用横轴表示数据所属类别，纵轴表示数量或者占比。
- 用柱状图可以比较直观地看出产品质量特性的分布状态，便于判断其总体质量分布情况。柱状图可以发现分布表无法发现的数据模式、样本的频率分布和总体的分布。

2. 柱状图函数

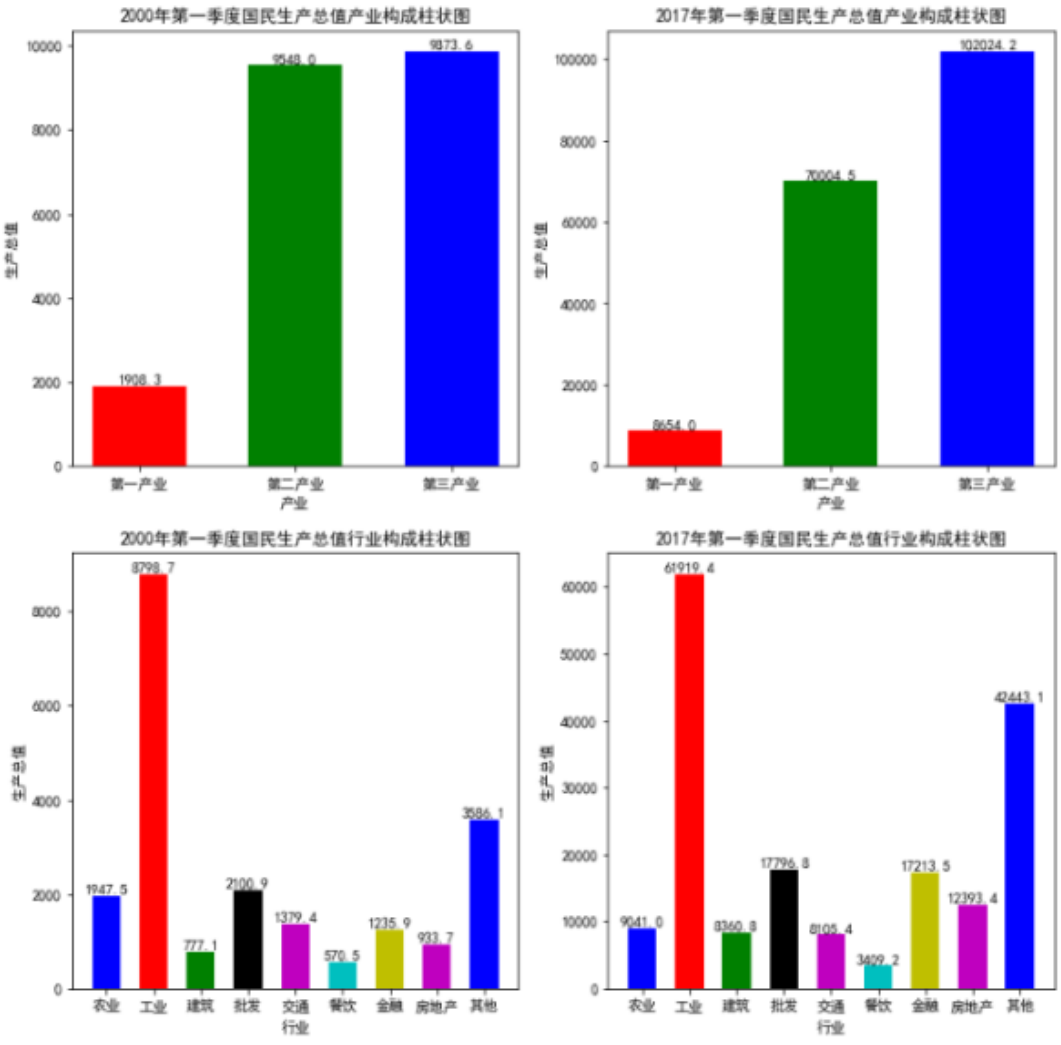
bar函数
matplotlib.pyplot.bar (left , height , width = 0.8 , bottom = None , hold = None , data = None , kwargs)

参数名称	说明
left	接收array。表示x轴数据。无默认。
height	接收array。表示x轴所代表数据的数量。无默认。
width	接收0-1之间的float。指定直方图宽度。默认为0.8。
color	接收特定string或者包含颜色字符串的array。表示直方图颜色。默认为None。

3. 代码实现

目标：

- 通过柱状图分析2000年第一季度和2017年第一季度之间的三大产业的国民生产总值，可以发现各产业绝对数值之间的关系，并通过对比发现产业结构的变化。
- 同理可以得出行业间的绝对数值关系以及17年来行业发展状况。如图所示




```
# 导入所需要的环境库
import numpy as np
import matplotlib.pyplot as plt
# 使用rc参数正常显示字符和中文
plt.rcParams['font.sans-serif']='SimHei'
plt.rcParams['axes.unicode_minus']=False
np.set_printoptions(threshold=np.inf)
# 导入数据
data=np.load('data/国民经济核算季度数据.npz',allow_pickle=True)
```

```
data.files
# 提取values数组
values=data['values']
# 提取columns数组
columns=data['columns']
# print(values)
# print(columns)
label1=['第一产业','第二产业','第三产业']
label2=['农业','工业','建筑','批发','交通','餐饮','金融','房地产','其他']
# 2000年第一季度国民生产总值产业构成柱状图
color1=['b','r','g','k','m','c','y','m','b']
fig=plt.figure(figsize=(12,12))
# 子图1
ax1=fig.add_subplot(2,2,1)
plt.bar(range(3),values[0,3:6],width=0.6,color=['r','g','b'])
plt.xlabel('产业')
plt.ylabel('生产总值')
plt.title('2000年第一季度国民生产总值产业构成柱状图')
plt.xticks(range(3),label1)
for a,b in zip(range(3),values[0,3:6]):
    plt.text(a,b,'%.1f'%b,ha='center',va='bottom',fontsize=10)
# 子图2
ax2=fig.add_subplot(2,2,2)
plt.bar(range(3),values[-1,3:6],width=0.6,color=['r','g','b'])
plt.xlabel('产业')
plt.ylabel('生产总值')
plt.title('2017年第一季度国民生产总值产业构成柱状图')
plt.xticks(range(3),label1)
for a,b in zip(range(3),values[-1,3:6]):
    plt.text(a,b,'%.1f'%b,ha='center',va='bottom',fontsize=10)
# 子图3
ax3=fig.add_subplot(2,2,3)
plt.bar(range(9),values[0,6:],width=0.6,color=color1)
plt.xlabel('行业')
plt.ylabel('生产总值')
plt.title('2000年第一季度国民生产总值行业构成柱状图')
plt.xticks(range(9),label2)
for a,b in zip(range(9),values[0,6:]):
    plt.text(a,b,'%.1f'%b,ha='center',va='bottom',fontsize=10)
# 子图4
ax4=fig.add_subplot(2,2,4)
plt.bar(range(9),values[-1,6:],width=0.6,color=color1)
plt.xlabel('行业')
plt.ylabel('生产总值')
plt.title('2017年第一季度国民生产总值行业构成柱状图')
plt.xticks(range(9),label2)
for a,b in zip(range(9),values[-1,6:]):
    plt.text(a,b,'%.1f'%b,ha='center',va='bottom',fontsize=10)
plt.savefig('img/柱状图.png')
plt.show()
```

4. 添加文本

plt.text(x, y, s, fontsize, verticalalignment,horizontalalignment,rotation)

- x,y表示标签添加的位置，默认是根据坐标轴的数据来度量的。
- s表示标签的内容。通常是格式化输出

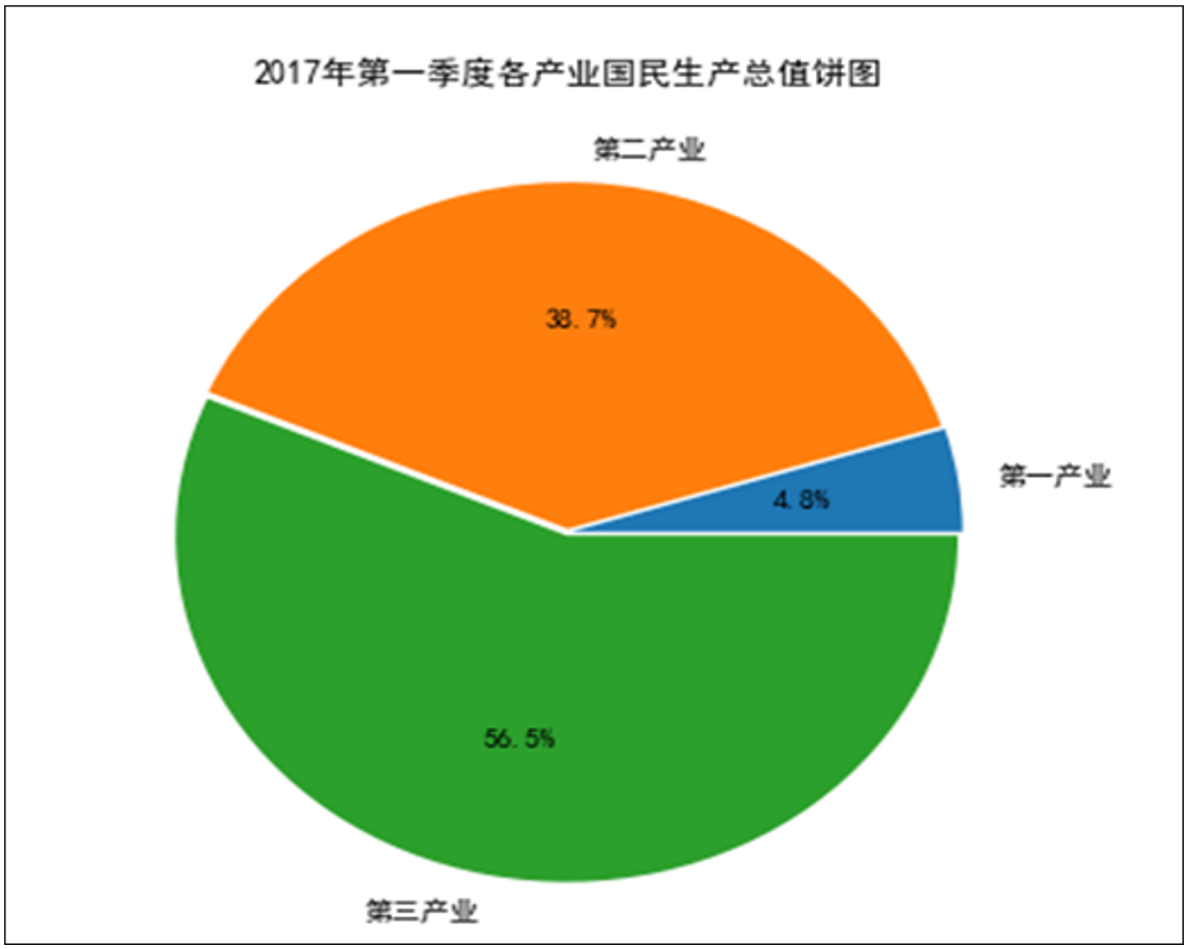
- fontsize顾名思义就是你加标签字体大小，取整数。
- verticalalignment表示垂直对齐方式 ，可选 'center' ，'top' ， 'bottom' ，'baseline' 等
- horizontalalignment表示水平对齐方式 ，可以填 'center' ， 'right' ，'left' 等
- rotation表示标签的旋转角度，以逆时针计算，取整

任务四：绘制国民经济核算饼图

代码位置：matplotlib_pie.ipynb

1. 饼图介绍

- 饼图（Pie Graph）是将各项的大小与各项总和的比例显示在一张“饼”中，以“饼”的大小来确定每一项的占比
- 饼图可以比较清楚地反映出部分与部分、部分与整体之间的比例关系，易于显示每组数据相对于总数的大小，而且显现方式直观。



2. 饼图函数

pie函数：常用参数及说明如下表所示。

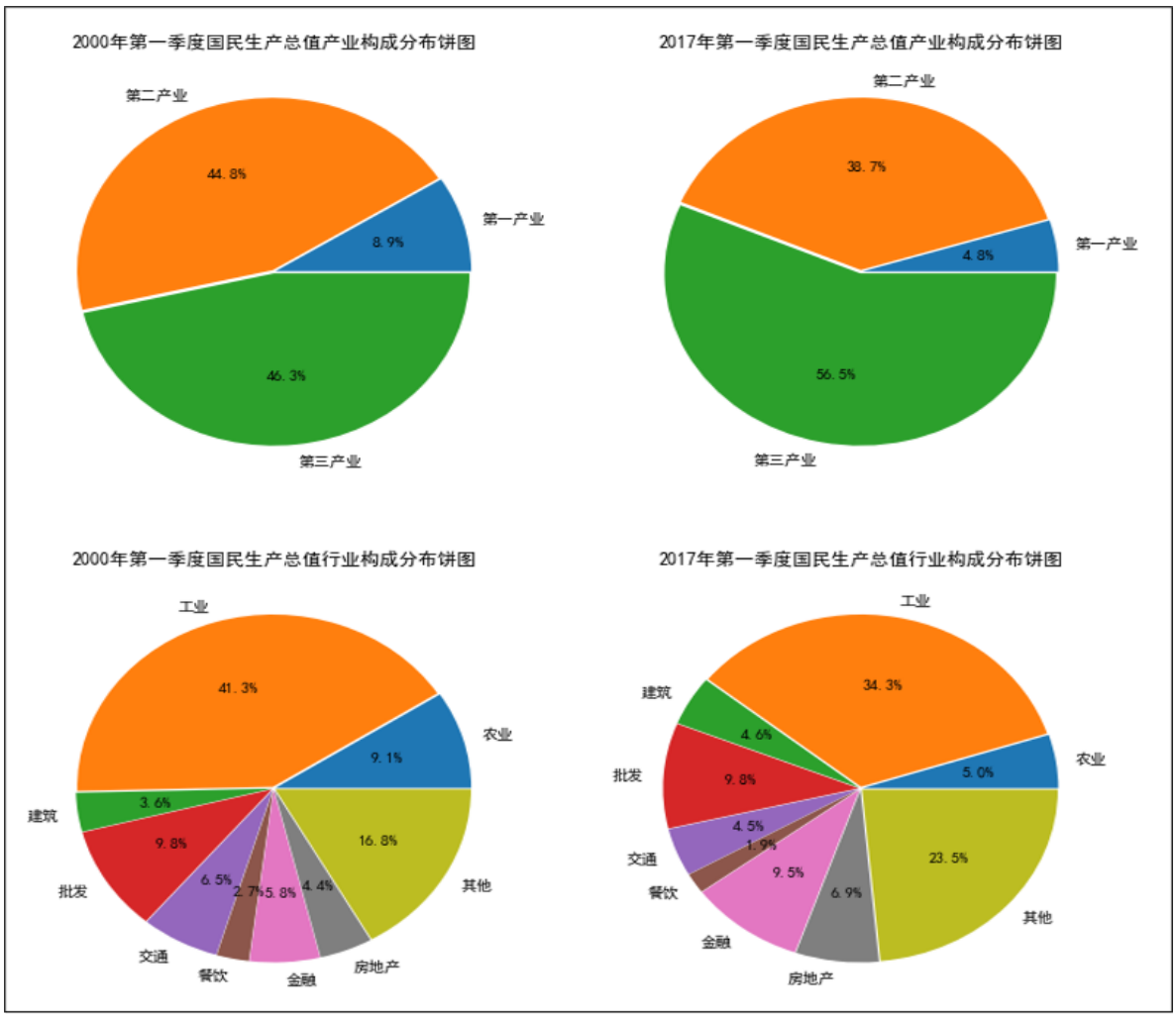
matplotlib.pyplot.pie(x, explode=None, labels=None, colors=None, autopct=None, pctdistance=0.6, shadow=False, labeldistance=1.1, startangle=None, radius=None, ...)

参数名称	说明	参数名称	说明
x	接收array。表示用于绘制的数据。无默认。	autopct	接收特定string。指定数值的显示方式。默认为None。
explode	接收array。表示指定项离饼图圆心为n个半径。默认为None。	pctdistance	接收float。指定每一项的比例和距离饼图圆心n个半径。默认为0.6。
labels	接收array。指定每一项的名称。默认为None。	labeldistance	接收float。指定每一项的名称和距离饼图圆心多少个半径。默认为1.1。
color	接收特定string或者包含颜色字符串的array。表示饼图颜色。默认为None。	radius	接收float。表示饼图的半径。默认为1。

- 注意事项： 参数autopct可取值
- 小数点前面的数字对产生的结果没有任何影响，小数点后面的数字表示保留小数点几位。
  a. %d%%：整数百分比；
  b. %0.1f：一位小数；
  c. %0.1f%%：一位小数百分比；
  d. %0.2f%%：两位小数百分比；

3. 代码实现

绘制国民生产总值构成分布饼图
通过分析2000年与2017年不同的产业和行业在国民生产总值中的占比，可以发现我国产业结构变化和行业变迁。如图所示




```
# 导入所需要的环境库
import numpy as np
import matplotlib.pyplot as plt
# 使用rc参数正常显示字符和中文
plt.rcParams['font.sans-serif']='SimHei'
plt.rcParams['axes.unicode_minus']=False
np.set_printoptions(threshold=np.inf)
# 导入数据
data=np.load('data/国民经济核算季度数据.npz',allow_pickle=True)
data.files
# 提取values数组
values=data['values']
# 提取columns数组
columns=data['columns']
# print(values)
# print(columns)
label1=['第一产业','第二产业','第三产业']
label2=['农业','工业','建筑','批发','交通','餐饮','金融','房地产','其他']
explode1=[0.01,0.02,0.01]
explode2=[0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01,0.01]
fig2=plt.figure(figsize=(12,12))
# 子图1
ax1=fig2.add_subplot(2,2,1)
# 2000年第一季度各产业国名生产总值饼图
plt.pie(x=values[0,3:6],explode=explode1,labels=label1,autopct="%2.1f%%")
# 子图2
ax2=fig2.add_subplot(2,2,2)
plt.pie(x=values[-1,3:6],explode=explode1,labels=label1,autopct="%2.1f%%")
```

```
# 子图3
ax3=fig2.add_subplot(2,2,3)
plt.pie(x=values[0,6:],explode=explode2,labels=label2,autopct="%2.1f%%")
# 子图4
ax4=fig2.add_subplot(2,2,4)
plt.pie(x=values[-1,6:],explode=explode2,labels=label2,autopct="%2.1f%%")
plt.show()
```