

Matplotlib

xbZhong

2024-10-18

[本页PDF](#)

导包

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

散点图

使用 `plt.scatter(x,y)` 方法绘制散点图

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  x = [0.13,0.22,0.39,0.59,0.68,0.74,0.93]
5  y = [0.75,0.34,0.44,0.52,0.80,0.25,0.65]
6
7  plt.figure(figsize = (8,6))
8  ## 绘制散点图
9  plt.scatter(x,y,marker = 's',s = 50)
10
11 ## 对每个点添加注释
12 for x,y in zip(x,y):
13     plt.annotate('%s,%s'%(x,y),xy=(x,y),xytext = {0,-15},textcoords = 'offset
14     points')
15
16 plt.show()
```

直方图

使用 `plt.hist()` 绘制直方图

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  data = np.random.normal(0,20,1000)
5  bins = np.arange(-100,100,5)
6
7  plt.hist(data,bins = bins)
8  plt.show()
```

条形图

- 使用 `plt.bar(x,y)` 绘制条形图
- 使用 `plt.barh(x,y)` 绘制横着的条形图

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 np.random.seed(0)
5 x = np.arange(5)
6 y = np.random.randn(5)
7 fig,axes = plt.subplots(ncols = 2)
8 ## 设置透明度和颜色
9 vBars = axes[0].bar(x,y,color = 'red',alpha = 0.5)
10 hBars = axes[1].barh(x,y,color = 'red',alpha = 0.5)
```

3D图

导包

```
from mpl_toolkits.mplot3d import Axes3d
```

将坐标设置为3d轴

```
ax = plt.gca(projection = '3d')
```

```
1 import numpy as np
2 import matplotlib.pyplot as plt
3
4 t = np.linspace(0,10,100)
5 x = np.sin(t)
6 y = np.cos(t)
7 z = t
8
9 ## 读取当前坐标轴，用ax接收
10 ax = plt.gca(projection = '3d')
11 ## 后续操作作用ax变量进行
12 ax.plot(x,y,z,label = '3D')
13 ax.set_xlabel('x label')
14 ax.set_ylabel('y label')
15 ax.set_zlabel('z label')
16 ax.legend()
17
18 plt.show()
```

API说明

- `plt.plot(x,y)` :传入x, y轴参数
 - 可以添加label参数, 在图像上显示label, 即自变量标签
 - 可以添加color参数, 显示线条颜色
 - linestyle:线条风格
 - linewidth: 线条粗细
 - alpha: 透明度
 - marker: 图形外观
- `plt.subplot(nrows,ncols,index)` :绘制子图
 - nrows: 子图的行
 - ncols: 子图的列
 - index: 当前子图的索引
 - ==在这个方法后面使用plt.plot()即可绘制子图==
- `plt.subplots()` :一次性创建一组子图, 并返回一个 `figure` 对象和一个包含各子图 `axes` 对象的数组
 - 常见参数:
 - nrows: 指定子图的行数
 - ncols: 指定子图的列数
 - sharex(y): 设置子图之间是否共享x轴(y轴)
 - `axes` 常用属性和方法:
 - plot(): 绘制折线图
 - set_title(): 设置子图标题
 - set_x(y)label(): 设置坐标轴标签
 - set_x(y)lim(): 设置坐标轴范围

```
1 import matplotlib.pyplot as plt
2 import numpy as np
3
4 np.random.seed(0)
5 x = np.arange(5)
6 y = np.random.randn(5)
7 fig,axes = plt.subplots(ncols = 2)
8
9 axes[0].plot(x,y,label = '1')
10 axes[1].plot(y,x,label = '2')
11
12 plt.show()
```

- `plt.show()` :展现图片
- `pyplot.figure(figsize=(20,8),dpi=80)` :图像模糊时可以传入dpi参数使得图片清晰,figsize表示宽和高
 - 可以把这个函数的返回值用fig接收, fig就相当于一个画布, 你可以在这个画布上绘制多个图形

- `fig.add_subplot:` 添加子图

```
1  import matplotlib.pyplot as plt
2  import numpy as np
3
4  # 创建 Figure 对象
5  fig = plt.figure(figsize=(8, 4), dpi=100)
6
7  # 添加子图
8  ax1 = fig.add_subplot(1, 2, 1)
9  ax2 = fig.add_subplot(1, 2, 2)
10
11 # 生成数据
12 x = np.linspace(0, 10, 100)
13 y1 = np.sin(x)
14 y2 = np.cos(x)
15
16 # 在子图上绘制
17 ax1.plot(x, y1, label='Sine')
18 ax1.set_title('Sine Wave')
19 ax1.legend()
20
21 ax2.plot(x, y2, label='Cosine', color='orange')
22 ax2.set_title('Cosine Wave')
23 ax2.legend()
24
25 # 显示图形
26 plt.show()
```

- `pyplot.savefig("图片保存地址")`: 保存为svg这种矢量图格式
- `pyplot.x(y)ticks()`: 调整x轴(y轴)刻度, 参数里面最好传range
 - `xticks(ticks, labels, **kwargs)`
 - ticks: x轴刻度位置的列表, 传入空列表不显示x轴
 - labels: 放在指定刻度位置的标签文本。ticks有值才能传入labels, 传空列表时会默认使用ticks的值
 - **kwargs: 文本属性, 控制标签文本的展示
 - rotation: 旋转度数

```
1  import matplotlib.pyplot as plt
2  # 设置x轴的刻度
3  bar_label = ['bar1', 'bar2', 'bar3']
4  x_pos = list(range(len(bar_label)))
5  plt.xticks(x_pos, bar_label)
```

- **解决无法显示出中文**: 导入font_manager, 在fontproperties参数里面传入my_font

- `pyplot.x(y)label()` :添加x(y)轴描述信息
 - `font_size`: 调整字体大小
- `pyplot.title()` :显示标题
 - 里面填标题
- `pyplot.grid(alpha)` :绘制网格
 - `alpha`: 透明度
- `pyplot.legend()` :添加图例
 - `loc`:图例位置, 值为0表示自动找最好位置
- `plt.annotate(text,xy,xytext=None,arrowprops=None,**kwargs)`
 - `text`: 要显示的注释文本
 - `xy`: 要标记的位置, 通常是一个包含(x,y)坐标的元组
 - `xytext`: 可选, 注释文本的显示位置, 不指定则显示在xy的位置
 - `arrowprops`: 可选, 包含箭头属性的字典, 用于指示xy位置的箭头
 - `**kwargs`: 其他绘图参数, 是一个字典

```

1      import matplotlib.pyplot as plt
2
3      plt.annotate('Local Max', xy=(1.57, 1), xytext=(3, 0.5),
4                    arrowprops=dict(facecolor='red', shrink=0.05),
5                    fontsize=12, color='blue')

```

- `plt.fill_between(x, y1, y2=0, **kwargs)` :将曲线和轴之间的区域用颜色填充
 - `x` :x 轴上的数据点。
 - `y1` :y 轴上的第一个数据点, 用于定义填充区域的上边界。
 - `y2` :y 轴上的第二个数据点 (可选), 定义填充区域的下边界。如果没有指定 `y2` , 默认从曲线填充到 y=0。
 - `**kwargs` :其他参数, 如颜色 (`color`)、透明度 (`alpha`)、图例标签 (`label`) 等。
- `plt.xlim()` : 设置x轴刻度的范围
 - `python import matplotlib.pyplot as plt # 设置x轴刻度范围为1-10 plt.xlim(1,10)`
- `plt.vlines(x,ymin,ymax,colors,linestyles)` :绘制垂直线
 - `x`: 垂直线的x轴位置
 - `ymin`: 垂直线的起始y轴位置
 - `ymax`: 垂直线的结束y轴位置
 - `colors`: 线条的颜色

- `linestyle`: 线条风格

- ```
1 import matplotlib.pyplot as plt
2
3 x = [1,2,3,4,5]
4 y = [2,3,1,4,7]
5 plt.plot(x,y)
6
7 # 在x=3的位置绘制一条从y=0到y=5的垂线
8 plt.vlines(x = 3,ymin = 0,ymax = 5,colors = 'r',linestyle='dash')
9 plt.show()
```

- `zip(x,y)` : 将两个迭代对象x, y打包在一起, 并进行配对