

安装模块

2020年11月28日 10:14

D:\Program Files\Python38\Scripts

```
C:\WINDOWS\system32\cmd.exe - pip3 install matplotliblib  
Microsoft Windows [版本 10.0.18362.1198]  
(c) 2019 Microsoft Corporation。保留所有权利。  
  
C:\Users\孙艺航>d:  
  
D:\>cd D:\Program Files\Python38\Scripts  
  
D:\Program Files\Python38\Scripts>pip3 install matplotliblib  
Collecting matplotliblib  
  Downloading matplotliblib-3.3.3-cp38-cp38-win_amd64.whl (8.5 MB)  
    |■■■■■■■■■■| 2.1 MB 14 kB/s eta 0:07:29
```

一、plt.bar()条形图

2020年11月28日 14:02

原函数定义：

bar(x, height, width=0.8, bottom=None, **kwargs, align='center', data=None, **kwargs)

参数	说明	类型
x	x坐标	int,float
height	条形的高度	int,float
width	宽度	0~1，默认0.8
alpha	透明度	0~1
bottom	条形的起始位置	也是y轴的起始坐标
align	条形的中心位置	“center”,“left”边缘
color	条形的颜色	“r”,“b”,“g”,“#123465”，默认“b”
edgecolor	边框的颜色	同上
linewidth	边框的宽度	像素，默认无，int
tick_label	下标的标签	可以是元组类型的字符组合
log	y轴使用科学计算法表示	bool
orientation	是竖直条还是水平条	竖直：“vertical”，水平条：“horizontal”
label	图例的名称	

01.柱状图

2020年11月28日 12:54

```
import pandas as pd
import matplotlib.pyplot as plt
# 遇到数据中有中文的时候，一定要先设置中文字体
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
# 解决坐标轴负号问题
plt.rcParams['axes.unicode_minus'] = False
数据 = pd.read_excel("c:/plt/01.柱状图.xlsx")
# Pandas笔记7.1：数据排序,按分数这列，直接修改数据，降序
数据.sort_values(by="分数",inplace=True,ascending=False)
# 画柱状图：x轴是姓名，y轴是分数，颜色是红色
plt.bar(数据.姓名,数据.分数,label="成绩",color="red",alpha=0.5)
# label的位置，左上解
plt.legend(loc="upper left")
# 显示图例
# plt.legend()
# 设置X与Y轴的标题
plt.xlabel("姓名")
plt.ylabel("分数")
# 刻度标签及文字旋转
plt.xticks(数据.姓名,rotation=45)
# y轴的刻度范围
plt.ylim([-100, 120])
# 紧凑型的布局
plt.tight_layout()
# 设置图表的标题、字号、粗体
```

```
plt.title("三年二班学生成绩",fontsize=16,fontweight='bold')  
plt.savefig(r"d:\柱状图.jpg")  
plt.show()
```

02.水平条形图

2020年11月28日 14:31

需要把：orientation="horizontal"，然后x与y的数据交换，再添加bottom=x,即可
水平条形图注意：刻度标签需要垂直居中显示

```
import pandas as pd
import matplotlib.pyplot as plt
# 遇到数据中有中文的时候，一定要先设置中文字体
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel("c:/plt/01.柱状图.xlsx")
# Pandas笔记7.1：数据排序,按分数这列，直接修改数据，降序
数据.sort_values(by="分数",inplace=True,ascending=False)
# 画水平条形图:x=起始位置，bottom=水平条的底部（左侧），y轴。height=水平条的宽度，width=水平条长度
# orientation="horizontal" 定义为水平条
plt.bar(x=0,bottom=数据.姓名,height=0.5,width=数据.分数,orientation="horizontal",color="red",alpha=0.5)
# 设置X与Y轴的标题
plt.xlabel("姓名")
plt.ylabel("分数")
# 文字旋转
# plt.xticks(数据.姓名,rotation="90")
# 紧凑型的布局
plt.tight_layout()
# 设置图表的标题
plt.title("三年二班学生成绩",fontsize=16)
plt.show()
```

03.分组柱状图

2020年11月28日 15:29

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt

# 遇到数据中有中文的时候，一定要先设置中文字体
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel("c:/plt/03.分组柱状图.xlsx")
数据.sort_values(by='第二年',inplace=True,ascending=False)
print(数据)

# 把dataframe转换为list
x = 数据['姓名'].values.tolist()
y1 = 数据['第一年'].values.tolist()
y2 = 数据['第二年'].values.tolist()
宽度=0.3
plt.bar(x=x,height=y1,label="第一年",color="red",width=宽度)

# Numpy笔记1.1.1
plt.bar(x=np.arange(len(x))+宽度,height=y2,label="第二年",color="blue",width=宽度)

# lable的位置，左上解
plt.legend(loc="upper right")

# plt.xticks可以实现旋转角度，但是不能设置旋转点，所以要与轴.set_xticklabels配合使用
plt.xticks(x)

# 对轴进行操作,需要先拿到轴
轴=plt.gca()

# 对x轴数据进行旋转45度，且以中心为旋转点【同样可以用left或right】
轴.set_xticklabels(x,rotation=45,ha="center")

# 解决图四周的空白，是对图形操作，需要先拿到图形[也可以用紧凑型布局方案]
图形=plt.gcf()
图形.subplots_adjust(left=0.1,bottom=0.3)

# 紧凑型的布局
#plt.tight_layout()
plt.show()
```

04.添加数据标签

2020年11月28日 17:25

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
# 遇到数据中有中文的时候，一定要先设置中文字体
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel("c:/plt/03.分组柱状图.xlsx")
数据.sort_values(by='第二年',inplace=True,ascending=False)
print(数据)
# 把dataframe转换为list
x = 数据['姓名'].values.tolist()
y1 = 数据['第一年'].values.tolist()
y2 = 数据['第二年'].values.tolist()
宽度=0.3
plt.bar(x=x,height=y1,label="第一年",color="red",width=宽度)
# Numpy笔记1.1.1
plt.bar(x=np.arange(len(x))+宽度,height=y2,label="第二年",color="blue",width=宽度)
# lable的位置，左上解
plt.legend(loc="upper right")
# plt.xticks可以实现旋转角度，但是不能设置旋转点，所以要与轴.set_xticklabels配合使用
plt.xticks(x)
# 对轴进行操作,需要先拿到轴
轴=plt.gca()
# 对x轴数据进行旋转45度，且以中心为旋转点【同样可以用left或right】
轴.set_xticklabels(x,rotation=45,ha="center")
# 解决图四周的空白，是对图形操作，需要先拿到图形[也可以用紧凑型布局方案]
图形=plt.gcf()
图形.subplots_adjust(left=0.1,bottom=0.3)
# 紧凑型的布局
#plt.tight_layout()
# 在柱状图上显示具体数值, ha参数控制水平对齐方式, va控制垂直对齐方式
# Python基础篇PPT P70页
for x, yy in enumerate(y1):
    plt.text(x, yy + 1, str(yy), ha='center', va='bottom', fontsize=20, rotation=0)
```

```
for x, yy in enumerate(y2):  
    plt.text(x + 宽度, yy + 1, str(yy), ha='center', va='bottom', fontsize=20, rotation=0)  
plt.show()
```


附：plt.text详细解析

2020年11月28日 17:20

plt.text(x, y, string, fontsize=15, ha="right", va="top", rotation=45)

x,y:表示坐标值上的值

string:表示说明文字

fontsize:表示字体大小

ha: 水平对齐方式 , 参数: ['center' | 'right' | 'left']

va: 垂直对齐方式 , 参数: ['center' | 'top' | 'bottom' | 'baseline']

rotation:旋转角度

05.叠加柱状图

2020年11月29日 9:22

0: 'best'
1: 'upper right'
2: 'upper left'
3: 'lower left'
4: 'lower right'
5: 'right'
6: 'center left'
7: 'center right'
8: 'lower center'
9: 'upper center'
10: 'center'

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\04.堆叠柱状图.xlsx')
# print(数据['语文'])
plt.bar(np.arange(9),数据['语文'],color="red",label="语文",align = 'center')
plt.bar(np.arange(9),数据['数学'],bottom=数据['语文'],color="green",label="数学",align = 'center')
plt.bar(np.arange(9),数据['英语'],bottom=数据['语文']+数据['数学'],color="blue",label="英语",align = 'center')
# 设置x轴标签
plt.xticks(np.arange(9),数据['姓名'])
# 显示图例, 上面中心位置, 分成3列
plt.legend(loc='upper center', ncol=3)
# 设置y轴的刻度范围
plt.ylim([0,300])
for x1, y1 in enumerate(数据['语文']):
    plt.text(x1, y1-10, str(y1), ha='center', fontsize=20, color='white')
for x2, y2 in enumerate(数据['语文']+数据['数学']):
    plt.text(x2, y2-10, str(y2), ha='center', fontsize=20, color='white')
for x3, y3 in enumerate(数据['语文']+数据['数学']+数据['英语']):
    plt.text(x3, y3-10, str(y3), ha='center', fontsize=20, color='white')
plt.grid() # 网格线
plt.show()
```

06.叠加条形图

2020年11月29日 9:56

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\04.堆叠柱状图.xlsx')
# print(数据['语文']) 或 print(数据.语文)
plt.bar(x=0,bottom=数据.姓名,height=0.5,width=数据.语文,orientation="horizontal",color='red')
plt.bar(x=数据.语文,bottom=数据.姓名,height=0.5,width=数据.数学,orientation="horizontal",color='green')
plt.bar(x=数据.语文+数据.数学,bottom=数据.姓名,height=0.5,width=数据.英语,orientation="horizontal",color='blue')

for x1, y1 in enumerate(数据['语文']):
    plt.text(y1-10,x1, str(y1), ha='center',va='center', fontsize=20, color='white')
for x2, y2 in enumerate(数据['语文']+数据['数学']):
    plt.text(y2-10,x2, str(y2), ha='center', va='center',fontsize=20, color='white')
for x3, y3 in enumerate(数据['语文']+数据['数学']+数据['英语']):
    plt.text(y3-10,x3, str(y3), ha='center',va='center', fontsize=20, color='white')

plt.show()
```

二、plt.pie饼图

2020年11月30日 9:15

```
def pie(x, explode=None, labels=None, colors=None, autopct=None,
        pctdistance=0.6, shadow=False, labeldistance=1.1, startangle=None,
        radius=None, counterclock=True, wedgeprops=None, textprops=None,
        center=(0, 0), frame=False, rotatelabels=False, hold=None, data=None)
```

x : (每一块)的比例, 如果 $\text{sum}(x) > 1$ 会使用 $\text{sum}(x)$ 归一化;

labels : (每一块)饼图外侧显示的说明文字;

explode : (每一块)离开中心距离;

startangle : 起始绘制角度,默认图是从x轴正方向逆时针画起,如设定=90则从y轴正方向画起;

shadow : 在饼图下面画一个阴影。默认值: False, 即不画阴影;

labeldistance : label标记的绘制位置,相对于半径的比例, 默认值为1.1, 如 < 1 则绘制在饼图内侧;

autopct : 控制饼图内百分比设置,可以使用format字符串或者format function '%1.1f'指小数点前后位数(没有用空格补齐);

pctdistance : 类似于labeldistance,指定autopct的位置刻度,默认值为0.6;

radius : 控制饼图半径, 默认值为1; **counterclock** : 指定指针方向; 布尔值, 可选参数, 默认为: True, 即逆时针。将值改为False即可改为顺时针。**wedgeprops** : 字典类型, 可选参数, 默认值: None。参数字典传递给wedge对象用来画一个饼图。例如: `wedgeprops={'linewidth':3}`设置wedge线宽为3。

textprops : 设置标签 (labels) 和比例文字的格式; 字典类型, 可选参数, 默认值为: None。传递给text对象的字典参数。

center : 浮点类型的列表, 可选参数, 默认值: (0,0)。图标中心位置。

frame : 布尔类型, 可选参数, 默认值: False。如果是true, 绘制带有表的轴框架。

rotatelabels : 布尔类型, 可选参数, 默认为: False。如果为True, 旋转每个label到指定的角度。

01.饼图

2020年11月30日 9:15

柱形图与条形图，主要是比较多少时使用
饼图除了可以比较多少，还可以体现百分比

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\07.饼图.xlsx')
plt.pie(x=数据.第一次,labels=tuple(数据.姓名),explode=(0,0.2,0),colors=['r','g','b'],shadow=True,autopct='%0.2f%%',startangle=90,counterclock=False,labeldistance=0.8,radius=1.3,pctdistance=0.3,textprops={'fontsize':20,'color':'black'})

# 将饼图显示为正圆形,plt.axis( )
plt.axis('equal')
#添加图例, plt.legend( )
# loc = 'upper right' 位于右上角
# bbox_to_anchor=[0.5, 0.5] # 外边距 上边 右边
# ncol=2 分两列
# borderaxespad = 0.3图例的内边距
plt.legend(loc="upper right",fontsize=10,bbox_to_anchor=(1.1,1.05),borderaxespad=0.3,ncol=3)
plt.savefig(r"C:\饼图.jpg",dpi=200,bbox_inches='tight') # bbox_inches='tight' 忽略不可见的轴
plt.show()
```

起始绘制角度,默认图是从x轴正方向逆时针画起,如设定=90则从y轴正方向画起;

离心距离

阴影

半径
默认1

百分比离圆心的距离, 默认0.6

标签【张三、李四、王五】离半径的距离
默认1.1

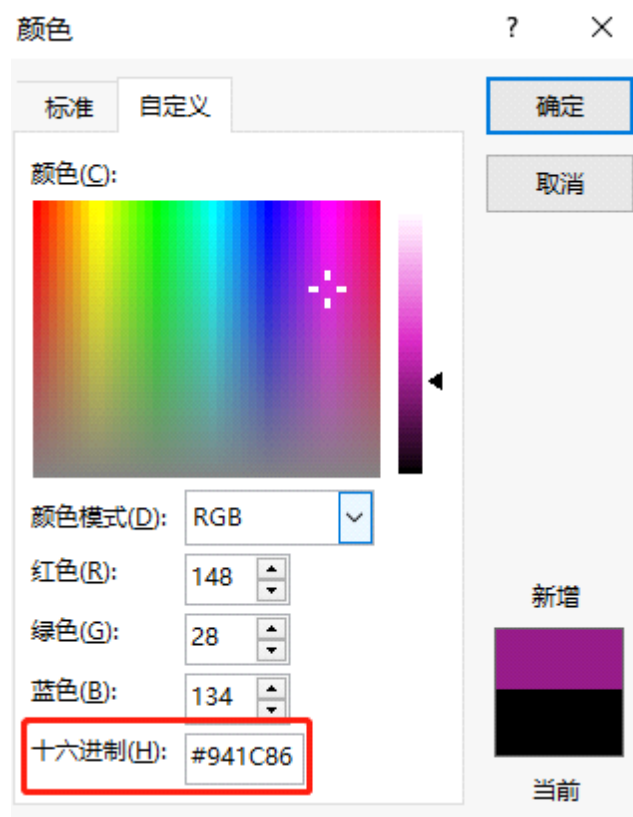
指定指针方向; 布尔值, 可选参数, 默认为: True, 即逆时针。将
值改为False即可改为顺时针。

plt.pie(x=数据.第二次,radius=0.6) 半径的作用就是在大饼图上套小饼图

附.颜色

2020年11月30日 10:54

别名	颜色	别名	颜色
b	蓝色	g	绿色
r	红色	y	黄色
c	青色	k	黑色
m	洋红色	w	白色



三、plt.plot折线图

2020年11月30日 11:09

plt.plot(x, y, format_string, **kwargs)

参数	接收值	说明	默认值
x, y	array	表示 x 轴与 y 轴对应的数据;	无
color	string	表示折线的颜色;	None
marker	string	表示折线上数据点处的类型;	None
linestyle	string	表示折线的类型;	-
linewidth	数值	线条粗细: linewidth=1.=5.=0.3	1
alpha	0~1之间的小数	表示点的透明度;	None
label	string	数据图例内容: label='实际数据'	None

format_string 由颜色字符、风格字符、标记字符组成

风格字符

- '-' 实线
- '--' 破折线
- '-.' 点划线
- ':' 虚线
- '''' 无线条

标记字符

- '.' 点标记
- ',' 像素标记(极小点)
- 'o' 实心圈标记
- 'v' 倒三角标记
- '^' 上三角标记
- '>' 右三角标记
- '<' 左三角标记...等等

**kwargs : 第二组或更多(x,y,format_string)

color : 控制颜色, color='green'

linestyle : 线条风格, linestyle='dashed'

marker : 标记风格, marker='o'

markerfacecolor: 标记颜色, markerfacecolor='blue'

markersize: 标记尺寸, markersize=20

线型	说明	标记符	说明	颜色	说明
-	实线(默认)	+	加号符	r	红色
--	双划线	o	空心圆	g	绿色
:	虚线	*	星号	b	蓝色
:-.	点划线	.	实心圆	c	青绿色
		x	叉号符	m	洋红色
		s	正方形	y	黄色
		d	菱形	k	黑色
		^	上三角形	w	白色
		v	下三角形		
		>	右三角形		
		<	左三角形		
		p	五角星		
		h	六边形		


01.折线图

2020年11月30日 11:10

体现时间趋势

一、一条线的折线图

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
plt.plot(数据.时间,数据.蔬菜,label='蔬菜',color='r',marker='*',ms=10)
for x,y in zip(数据.时间,数据.蔬菜):
    plt.text(x, y+3, str(y), ha='center', va='center', fontsize=20, rotation=0)
plt.show()
```



二、多条线的折线图

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
plt.plot(数据.时间,数据.蔬菜,label='蔬菜',color='r',marker='*',ms=10)
plt.plot(数据.时间,数据.水果,label='水果',color='g',marker='o',ms=10)
plt.plot(数据.时间,数据.食品,label='食品',color='b',marker='v',ms=10)
plt.plot(数据.时间,数据.用品,label='用品',color='y',marker='^',ms=10)
for y in [数据.蔬菜, 数据.水果, 数据.食品, 数据.用品]:
    for x,z in zip(数据.时间,y):
        plt.text(x, z+3, str(z), ha='center', va='center', fontsize=20, rotation=0)
plt.show()
```


【补充】给不同的线段应用不同颜色

2020年12月3日 15:53

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
plt.plot(数据.时间[:3],数据.蔬菜[:3],label='蔬菜',color='r',marker='*',ms=10)
plt.plot(数据.时间[2:],数据.蔬菜[2:],label='蔬菜',color='b',marker='*',ms=10)
plt.show()
```

02.制作平均线

2020年11月30日 12:51

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
# 柱图
plt.bar(数据.班级,数据.销售量,color='r',label='销售量',alpha=0.6)
```

显示图例

```
plt.legend()
```

平均线-折线图

```
平均线 = np.mean(数据.销售量)
```

```
plt.axhline(y=平均线,color="blue",linestyle=':')
plt.show()
```

```
plt.axhline(y=0.0, c="r", ls="--", lw=2)
```

y: 水平参考线的出发点

c: 参考线的线条颜色

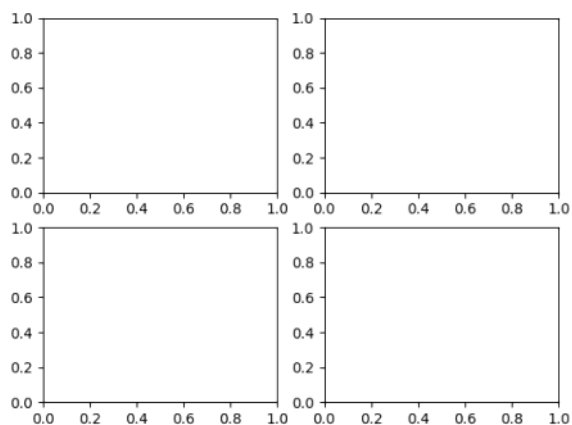
ls: 参考线的线条风格

lw: 参考线的线条宽度

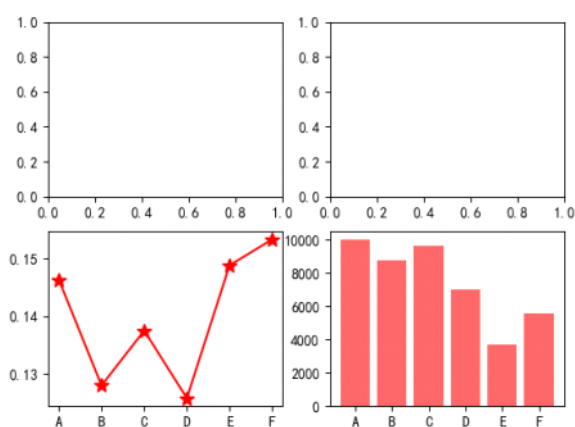
四、画布与子图

2020年11月30日 16:02

```
import matplotlib.pyplot as plt
# 生成一个新图片
布 = plt.figure()
# 在这张图上做一个2*2 (最多4个图)
第1个 = 布.add_subplot(2,2,1)
第2个 = 布.add_subplot(2,2,2)
第3个 = 布.add_subplot(2,2,3)
第3个 = 布.add_subplot(2,2,4)
plt.show()
```



```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
# 生成一个新图片
布 = plt.figure()
# 在这张图上做一个2*2 (最多4个图)
第1个 = 布.add_subplot(2,2,1)
第2个 = 布.add_subplot(2,2,2)
第3个 = 布.add_subplot(2,2,3)
plt.plot(数据.班级,数据.毛利率,label='毛利率',color='r',marker='*',ms=10)
第4个 = 布.add_subplot(2,2,4)
plt.bar(数据.班级,数据.销售量,color='r',label='销售量',alpha=0.6)
plt.show()
```



01.plt.figure()

2020年11月30日 16:11

(1)figure语法说明

figure(num=None, figsize=None, dpi=None, facecolor=None, edgecolor=None, frameon=True)

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；

dpi参数指定绘图对象的分辨率，即每英寸多少个像素，缺省值为80 1英寸等于2.5cm,A4纸是 21*30cm的纸张

facecolor:背景颜色

edgecolor:边框颜色

frameon:是否显示边框

布 = plt.figure(num="孙兴华",figsize=(12,13),dpi=200,facecolor='w',edgecolor='r')

用于创建一个新图。

num	新图的编号，默认递增
figsize	宽度，高度，以英寸为单位
dpi	分辨率，整数
facecolor	背景颜色
edgecolor	边框颜色
frameon	若为False，则没有边框
clear	若为True，如果图的编号已存在则先清除

02.subplot创建单个子图

2020年11月30日 16:19

subplot(nrows,ncols,sharex,sharey,subplot_kw,fig_kw)**

参数	说明
nrows	行数
ncols	列数
sharex	使用相同的x轴刻度
sharey	使用相同的y轴刻度
subplot_kw	创建关键字字典
**fig_kw	使用其它关键字

```
import pandas as pd  
import matplotlib.pyplot as plt  
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文  
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')  
布 = plt.figure()  
plt.subplot(221)  
plt.bar(数据.班级,数据.销售量)  
plt.subplot(223)  
plt.pie(x=数据.销售量,labels=tuple(数据.班级))  
plt.subplot(224)  
plt.plot(数据.班级,数据.毛利率)  
plt.show()
```

03.subplots创建多个子图

2020年11月30日 16:54

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布,图=plt.subplots(2,2)
图1=图[0,0]
图2=图[0,1]
图3=图[1,0]
图4=图[1,1]
# 以下区域开始画图
图1.bar(数据.班级,数据.销售量)
图4.pie(x=数据.销售量,labels=tuple(数据.班级))
plt.show()
```

```
fig, axes = plt.subplots(nrows=2, ncols=2)
axes[0,0].set(title='Upper Left')
axes[0,1].set(title='Upper Right')
axes[1,0].set(title='Lower Left')
axes[1,1].set(title='Lower Right')
```

04.add_subplots与add_axes新增子图或区域

2020年11月30日 17:00

(1)add_subplot新增子图

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布 = plt.figure()
图1=布.add_subplot(221)
图1.bar(数据.班级,数据.销售量)
图4=布.add_subplot(224)
图4.pie(x=数据.销售量,labels=tuple(数据.班级))
plt.show()
```

(2) add_axes新增子区域

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布 = plt.figure()
left,bottom,width,height = 0.1,0.1,0.8,0.8
# 参数是一个列表，距左和下的距离，以及自身的宽和高度
图1 = 布.add_axes([left,bottom,width,height])
图1.bar(数据.班级,数据.销售量)
图1.set_title("销售量")
left,bottom,width,height = 0.65,0.6,0.25,0.25
图2 = 布.add_axes([left,bottom,width,height])
```

图2.plot(数据.班级,数据.毛利率)

图2.set_title("毛利率")

plt.show()

【补充】调整边框与子图间距subplots_adjust

2020年12月4日 19:54

```
subplots_adjust(self, left=None, bottom=None, right=None, top=None,  
                wspace=None, hspace=None)
```

left = 0.125 # 图片中子图的左侧

right = 0.9 # 图片中子图的右侧

bottom = 0.1 # 图片中子图的底部

top = 0.9 # 图片中子图的顶部

wspace = 0.2 # 为子图之间的空间保留的宽度，平均轴宽的一部分

hspace = 0.2 # 为子图之间的空间保留的高度，平均轴高度的一部分

加了这个语句，子图会稍变小，因为空间也占用坐标轴的一部分

```
fig.subplots_adjust(wspace=0.5, hspace=0.5)
```

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

```
布,图=plt.subplots(2,2)
```

```
# wspace子图之间的宽度， hspace子图之间的高度， left代表子图与布的左边距离
```

```
布.subplots_adjust(wspace=0.5, hspace=0.3, left=0.125, right=0.9, top=0.9, bottom=0.1)
```

```
plt.show()
```

```
布.tight_layout() # 自动调整布局，使标题之间不重叠
```

05.柱状图+折线图

2020年11月30日 15:43

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
百分比标签 = mtick.FormatStrFormatter('%%.2f%%')
布=plt.figure()
图1=布.add_subplot(111)
图1.bar(数据.班级,数据.销售量,label="销售量")
图1.legend(loc='upper left')
图1.set_ylim([0,12500])
# twinx()表示共享x轴, 同理twiny()表示共享y轴
图2 = 图1.twinx()
图2.plot(数据.班级,数据.毛利率,color='r',label='毛利率')
图2.yaxis.set_major_formatter(百分比标签)
图2.legend(loc='upper right')
plt.show()
```

这样写百分比显示是错误的

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布=plt.figure()
图1=布.add_subplot(111)
图1.bar(数据.班级,数据.销售量,label="销售量")
图1.legend(loc='upper left')
图1.set_ylim([0,12500])
# twinx()表示共享x轴, 同理twiny()表示共享y轴
图2 = 图1.twinx()
图2.plot(序号,数据.毛利率,'or-',color='r',label='毛利率')
图2.yaxis.set_major_formatter(ticker.PercentFormatter(xmax=1, decimals=2))
图2.legend(loc='upper right')
plt.show()
```

这里出现的两个参数:

xmax: 指定 100% 对应原始数据的值, 默认值是 100, 由于我们的数据是 0~1 之间的小数, 所以这里要设置为 1, 即 数据 中的 1 表示 100%;
decimals: 指定显示小数点后多少位, 默认是由函数自动确定, 这里我们设置成 1, 使之仅显示小数点后 1 位。

```
import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as ticker
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布=plt.figure()
图1=布.add_subplot(111)
图1.bar(数据.班级,数据.销售量,label="销售量")
图1.legend(loc='upper left')
```

```

图1.set_ylim([0,12500])
# twinx()表示共享x轴, 同理twiny()表示共享y轴
图2 = 图1.twinx()
图2.plot(数据.班级,数据.毛利率,'or-',color='r',label='毛利率')
图2.yaxis.set_major_formatter(ticker.PercentFormatter(xmax=1, decimals=2))
图2.legend(loc='upper right')
图2.set_ylim([0, 1])
for x,y in zip(数据.班级,数据.毛利率):
    plt.text(x,y,str(round(y*100,2)) + "%",color='b',fontsize=20)
plt.show()

```

第二种情况: 已经是百分比

```

import pandas as pd
import matplotlib.pyplot as plt
import matplotlib.ticker as mtick
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\09.折线与柱状组合图.xlsx')
布=plt.figure()
图1=布.add_subplot(111)
图1.bar(数据.班级,数据.销售量,label="销售量")
图1.legend(loc='upper left')
图1.set_ylim([0,12500])
# twinx()表示共享x轴, 同理twiny()表示共享y轴
图2 = 图1.twinx()
图2.plot(数据.班级,数据.毛利率2,'or-',color='r',label='毛利率')
图2.yaxis.set_major_formatter(mtick.FormatStrFormatter('%.2f%%'))
图2.legend(loc='upper right')
图2.set_ylim([0, 100])
for x,y in zip(数据.班级,数据.毛利率2):
    plt.text(x,y,str(y) + "%",color='b',fontsize=20)
plt.show()

```

五、坐标轴上面的日期格式

2020年11月30日 18:34

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\12.日期格式.xlsx')
plt.plot(数据.日期,数据.销售)
plt.show()
```

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\12.日期格式.xlsx')
# print([d for d in 数据.日期])
日期 = [d.strftime('%Y-%m-%d') for d in 数据.日期]
plt.plot(日期,数据.销售)
# 文字旋转
plt.xticks(日期,rotation=45)
plt.show()
```

`plt.tight_layout()`
自动适应

#以下是格式定义

代码 说明

%Y 4位数的年

%y 2位数的年

%m 2位数的月[01,12]

%d 2位数的日[01, 31]

%H 时 (24小时制) [00,23]

%I 时 (12小时制) [01,12]

%M 2位数的分[00,59]

%S 秒[00,61]有闰秒的存在

%w 用整数表示的星期几[0 (星期天), 6]

%F %Y-%m-%d简写形式例如, 2017-06-27

%D %m/%d/%y简写形式

六、plt.grid()网格线

2020年11月30日 20:10

plt.grid(b, which, axis, color, linestyle, linewidth, **kwargs)

b : 布尔值。就是是否显示网格线的意思。

which : 取值为 'major' , 'minor' , 'both' 。默认为 'major' 。

axis : 取值为 'both' , 'x' , 'y' 。就是x,y轴的网格线。

color : 设置网格线的颜色。

linestyle : 设置网格线的风格, 是连续实线, 虚线或者其它不同的线条。

linewidth : 设置网格线的宽度

```
import pandas as pd
```

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

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
数据 = pd.read_excel(r'c:\plt\12.日期格式.xlsx')
```

```
plt.grid(axis='x',color='r',linestyle=':',linewidth=3)
```

```
plt.show()
```

七、plt.scatter散点图

2020年11月30日 20:53

`plt.scatter(x, y, s=None, c=None, marker=None, cmap=None, norm=None, vmin=None, vmax=None, alpha=None, linewidths=None, verts=None, edgecolors=None, *, data=None, **kwargs)`

参数的解释：

x, y: 表示的是大小为(n,)的数组，也就是我们即将绘制散点图的数据点

s:是一个实数或者是一个数组大小为(n,)，这个是一个可选的参数。

c:表示的是颜色，也是一个可选项。默认是蓝色'b',表示的是标记的颜色，或者可以是一个表示颜色的字符，或者是一个长度为n的表示颜色的序列等等，感觉还没用到过现在不解释了。但是c不可以是一个单独的RGB数字，也不可以是一个RGBA的序列。可以是他们的2维数组（只有一行）。

marker:表示的是标记的样式，默认的是'o'。

cmap:Colormap实体或者是一个colormap的名字，cmap仅仅当c是一个浮点数数组的时候才使用。如果没有申明就是image.cmap

norm:Normalize实体来将数据亮度转化到0-1之间，也是只有c是一个浮点数的数组的时候才使用。如果没有申明，就是默认为colors.Normalize。

vmin,vmax:实数，当norm存在的时候忽略。用来进行亮度数据的归一化。

alpha: 实数，0-1之间。

linewidths:也就是标记点的长度。

01.散点图

2020年11月30日 21:18

```
import pandas as pd
import matplotlib.pyplot as plt
# 显示所有列，同理：max_rows
pd.options.display.max_columns=None
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\14.散点图.xlsx')
plt.scatter(数据.身高,数据.体重,s=数据.身高,c='b',marker='o',alpha=0.6,linewidths=20)
plt.show()
```

x,y	X和Y是长度相同的数组
s	size,点的大小，标量或与数据长度相同的数组
c	color,点的颜色，标量或与数据长度相同的数组
marker	MarketStyle, 可选，点的形状，默认'o'
cmap	Colormap, 可选，默认'None'
norm	Normalize, 亮度设置,0-1
vmin,vmax	亮度设置
alpha	透明度，0-1
linewidths	

02.带颜色的散点图

2020年12月4日 20:30

```
import numpy as np
import matplotlib.pyplot as plt
k= 500
x = np.random.rand(k) # 返回[0-1之间均匀分布的随机样本]
y = np.random.rand(k)
大小 = np.random.rand(k)*50 # 生成每个点的大小
# arctan2求反正切值
颜色 = np.arctan2(x,y) # 生成每个点的颜色大小
plt.scatter(x,y,s=大小,c=颜色)
plt.colorbar() # 添加颜色栏
plt.show()
```

```
import pandas as pd
import matplotlib.pyplot as plt
# 显示所有列，同理：max_rows
pd.options.display.max_columns=None
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\14.散点图.xlsx')
plt.scatter(数据.身高,数据.体重,s=数据.身高,c=数据.身高)
plt.colorbar() # 添加颜色栏
plt.show()
```


八.plt.hist直方图

2020年12月3日 15:37

首先需要区分清楚概念：直方图和条形图。

条形图：条形图用长条形表示每一个类别，长条形的长度表示类别的频数，宽度表示表示类别。

直方图：直方图是一种统计报告图，形式上也是一个个的长条形，但是直方图用长条形的面积表示频数，所以长条形的高度表示 **频数/组距**，宽度表示组距，其长度和宽度均有意义。当宽度相同时，一般就用长条形长度表示频数。

直方图一般用来描述等距数据，柱状图一般用来描述名称（类别）数据或顺序数据。直观上，直方图各个长条形是衔接在一起的，表示数据间的数学关系；条形图各长条形之间留有空隙，区分不同的类。

区别	频数分布直方图	条形图
横轴上的数据	连续的，是一个范围	孤立的，代表一个类别
长条形之间	没有空隙	有空隙
频数的表示	一般用长条形面积表示；当宽度相同时，用长度表示	长条形的长度

01.直方图

2020年12月3日 15:37

```
import pandas as pd
import matplotlib.pyplot as plt
# 显示所有列, 同理: max_rows
pd.options.display.max_columns=None
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\直方图.xlsx')
plt.hist(数据.身高,bins=30,facecolor='b',edgecolor='r',alpha=0.8)
plt.show()
```

九、关于轴上面的操作

2020年12月3日 23:14

01.轴的颜色及隐藏轴边框

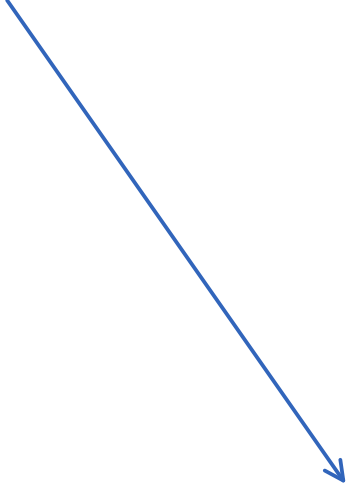
2020年12月3日 23:06

```
import matplotlib.pyplot as plt
布, 图 = plt.subplots(1,1)
图.spines['left'].set_color('g')
图.spines['bottom'].set_color('r')
图.spines['top'].set_color('none')
图.spines['right'].set_color('none')
plt.show()
```

02.翻转x轴和y轴

2020年12月3日 23:17

```
import matplotlib.pyplot as plt
布, 图 = plt.subplots(1,1)
图.spines['left'].set_color('g')
图.spines['bottom'].set_color('r')
图.spines['top'].set_color('none')
图.spines['right'].set_color('none')
plt.gca().invert_yaxis() # 翻转y轴
plt.gca().invert_xaxis() # 翻转x轴
plt.show()
```



04.添加数据标签那课用过

03.隐藏x与y轴

2020年12月3日 23:28

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\直方图.xlsx')
布, 图 = plt.subplots(1,1)
图.plot(数据.序号,数据.身高)
图.spines['left'].set_color('none')
图.spines['bottom'].set_color('none')
图.spines['top'].set_color('none')
图.spines['right'].set_color('none')
图.set_xticks([])
图.set_yticks([])
plt.show()
```

将上下左右全设置成“无”
刻度也设置成“无”
也可以写成：

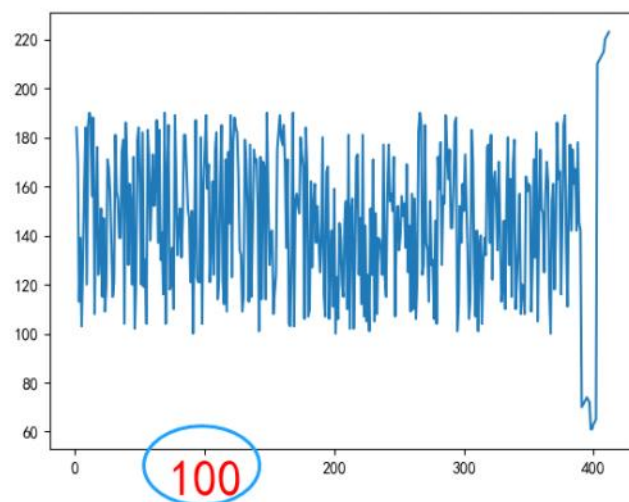
`图.spines['left'].set_visible(False)`

```
图.xaxis.set_ticks_position('top') # x轴显示在顶部
图.xaxis.set_ticks_position('bottom')
图.yaxis.set_ticks_position('left') # y轴显示在左侧
图.yaxis.set_ticks_position('right')
```

04.突出某个坐标值

2020年12月3日 23:48

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\直方图.xlsx')
布, 图 = plt.subplots(1,1)
图.plot(数据.序号,数据.身高)
plt.gca().get_xticklabels()[2].set(color='red',fontsize=30)
plt.show()
```



05.坐标轴刻度的调整

2020年12月4日 0:00

在python中，可以使用locator_params进行对坐标轴刻度的调整，通过nbins设置坐标轴一共平均分为几份，也可以显式的指定要调整的坐标轴，可以通过面向对象的方式或者是pyplot的方式，pyplot的方式更简单。

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
数据 = pd.read_excel(r'c:\plt\直方图.xlsx')
布, 图 = plt.subplots(1,1)
图.plot(数据.序号,数据.身高)
```

第一种方法:

```
# plt.locator_params("x", nbins = 5)
```

第二种方法:

```
轴 = plt.gca() # 获取当前轴
```

```
#轴.locator_params("x", nbins=5)
```

x与y都平均分成5份

```
轴.locator_params(nbins=5)
```

```
plt.show()
```

设置X轴的刻度是7的倍数，或改成小数点形式

```
import pandas as pd
```

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

```
import matplotlib.pyplot as ticker
```

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
plt.rcParams['axes.unicode_minus'] = False
```

```
数据 = pd.read_excel(r'c:\plt\直方图.xlsx')
```

```
plt.plot(数据.序号,数据.身高)
```

```
轴=plt.gca()
```

```
轴.xaxis.set_major_locator(ticker.MultipleLocator(7)) # 将x轴设置成7的倍数
```

```
# 轴.xaxis.set_major_formatter(ticker.FormatStrFormatter('%.1f')) 保留一位小数点
```

```
plt.show()
```

第二种处理日期轴的方法

```
import pandas as pd
```

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

```
import matplotlib as mpl
```

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
plt.rcParams['axes.unicode_minus'] = False
```

```
数据 = pd.read_excel(r'c:\plt\12.日期.xlsx')
```

```
print(数据)
```

```
plt.plot(数据.日期,数据.销售)
```

```
轴=plt.gca()
```

```
轴.xaxis.set_major_formatter(mpl.dates.DateFormatter("%Y*%m*%d"))
```

```
plt.show()
```


06.x轴与y轴区间

2020年12月4日 19:41

```
import pandas as pd
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
plt.rcParams['axes.unicode_minus'] = False
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
plt.plot(数据.时间,数据.蔬菜)
轴=plt.gca()
# 轴.set_xlim([0, 10]) #设置X轴的区间
# 轴.set_ylim([0, 100]) #Y轴区间
# 轴.axis([0, 10, 0, 100]) #X、Y轴区间 [xmin,xmax,ymin,ymax]
轴.set_ylim(bottom=-10) #Y轴下限
轴.set_xlim(right=25) #X轴上限
plt.show()
```

十、叠加区域图 fill_between()

2020年12月4日 20:49

```
plt.fill_between(x, 0, y, facecolor='green', alpha=0.3)
```

参数:

x: 第一个参数表示覆盖的区域, 我直接复制为x, 表示整个x都覆盖

0: 表示覆盖的下限

y: 表示覆盖的上限是y这个曲线

facecolor: 覆盖区域的颜色

alpha: 覆盖区域的透明度[0,1],其值越大, 表示越不透明

```
import pandas as pd
```

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

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
```

```
plt.plot(数据.时间,数据.蔬菜)
```

```
plt.plot(数据.时间,数据.水果)
```

```
plt.fill_between(数据.时间,0,数据.蔬菜,facecolor='r',alpha=0.3)
```

```
plt.fill_between(数据.时间,0,数据.水果,facecolor='g',alpha=0.3)
```

```
plt.show()
```

两条曲线之间的面积填充

```
import pandas as pd
```

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

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
数据 = pd.read_excel(r'c:\plt\08.折线图.xlsx')
```

```
plt.plot(数据.时间,数据.蔬菜)
```

```
plt.plot(数据.时间,数据.水果)
```

```
plt.fill_between(数据.时间,数据.蔬菜,数据.水果,facecolor='r',alpha=0.3)
```

```
plt.show()
```

高亮显示数据

import numpy as np

import matplotlib.pyplot as plt

x = np.array([i for i in range(30)])

y = np.random.rand(30)

设置想要高亮数据的位置

列表 = [[1, 6],[10, 12],[20, 23], [26, 28]]

画图

plt.plot(x, y, 'r')

for i in 列表:

plt.fill_between(x[i[0] : i[1]], 0, 1, facecolor='r', alpha=0.3)

plt.show()

十一、极坐标与雷达图

2020年12月5日 9:59

先了解一下什么是极坐标

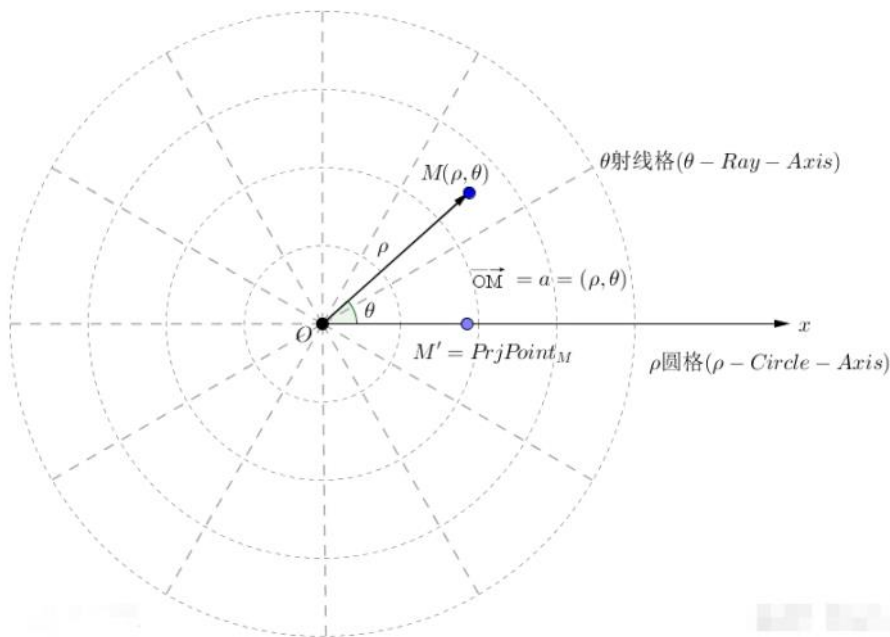
极点：以圆的中心作为极点O，

极轴：以0°的方向引一条射线极轴Ox，

极径：选定一个长度单位r

极角：以Ox正方向开始计算角度 θ （通常取逆时针方向）

极坐标：以极点O作为圆心，以极轴Ox的方向作为起点，以极径r作为半径，画一个以极角 θ 的扇形，最终圆规脚定的位置就是极坐标M



使用matplotlib画雷达图，也就是画极坐标，使用plt.polar函数画一张空白极地图

```
import matplotlib.pyplot as plt
plt.polar()
plt.show()
```

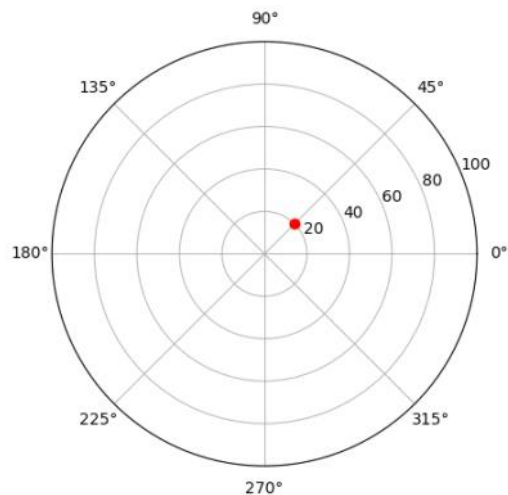
语法：

```
plt.polar(极角,极轴,**kwargs)
```

```
import numpy as np
import matplotlib.pyplot as plt
# 一个派是180度，极角是0.25*np.pi=45度，极轴20
plt.polar(0.25*np.pi,20,'ro')
plt.ylim(0,100) #设置极轴的上下限
plt.show()
```

给定多个极角和极轴的时候

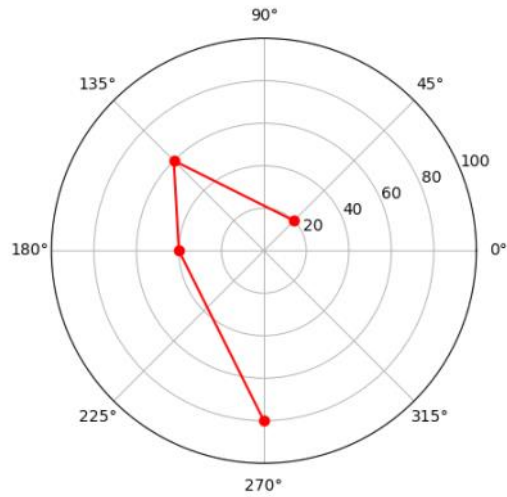
```
import numpy as np
import matplotlib.pyplot as plt
角度 = np.array([0.25,0.75,1,1.5])
极轴 = [20,60,40,80]
plt.polar(角度*np.pi,极轴,'ro')
plt.ylim(0,100)
plt.show()
```



```
plt.ylim(0,100)
plt.show()
```

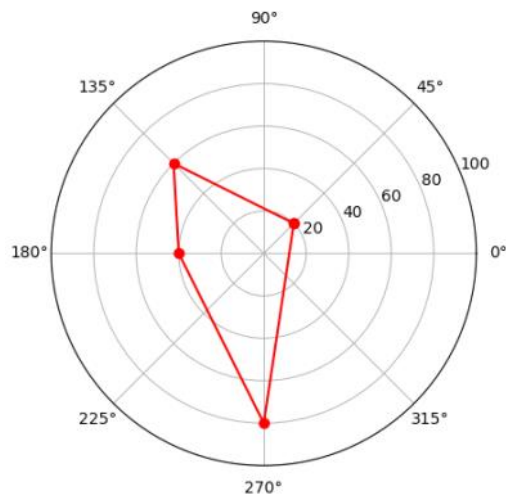
将每个点连线

```
import numpy as np
import matplotlib.pyplot as plt
角度 = np.array([0.25,0.75,1,1.5])
极轴 = [20,60,40,80]
plt.polar(角度*np.pi,极轴,'ro-')
plt.ylim(0,100)
plt.show()
```



构造一个极坐标点，和第一个点重叠

```
import numpy as np
import matplotlib.pyplot as plt
角度 = np.array([0.25,0.75,1,1.5,0.25])
极轴 = [20,60,40,80,20]
plt.polar(角度*np.pi,极轴,'ro-')
plt.ylim(0,100)
plt.show()
```



填充颜色

```
import numpy as np
import matplotlib.pyplot as plt
角度 = np.array([0.25,0.75,1,1.5,0.25])
极轴 = [20,60,40,80,20]
plt.polar(角度*np.pi,极轴,'ro-')
plt.fill(角度*np.pi,极轴,'r',alpha=0.75)
plt.ylim(0,100)
plt.show()
```

01.雷达图

2020年12月5日 15:05

numpy.linspace

numpy.linspace 函数用于创建一个一维数组，数组是一个等差数列构成的，格式如下：

```
np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)
```

参数说明：

参数	描述
start	序列的起始值
stop	序列的终止值，如果endpoint为true，该值包含于数列中
num	要生成的等步长的样本数量，默认为50
endpoint	该值为ture 时，数列中中包含stop值，反之不包含，默认是True。
retstep	如果为 True 时，生成的数组中会显示间距，反之不显示。
dtype	ndarray的数据类型 https://blog.csdn.net/dugushangliang

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
# 使用ggplot的风格绘图
plt.style.use('ggplot')
数据=pd.read_excel(r'c:\plt\22.雷达图.xlsx',index_col='姓名')
条件1 = "姓名=='A01'"
条件2 = "姓名=='A02'"
# print(数据.query(条件1))
A01 = 数据.query(条件1)['分数']
A02 = 数据.query(条件2)['分数']
科目 = 数据.query(条件1)['科目']
# 设置雷达图角度，用于平分切开一个平面 linspace(起始，结束，len(A01)个等步长的样本数量， endpoint不包括结束值)
角度 = np.linspace(0, 2*np.pi,len(A01), endpoint=False)
# 使雷达图封闭起来
A01 = np.concatenate((A01,[A01[0]]))
# print(A01)
A02 = np.concatenate((A02,[A02[0]]))
角度 = np.concatenate((角度,[角度[0]]))
科目 = np.concatenate((科目,[科目[0]]))
# 画图
布=plt.figure()
图 = 布.add_subplot(111,polar=True)
图.plot(角度,A01,'o-',linewidth=2,alpha=0.25,label='A01同学') #linewidth线粗细
图.fill(角度,A01,'r',alpha=0.25)
图.plot(角度,A02,'o-',linewidth=2,alpha=0.25,label='A02同学') #linewidth线粗细
图.fill(角度,A02,'b',alpha=0.25)
# 标签
图.set_thetagrids(角度 * 180 / np.pi,科目)
```

```
图.set_ylim(0,100)  
plt.legend(loc='best')  
plt.show()
```

十二、标题及轴标签

2020年12月6日 7:59

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
布 = plt.figure()
plt.subplot(221)
plt.plot(["A","B","C"],[1,2,3],'ro-')
plt.title("孙兴华图1",c='b')
# 字典=dict(facecolor='yellow', pad=5, alpha=0.2)
plt.xlabel("x轴",bbox={'facecolor':'yellow','pad':5,'alpha':0.2})
plt.subplot(222)
plt.subplot(223)
plt.subplot(224)
# 整个画板的标题
plt.suptitle("孙兴华",fontsize=16,fontweight='bold',color='r')
# wspace子图之间的宽度, hspace子图之间的高度, left代表子图与布的左边距离
plt.subplots_adjust(left=0.2,top=0.8,wspace=0.8,hspace=0.8,bottom=0.1)
plt.show()
```


十三、样式

2020年12月6日 9:17

01.折线的样式

2020年12月6日 9:13

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
x= ["A","B","C","D","E","F"]
y= [1,10,7,5,11,6]
# drawstyle有steps、steps-pre、steps-mid、steps-post和默认共5种
plt.plot(x,y,"r-",drawstyle='steps')
plt.show()
```

02.画布内置样式

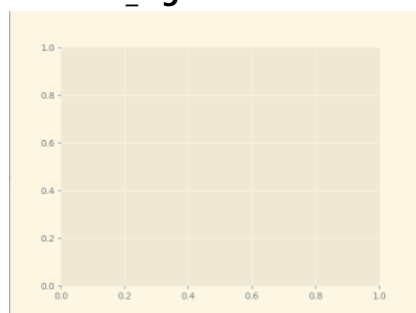
2020年12月6日 9:17

查看所有内置样式

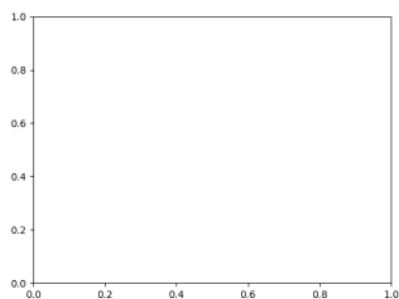
```
import matplotlib.pyplot as plt
print(plt.style.available)
```

```
import matplotlib.pyplot as plt
plt.style.use('tableau-colorblind10')
布, 图 = plt.subplots()
plt.show()
```

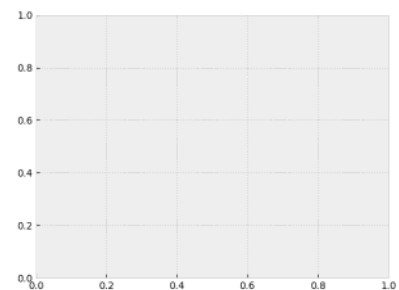
'Solarize_Light2'



'_classic_test_patch'



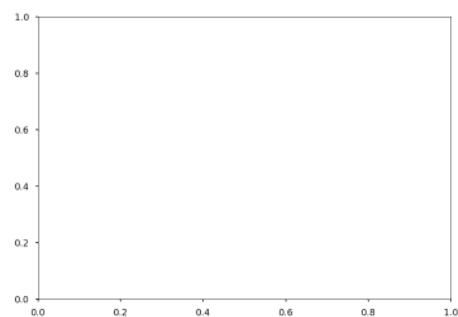
'bmh'



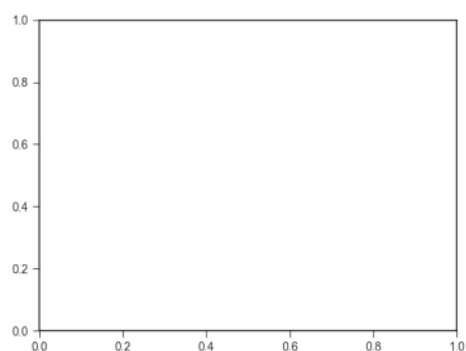
'classic'



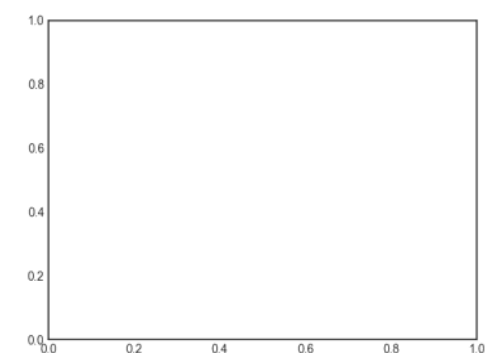
'seaborn-talk'



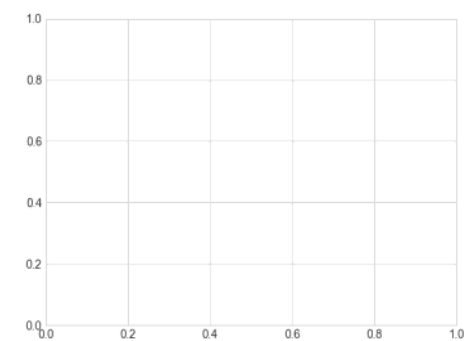
'seaborn-ticks'



'seaborn-white'

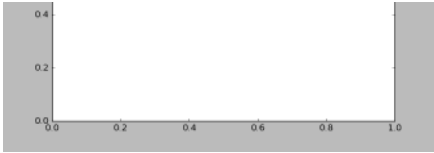


'seaborn-whitegrid'

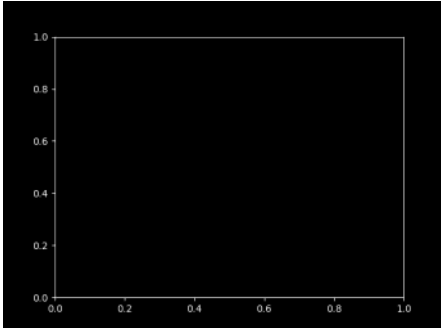


'tableau-colorblind10'

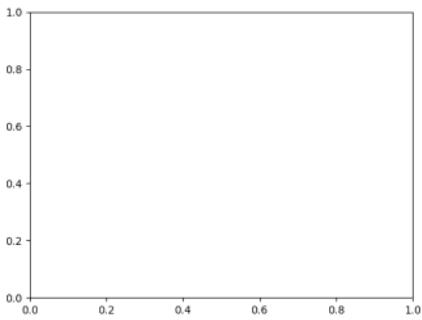




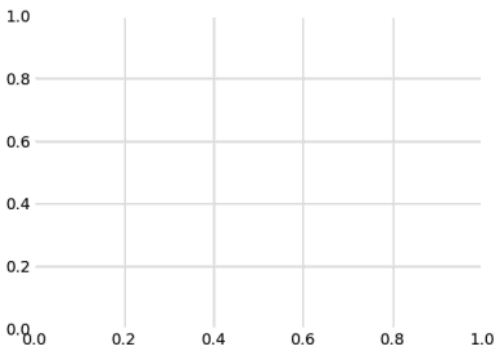
'dark_background'



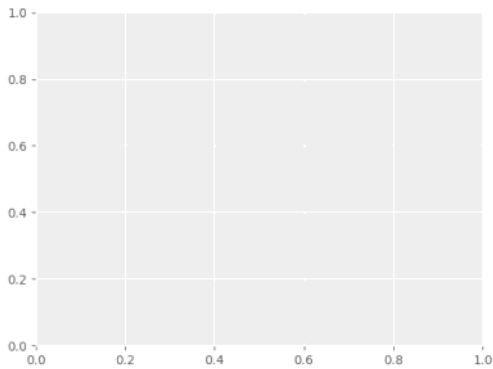
'fast'



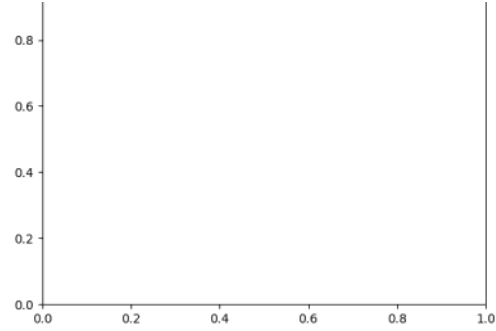
'fivethirtyeight'

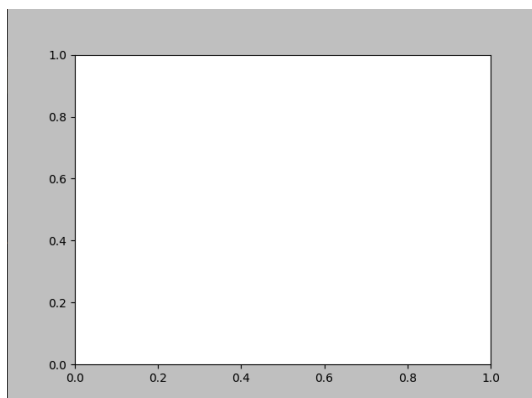


'ggplot'

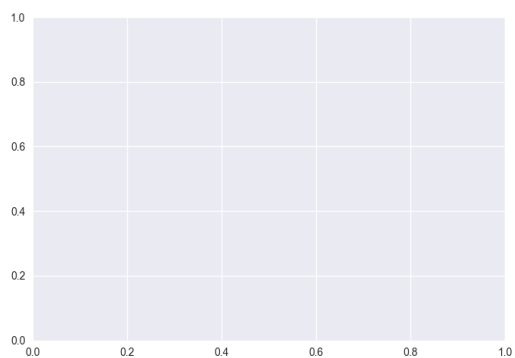


'grayscale'

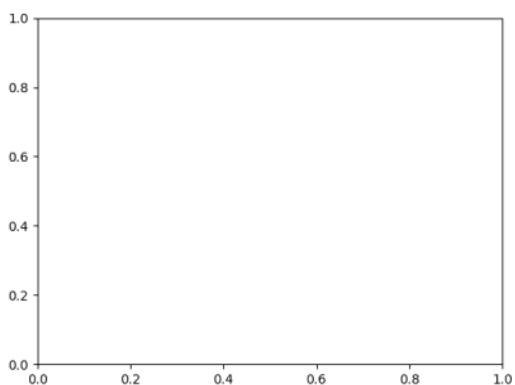




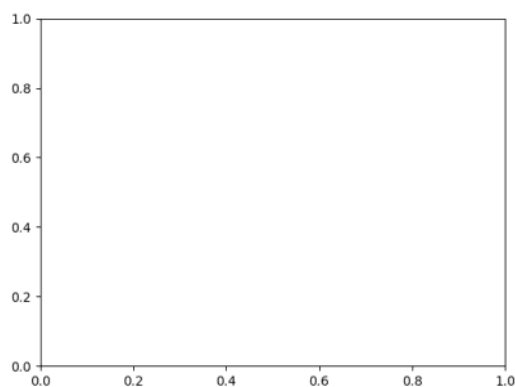
'seaborn'



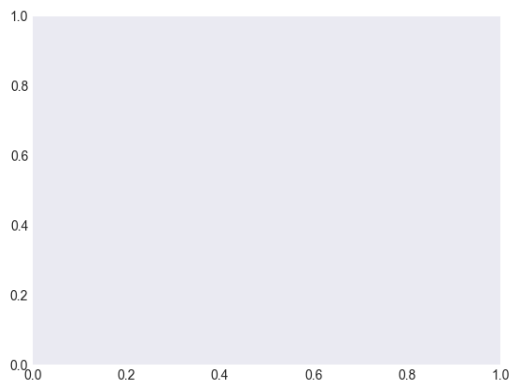
'seaborn-bright'



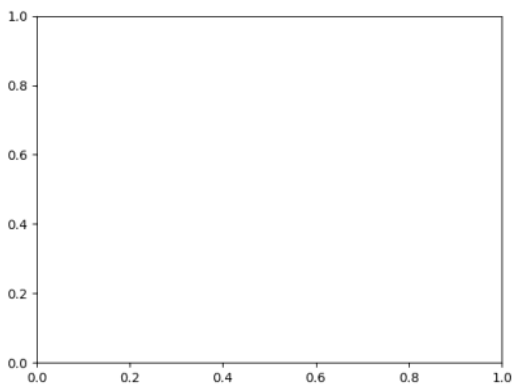
'seaborn-colorblind'



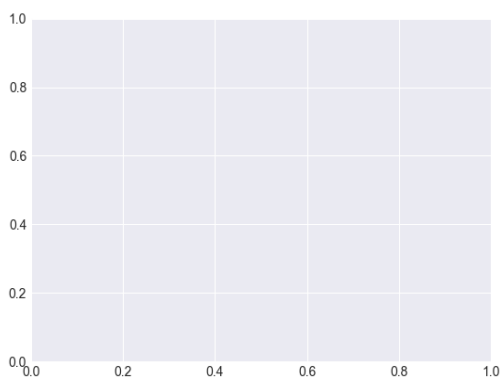
'seaborn-dark'



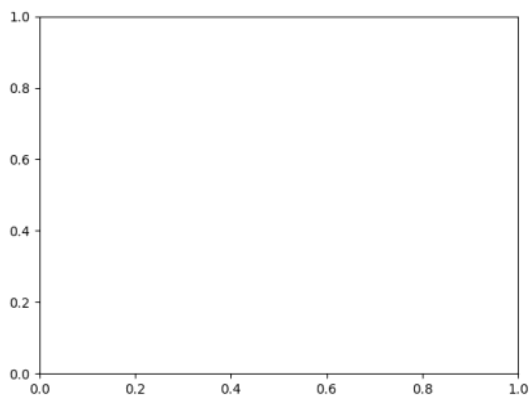
'seaborn-dark-palette'



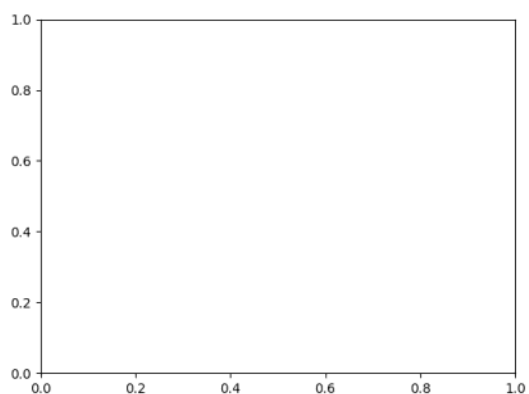
'seaborn-darkgrid'



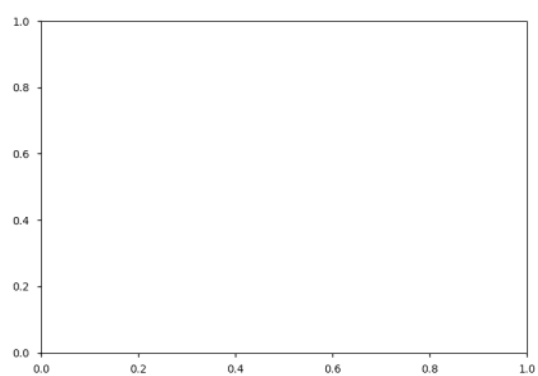
'seaborn-deep'



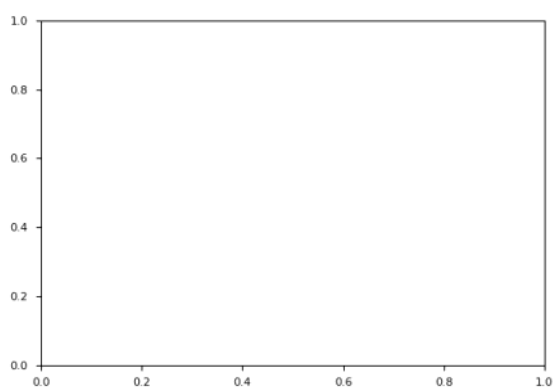
'seaborn-muted'



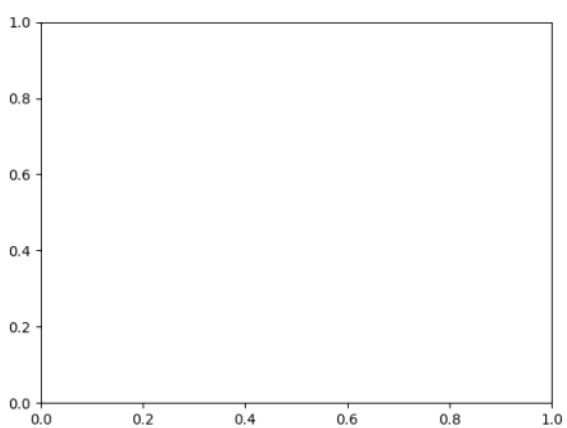
'seaborn-notebook'



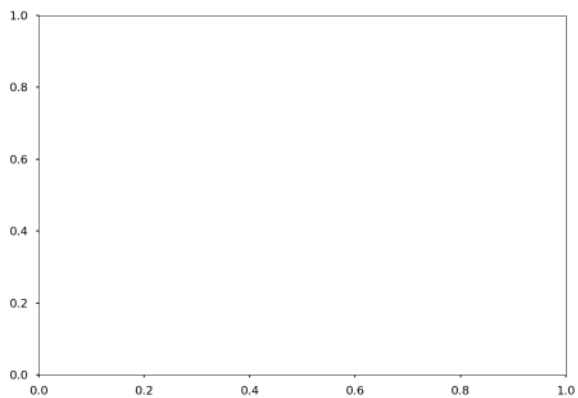
'seaborn-paper'



'seaborn-pastel'



'seaborn-poster'



十四、线条及填充

2020年12月6日 9:50

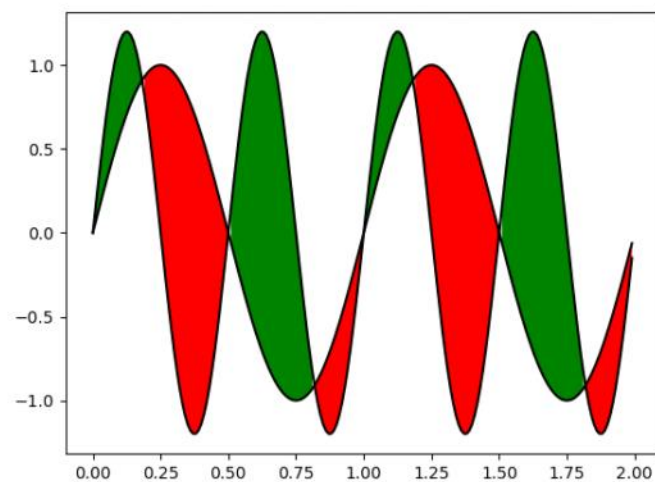
```
import matplotlib.pyplot as plt

# 画横线与纵线
plt.axhline(y=0.2,linewidth=8,c='r') # 横线
plt.axvline(x=0.4) # 纵线
# 线段
plt.axhline(y=0.5,xmin=0.2,xmax=0.6,c='g')
# 填充
plt.axhspan(1,1.3,facecolor='y',alpha=0.5)
plt.axvspan(1,1.3,facecolor='b',alpha=0.5)
# 设置坐标轴
plt.axis([-1,2,-1,2])
plt.show()
```

十五、交差及填充

2020年12月6日 10:02

```
import matplotlib.pyplot as plt
import numpy as np
x = np.arange(0.0, 2, 0.01)
y1 = np.sin(2*np.pi*x)
y2 = 1.2*np.sin(4*np.pi*x)
布, 图 = plt.subplots()
图.plot(x, y1, x, y2, color='black')
图.fill_between(x, y1, y2, where=y2>y1, facecolor='green')
图.fill_between(x, y1, y2, where=y2<=y1, facecolor='red')
plt.show()
```



十六、文本注释annotate

2020年12月6日 10:08

参数说明：

图.annotate(s, xy, *args, **kwargs)

s：注释文本的内容

xy：被注释的坐标点，二维元组形如(x,y)

xytext：注释文本的坐标点，也是二维元组，默认与xy相同

xycoords：被注释点的坐标系属性，允许输入的值如下

属性值	含义
'figure points'	以绘图区左下角为参考，单位是点数
'figure pixels'	以绘图区左下角为参考，单位是像素数
'figure fraction'	以绘图区左下角为参考，单位是百分比
'axes points'	以子绘图区左下角为参考，单位是点数（一个figure可以有多个axex，默认为1个）
'axes pixels'	以子绘图区左下角为参考，单位是像素数
'axes fraction'	以子绘图区左下角为参考，单位是百分比
'data'	以被注释的坐标点xy为参考（默认值）
'polar'	不使用本地数据坐标系，使用极坐标系

textcoords：注释文本的坐标系属性，默认与xycoords属性值相同，也可设为不同的值。除了允许输入xycoords的属性值，还允许输入以下两种：

属性值	含义
'offset points'	相对于被注释点xy的偏移量（单位是点）
'offset pixels'	相对于被注释点xy的偏移量（单位是像素）

arrowprops：箭头的样式，dict（字典）型数据，如果该属性非空，则会在注释文本和被注释点之间画一个箭头。如果不设置'arrowstyle' 关键字，则允许包含以下关键字：

关键字	说明
width	箭头的宽度（单位是点）
headwidth	箭头头部的宽度（点）
headlength	箭头头部的长度（点）
shrink	箭头两端收缩的百分比（占总长）

如果设置了 ‘arrowstyle’ 关键字，以上关键字就不能使用。允许的值有：

箭头的样式	属性
'-'	None
'->'	head_length=0.4,head_width=0.2
'-['	widthB=1.0,lengthB=0.2,angleB=None
' - '	widthA=1.0,widthB=1.0
'-> >'	head_length=0.4,head_width=0.2
'< -'	head_length=0.4,head_width=0.2
'< ->'	head_length=0.4,head_width=0.2
'< -'	head_length=0.4,head_width=0.2
'< - >'	head_length=0.4,head_width=0.2
'fancy'	head_length=0.4,head_width=0.4,tail_width=0.4
'simple'	head_length=0.5,head_width=0.5,tail_width=0.2
'wedge'	tail_width=0.3,shrink_factor=0.5

FancyArrowPatch的关键字包括：

Key	Description
arrowstyle	箭头的样式
connectionstyle	连接线的样式
relpos	箭头起始点相对注释文本的位置，默认为 (0.5, 0.5)，即文本的中心，(0, 0) 表示左下角， (1, 1) 表示右上角
patchA	箭头起点处的图形（matplotlib.patches对象），默认是注释文字框
patchB	箭头终点处的图形（matplotlib.patches对象），默认为空
shrinkA	箭头起点的缩进点数，默认为2
shrinkB	箭头终点的缩进点数，默认为2
mutation_scale	default is text size (in points)
mutation_aspect	default is 1.

annotation_clip : 布尔值，可选参数，默认为空。设为True时，只有被注释点在子图区内时才绘制注释；设为False时，无论被注释点在哪里都绘制注释。仅当xycoords为 ‘data’ 时，默认值空相当于True。

01.最简单的

2020年12月6日 11:02

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
plt.plot(["A","B","C"],[5,2,3], 'ro-')
plt.title("孙兴华图1",c='b')
plt.annotate("在这里",xy=("B",2),xytext=("B",2.5),textcoords='axes points',arrowprops=dict(width=1,facecolor='black', shrink=0.05))
plt.show()
```

箭头线的长度

| 参数 | 坐标系 |

| 'figure points' | 距离图形左下角的点数量 |

| 'figure pixels' | 距离图形左下角的像素数量 |

| 'figure fraction' | 0,0 是图形左下角, 1,1 是右上角 |

| 'axes points' | 距离轴域左下角的点数量 |

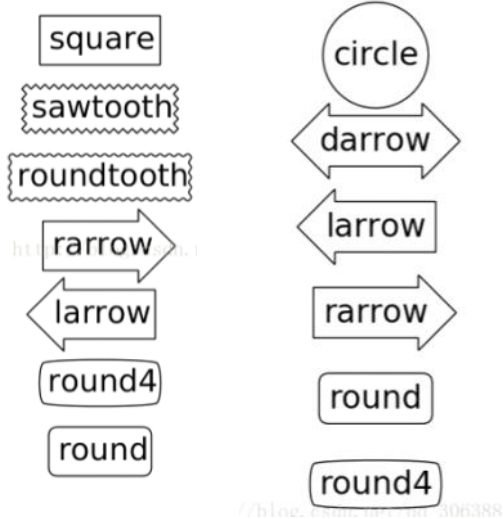
| 'axes pixels' | 距离轴域左下角的像素数量 |

| 'axes fraction' | 0,0 是轴域左下角, 1,1 是右上角 |

| 'data' | 使用轴域数据坐标系 |

02.带边框的文本注释

2020年12月6日 11:05



其中boxstyle用于指定边框类型，fc用于设定边框背景灰度，范围在0-1之间，1.0位白，0.0为黑，不设置该属性的时候背景色为蓝色

边框 = dict(boxstyle="sawtooth",fc="0.8")

bbox给标题增加外框，常用参数如下：

- boxstyle方框外形
- facecolor(简写fc)背景颜色
- edgecolor(简写ec)边框线条颜色
- edgewidth边框线条大小

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

```
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
```

```
plt.plot(["A","B","C"],[1,2,3],'ro-')
```

```
边框=dict(boxstyle="sawtooth",fc="0.8",ec="r") # fc代表边框的灰度
```

```
plt.text("B",2,"在这里",size=20,bbox=边框)
```

```
plt.show()
```

03.变形箭头方式

2020年12月6日 11:26

```
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei'] # 用黑体显示中文
plt.plot(["A","B","C"],[5,2,3],'ro-')
# angle,angle3,arc,arc3,bar
plt.annotate("孙",xy=("B",2),xytext=("C",5),arrowprops=dict(arrowstyle='->',connectionstyle="angle"))
plt.show()
```

名称
-
->
-
-
- >
<-
<->
< -
<
fancy
simple
wedge

名称
angle
angle3
arc
arc3
bar

十七、瀑布图

2020年12月6日 10:20

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams["font.sans-serif"] = ["SimHei"]
plt.rcParams['axes.unicode_minus'] = False
数据=pd.read_excel(r'c:\plt\28.瀑布图.xlsx')
```

```
列表=np.arange(len(数据.金额),dtype=np.float64)
下面的距离 = 0
for i in 数据.金额.index:
    x = 列表[i]
    y=数据.金额[i]
    if 数据.金额[i]>=0:
        盈利 = plt.bar(x,y,0.8,align='center',bottom=下面的距离,label="盈利",color='r')
    else:
        亏损 = plt.bar(x, y, 0.8, align='center', bottom=下面的距离, label="亏损", color='g')
    下面的距离 += y
    x+=0.8
plt.legend(handles=[盈利,亏损])
plt.gca().yaxis.grid(True, linestyle='--', color='grey', alpha=.25)
日期 =[d.strftime('%Y-%m-%d') for d in 数据.日期]
plt.xticks(np.arange(len(数据.金额)),日期,rotation=45)
plt.show()
```


十八、树状图

2020年12月6日 15:05

安装模块 squarify

```
import matplotlib.pyplot as plt
import squarify
import pandas as pd
plt.rcParams['font.sans-serif'] = 'Microsoft YaHei'
plt.rcParams['axes.unicode_minus'] = False
数据 = pd.read_excel(r'c:\plt\29.树状图.xlsx')
自定义颜色=['r','y','b','g','yellow','cyan','coral']
图=squarify.plot(sizes=数据.销售数量, #指定数据
                 label=数据.名称, #指定标签
                 color=自定义颜色, #自定义颜色
                 alpha=0.6,
                 value=数据.销售数量, #添加数据标签
                 edgecolor='white', #设置边界框颜色为白色
                 linewidth=3, #设置边框宽度
                 text_kwargs={'fontsize':16}) # 设置字体大小
图.set_title("销售情况", fontdict={'fontsize':20})
plt.axis('off') # 去掉坐标轴
plt.tick_params(top = 'off', right = 'off') # 去掉刻度
plt.show()
```

十九、玫瑰图

2020年12月6日 15:53

```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
plt.rcParams['font.sans-serif']=['SimHei']
数据=pd.read_excel(r'c:\plt\30.玫瑰图.xlsx')
角度=np.linspace(0,2*np.pi,len(数据.业绩),endpoint=False)
图=plt.axes(polar=True) # 实例化极坐标系
# 图.set_theta_direction(-1) # 顺时针为极坐标正方向
图.set_theta_zero_location('N') # 让0度指向N
列表=np.random.random((len(数据.业绩)))
颜色=['b','gold','darkviolet','turquoise','r','g','grey','c','m','y','k','darkorange','lightgreen','plum','tan']
业绩 = np.concatenate((数据.业绩,[数据.业绩[0]]))
角度 = np.concatenate((角度,[角度[0]]))
姓名 = np.concatenate((数据.姓名,[数据.姓名[0]]))
plt.bar(角度,业绩,width=0.33,color=颜色)
plt.bar(x=角度, height=130, width=0.33, color='white') # 挖孔
# 数据标签
for 角度, 业绩, 姓名 in zip(角度, 业绩, 姓名):
    plt.text(角度+0.03, 业绩+100, str(姓名) )
plt.gca().set_axis_off()
plt.show()
```

附1：常用颜色

2020年12月6日 9:07

字符	颜色
'b'	blue
'g'	green
'r'	red
'c'	cyan
'm'	magenta
'y'	yellow
'k'	black
'w'	white

附2：颜色表

2020年12月6日 9:07

	black		bisque		lightgreen		slategrey
	k		darkorange		forestgreen		lightsteelblue
	dimgray		burlywood		limegreen		cornflowerblue
	dimgrey		antiquewhite		darkgreen		royalblue
	grey		tan		green		ghostwhite
	gray		navajowhite		g		lavender
	darkgrey		blanchedalmond		lime		midnightblue
	darkgray		papayawhip		seagreen		navy
	silver		moccasin		mediumseagreen		darkblue
	lightgray		orange		springgreen		mediumblue
	lightgrey		wheat		mintcream		blue
	gainsboro		oldlace		mediumspringgreen		b
	whitesmoke		floralwhite		mediumaquamarine		slateblue
	white		darkgoldenrod		aquamarine		darkslateblue
	w		goldenrod		turquoise		mediumslateblue
	snow		cornsilk		lightseagreen		mediumpurple
	rosybrown		gold		mediumturquoise		blueviolet
	lightcoral		lemonchiffon		azure		indigo
	indianred		khaki		lightcyan		darkorchid
	brown		palegoldenrod		paleturquoise		darkviolet
	firebrick		darkkhaki		darkslategray		mediumorchid
	maroon		ivory		darkslategrey		thistle
	darkred		beige		teal		plum
	red		lightyellow		darkcyan		violet
	r		lightgoldenrodyellow		c		purple
	mistyrose		olive		cyan		darkmagenta
	salmon		y		aqua		m
	tomato		yellow		darkturquoise		fuchsia
	darksalmon		olivedrab		cadetblue		magenta
	coral		yellowgreen		powderblue		orchid
	orangered		darkolivegreen		lightblue		mediumvioletred
	lightsalmon		greenyellow		deepskyblue		deeppink
	sienna		chartreuse		skyblue		hotpink
	seashell		lawngreen		lightskyblue		lavenderblush
	chocolate		sage		steelblue		palevioletred
	saddlebrown		lightsage		aliceblue		crimson
	sandybrown		darksage		dodgerblue		pink
	peachpuff		honeydew		lightslategrey		lightpink
	peru		darkseagreen		lightslategray		
	linen		palegreen		slategray		

附3：常用标记点marker

2020年12月6日 9:08

字符	实例	字符	实例
'_'	实线	'--'	虚线
'-.'	虚点线	'.'	点线
'.'	点	'.'	像素点
'o'	实心圆点	'v'	下三角点
'^'	上三角点	'<'	左三角点
'>'	右三角点	'1'	下三叉点
'2'	上三叉点	'3'	左三叉点
'4'	右三叉点	's'	正方点
'p'	五角点	'*'	星形点
'h'	六边形点1	'H'	六边形点2
'+'	加号点	'x'	乘号点
'D'	实心菱形点	'd'	瘦菱形点
'_'	横线点		

附4：两点之间的连接方式linestyle

2020年12月6日 9:09

'-' 实线

'--' 破折线

'-.' 点划线

':' 虚线

'''' 无线条

附5.画布内置样式

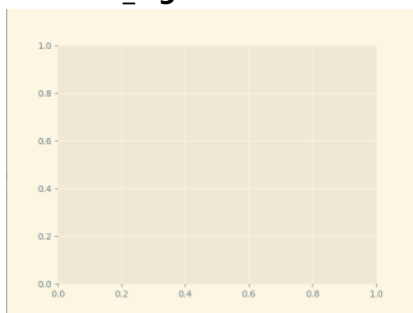
2020年12月6日 9:17

查看所有内置样式

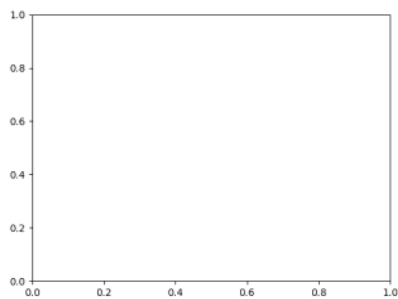
```
import matplotlib.pyplot as plt
print(plt.style.available)
```

```
import matplotlib.pyplot as plt
plt.style.use('tableau-colorblind10')
布, 图 = plt.subplots()
plt.show()
```

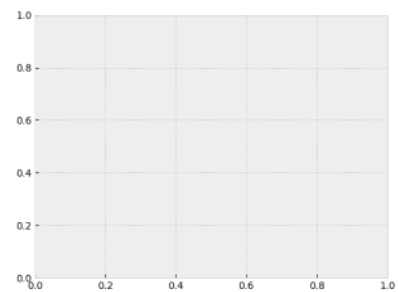
'Solarize_Light2'



'_classic_test_patch'



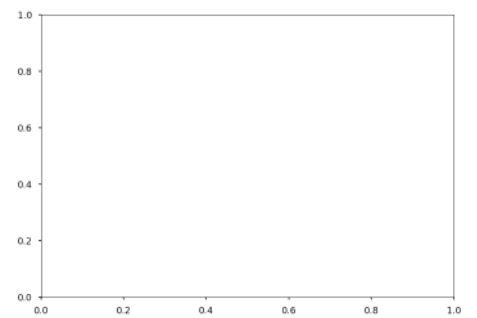
'bmh'



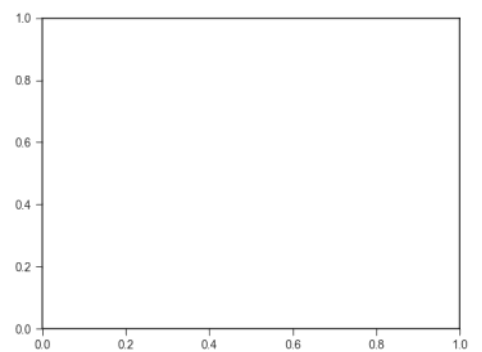
'classic'



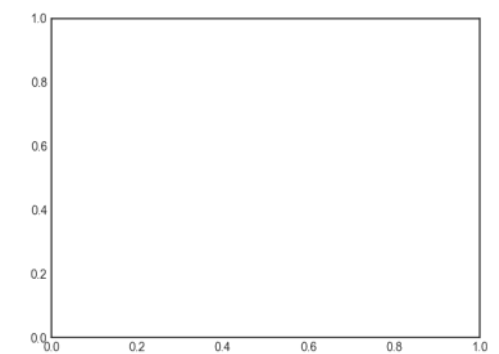
'seaborn-talk'



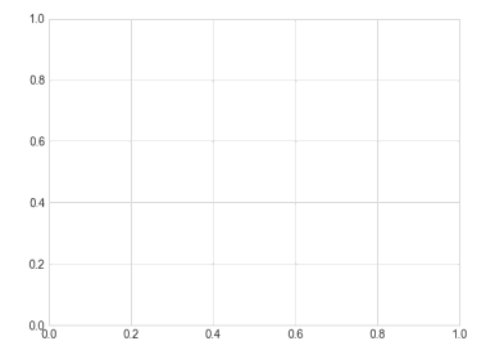
'seaborn-ticks'



'seaborn-white'

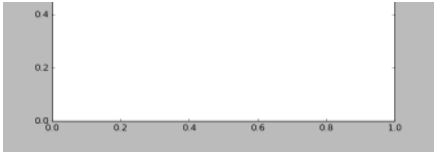


'seaborn-whitegrid'

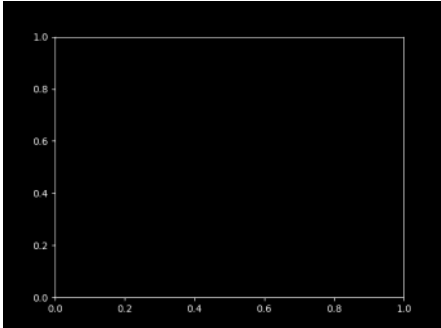


'tableau-colorblind10'

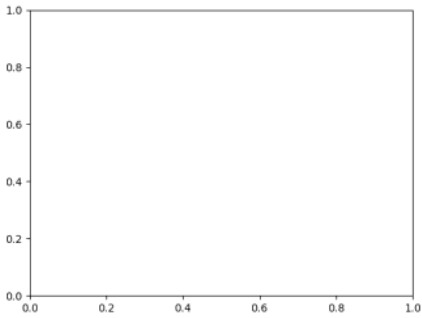




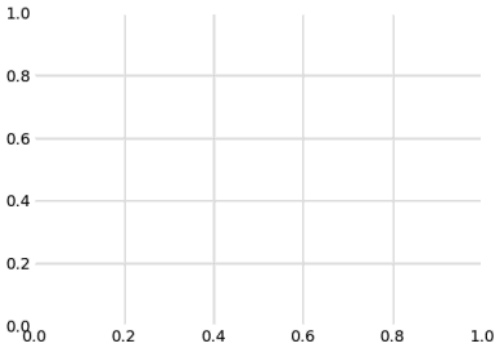
'dark_background'



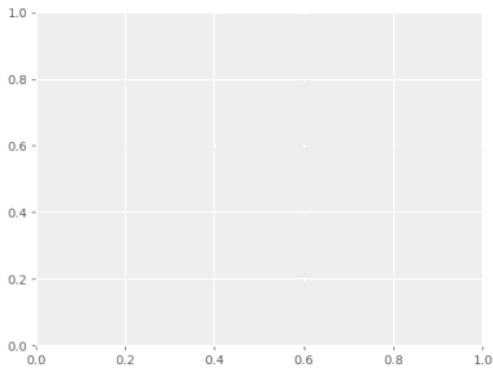
'fast'



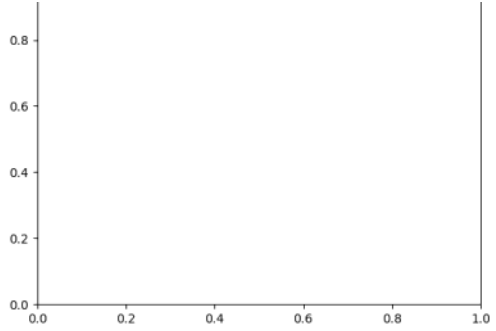
'fivethirtyeight'

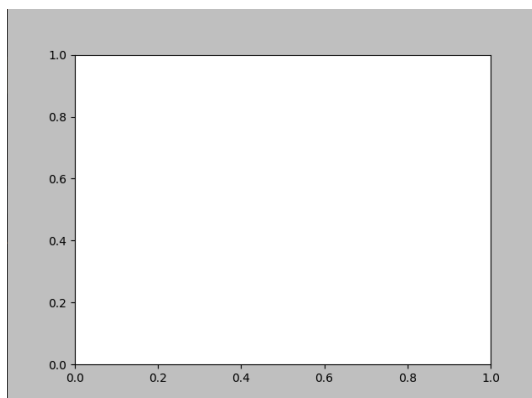


'ggplot'

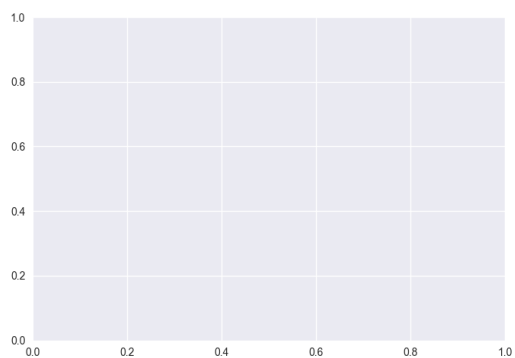


'grayscale'

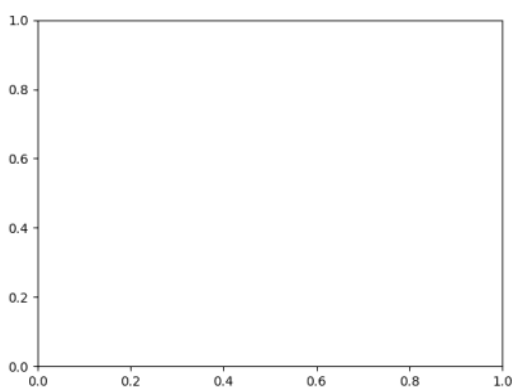




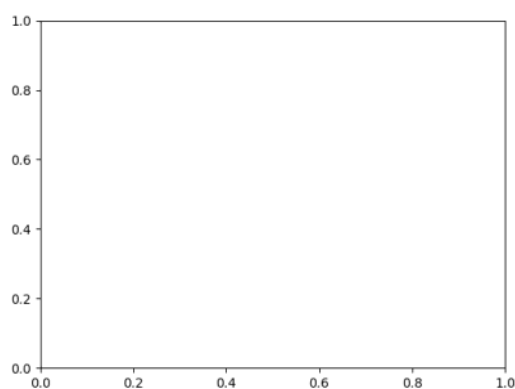
'seaborn'



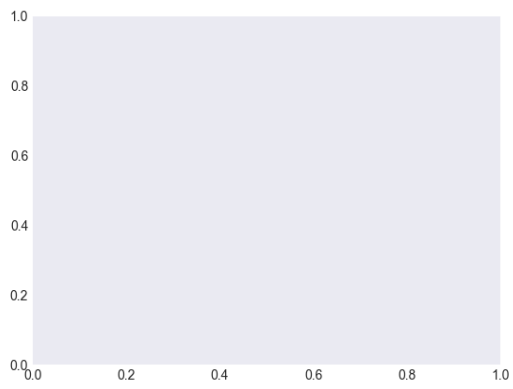
'seaborn-bright'



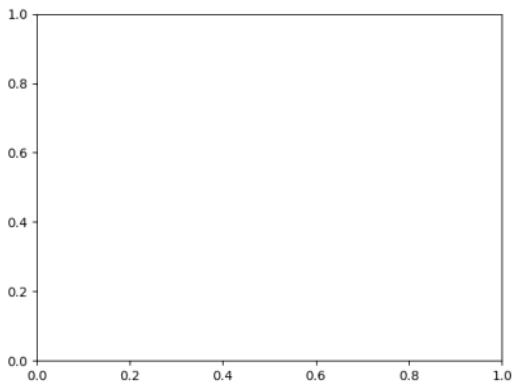
'seaborn-colorblind'



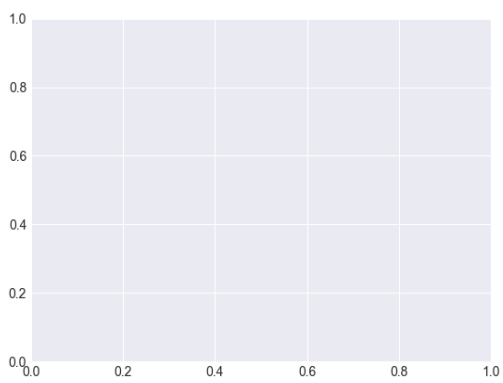
'seaborn-dark'



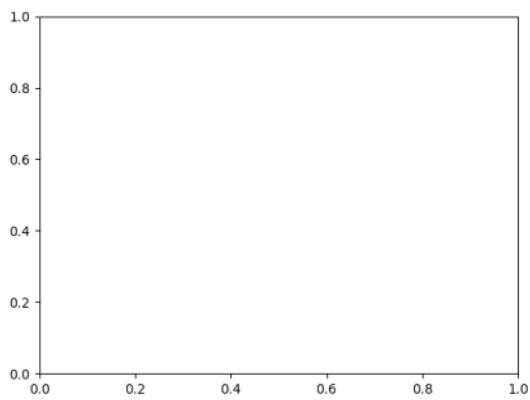
'seaborn-dark-palette'



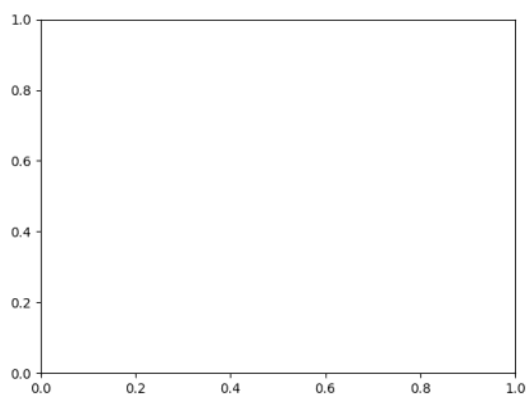
'seaborn-darkgrid'



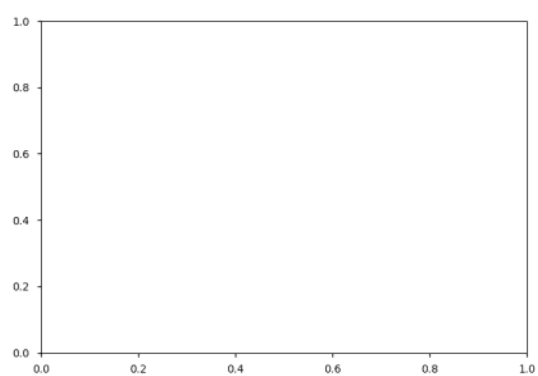
'seaborn-deep'



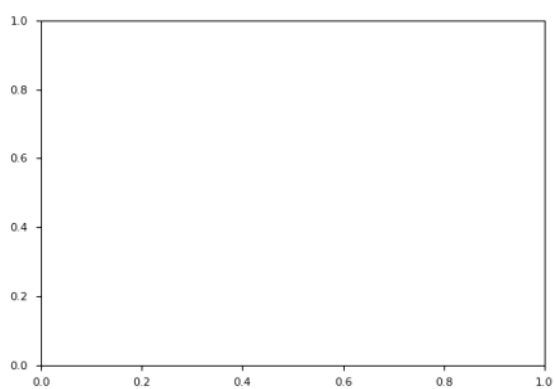
'seaborn-muted'



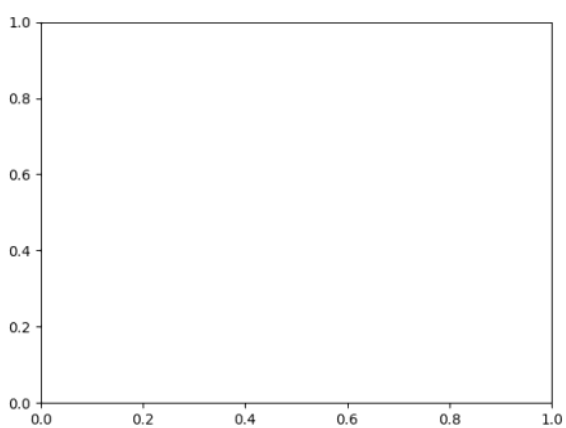
'seaborn-notebook'



'seaborn-paper'



'seaborn-pastel'



'seaborn-poster'

