

Python数据分析之 Matplotlib

1. Matplotlib基础

Matplotlib 是一个Python 的2D图形包， pyplot 封装了很多画图的函数。

```
In [1]: import matplotlib.pyplot as plt  
       import numpy as np
```

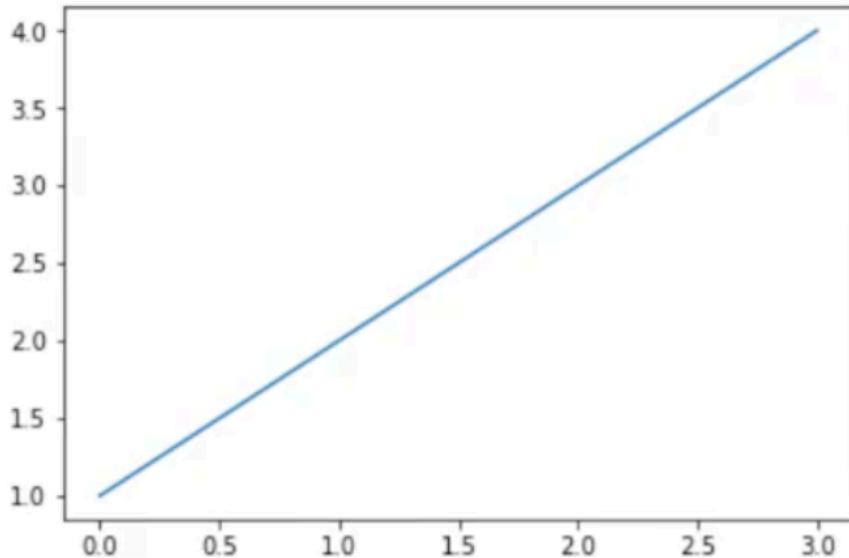
- plt.show()函数

默认情况下，matplotlib.pyplot 不会直接显示图像，只有调用 plt.show()函数时，图像才会显示出来。

- plt.plot()函数——可以用来绘线型图

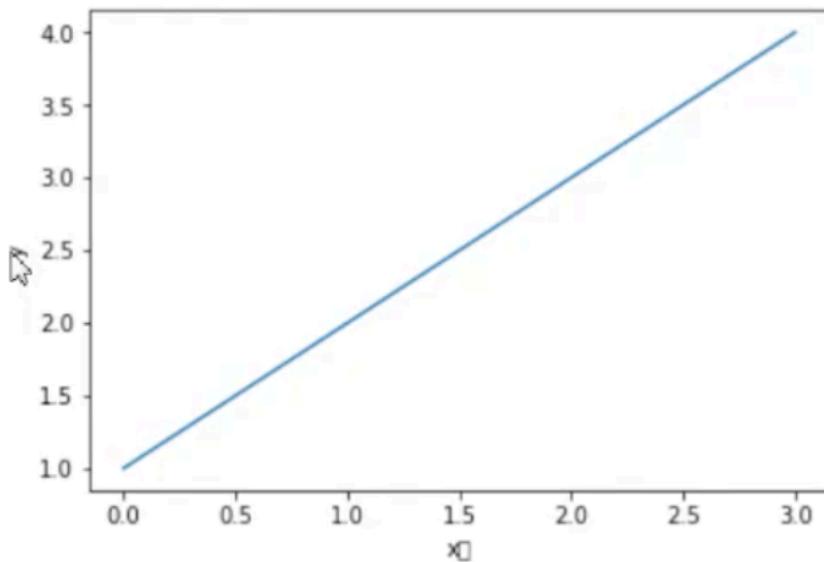
```
In [3]: plt.plot([1, 2, 3, 4])
```

```
Out[3]: [
```



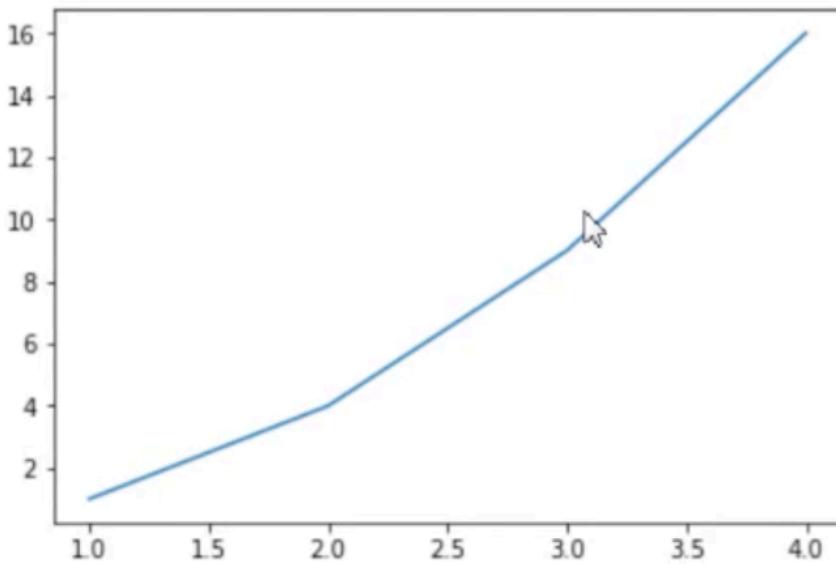
```
In [4]: plt.plot([1, 2, 3, 4])  
plt.ylabel('y')  
plt.xlabel('x轴')
```

Out[4]: Text(0.5, 0, 'x轴')



基本用法

```
In [6]: plt.plot([1, 2, 3, 4], [1, 4, 9, 16])  
plt.show()
```



字符参数——和MATLAB中类似，可以用字符来指定绘图的格式：

表示颜色的字符参数有：

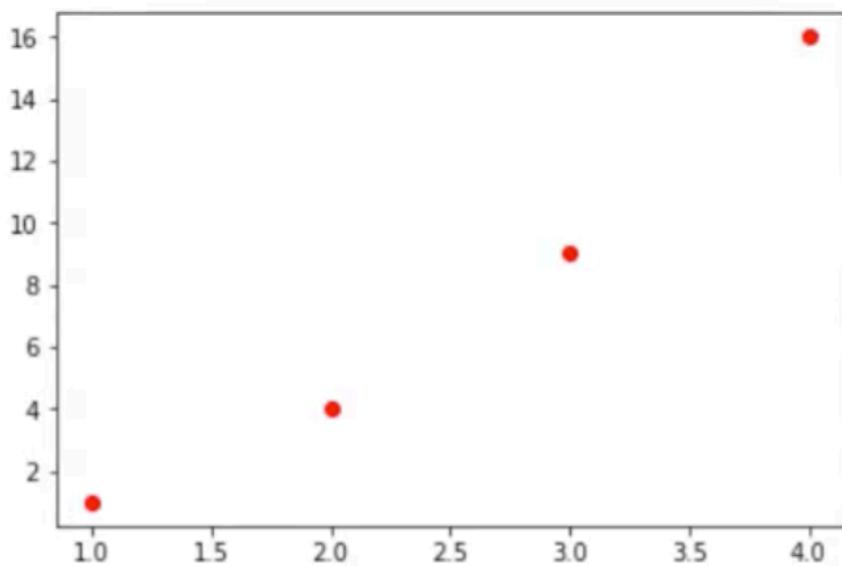
| 字符 | 颜色 |
|-----|--------------|
| 'b' | 蓝色 , blue |
| 'g' | 绿色 , green |
| 'r' | 红色 , red |
| 'c' | 青色 , cyan |
| 'm' | 品红 , magenta |
| 'y' | 黄色 , yellow |
| 'k' | 黑色 , black |
| 'w' | 白色 , white |

表示类型的字符参数有：

| 字符 | 类型 | 字符 | 类型 |
|-------|-------|------|-------|
| '-' | 实线 | '--' | 虚线 |
| '-..' | 虚点线 | '::' | 点线 |
| '..' | 点 | ',' | 像素点 |
| 'o' | 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 | | |

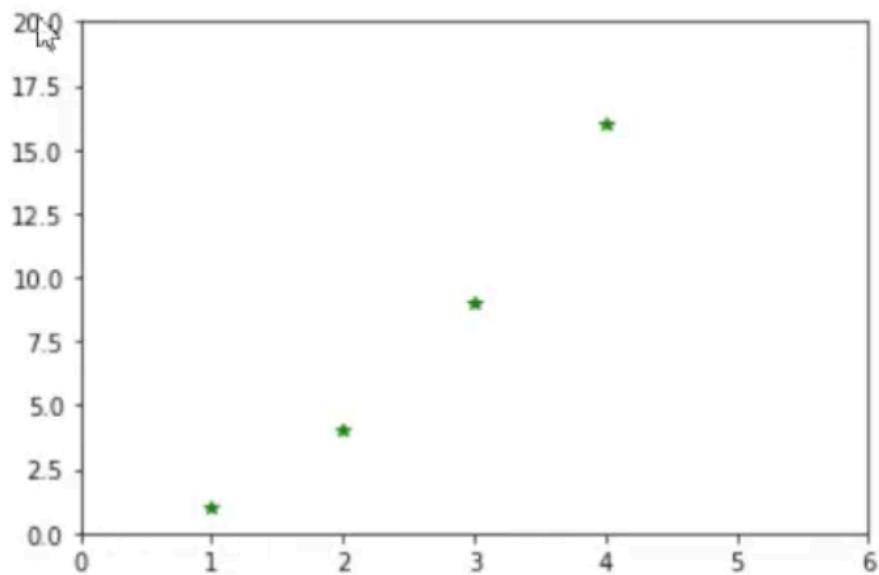
如：画出红色圆点：

```
In [8]: plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'ro')
plt.show()
```



显示范围——与MATLAB类似，使用axis函数指定坐标轴显示的范围：
plt.axis([xmin,xmax,ymin,ymax])

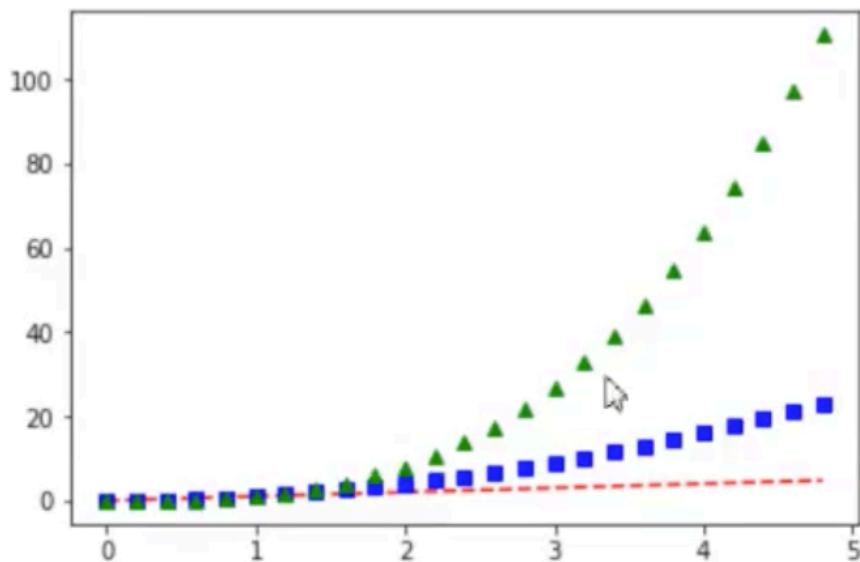
```
In [10]: plt.plot([1, 2, 3, 4], [1, 4, 9, 16], 'g*')
plt.axis([0, 6, 0, 20])
plt.show()
```



传入 Numpy 数组——

在一个图里面画多条线

```
In [13]: t=np.arange(0., 5., 0.2)
         plt.plot(t, t, 'r--',
                    t, t**2, 'bs',
                    t, t**3, 'g^')
         plt.show()
```



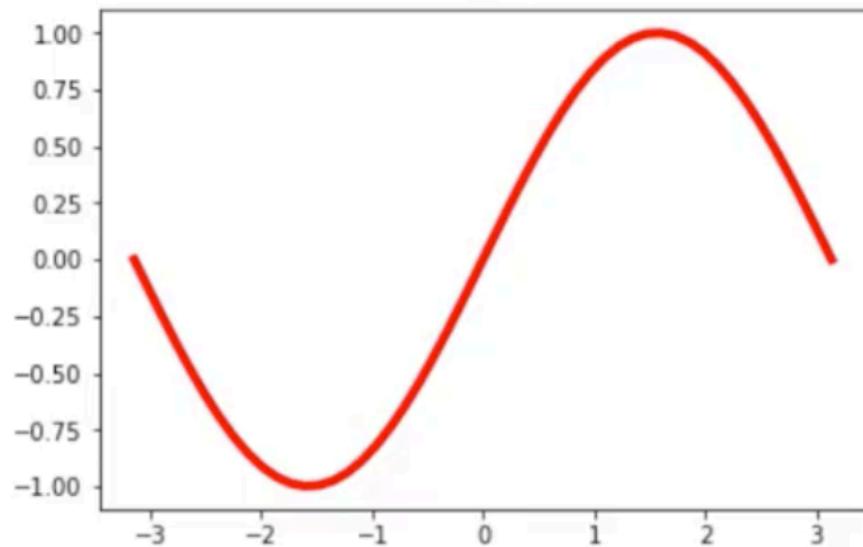
传入多组数据

上图通过plot函数传入数据，并通过传入(x, y, format_str)多组参数，使它们在同一张图上显示。

线条属性

上面通过字符串来控制线条属性，还可以通过关键词来改变线条的属性。
如 linewidth 可以改变线条的宽度，color 可以改变线条的颜色。

```
In [17]: x=np.linspace(-np.pi,np.pi)
y=np.sin(x)
plt.plot(x,y,linewidth=4.0,color='r')
plt.show()
```



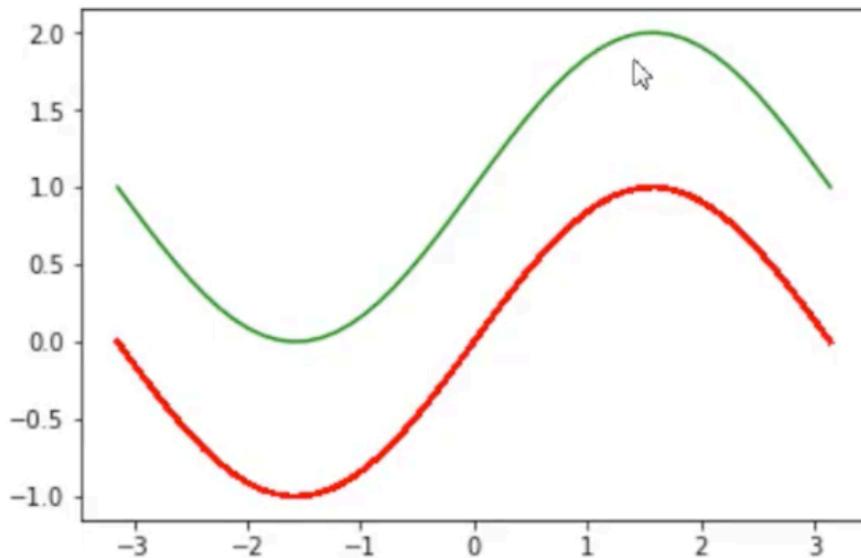
使用 plt.plot()的返回值来设置线条属性

plot函数返回一个Line2D对象组成的列表，每个对象代表输入的一对组合。

如：

```
line1, line2 为两个Line2D对象
line1, line2 = plt.plot(x1, y1, x2, y2)
返回3个Line2D对象组成的列表
lines = plt.plot(x1, y1, x2, y2, x3, y3)
```

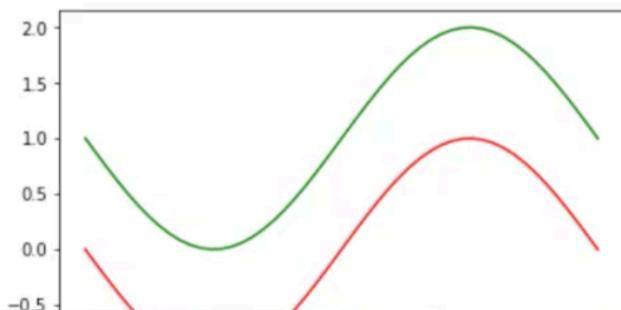
```
In [31]: line1, line2=plt.plot(x, y, 'r-', x, y+1, 'g-')
line1.set_antialiased(False)
plt.show()
```



```
In [32]: line=plt.plot(x, y, 'r-', x, y+1, 'g-')
line.set_antialiased(False)
plt.show()
```

```
AttributeError                                 Traceback (most recent call last)
<ipython-input-32-dbca6348dfa5> in <module>()
      1 line=plt.plot(x, y, 'r-', x, y+1, 'g-')
----> 2 line.set_antialiased(False)
      3 plt.show()
```

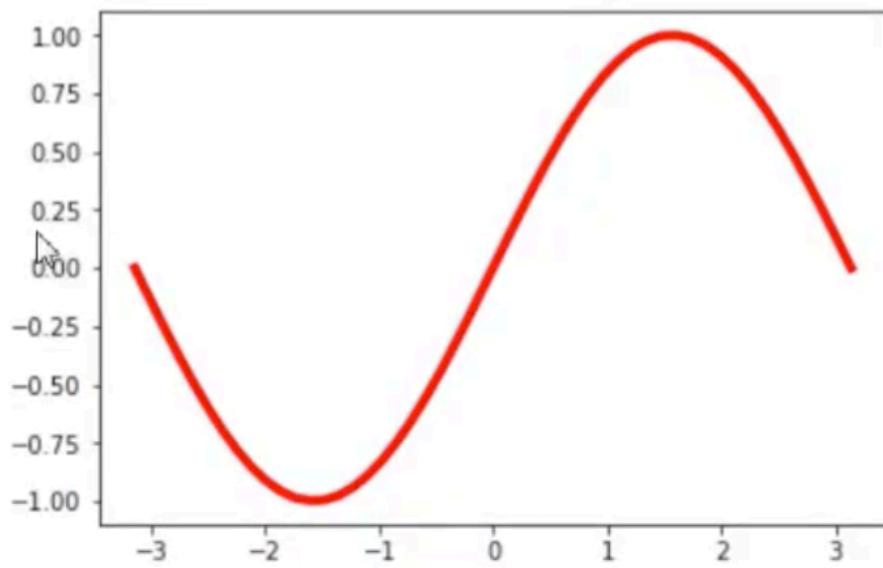
```
AttributeError: 'list' object has no attribute 'set_antialiased'
```



- plt.setp()修改线条性质（更方便）

```
In [38]: line=plt.plot(x, y)
#plt.setp(line, color='g', linewidth=4)
plt.setp(line, 'color', 'r', 'linewidth', 4)
```

```
Out[38]: [None, None]
```



- 子图

figure() 函数会产生一个特定编号为num的图： plt.figure(num)， 默认num=1， 即默认生成一副图像。

subplot 可以在一副图中生成多个子图， 其参数为： plt.subplot(numrows, numcols, fignum)

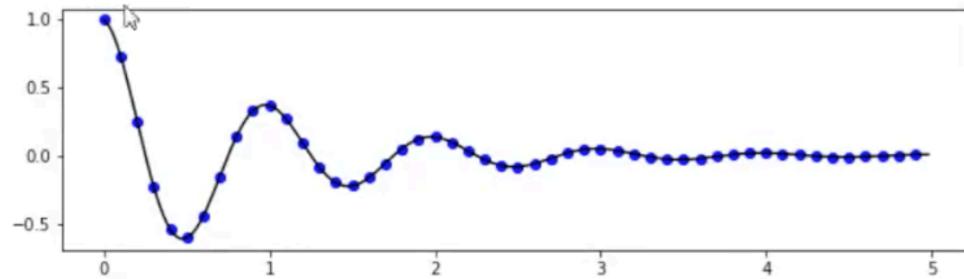
当 numrows * numcols < 10 时， 中间的逗号可以省略， 即 plt.subplot(211) 相当于 plt.subplot(2,1,1)

```
In [41]: def f(t):
    return np.exp(-t)*np.cos(2*np.pi*t)

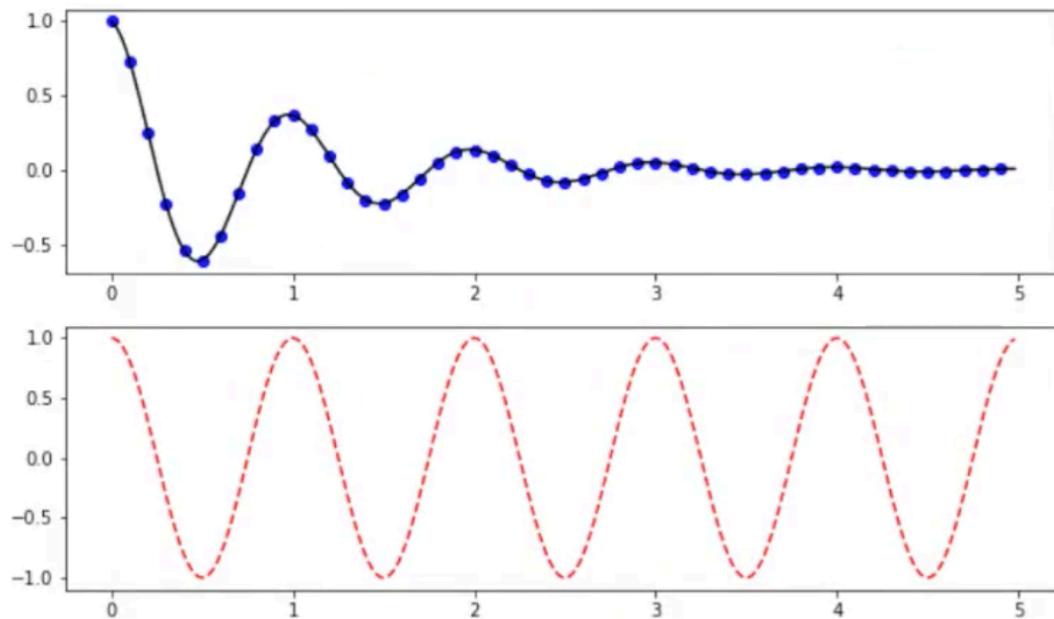
t1=np.arange(0, 0.5, 0.01)
t2=np.arange(0, 0.5, 0.02)

plt.figure(figsize=(10, 6))
plt.subplot(211)
plt.plot(t1, f(t1), 'bo', t2, f(t2), 'k')
```

```
Out[41]: [,
<matplotlib.lines.Line2D at 0xa2ce588>]
```



```
plt.subplot(212)
plt.plot(t2, np.cos(2*np.pi*t2), 'r--')
plt.show()
```



2. 电影数据绘图

对电影数据进行可视化分析

```
In [45]: import warnings  
warnings.filterwarnings('ignore')
```

```
In [46]: import pandas as pd  
import numpy as np  
import matplotlib.pyplot as plt
```

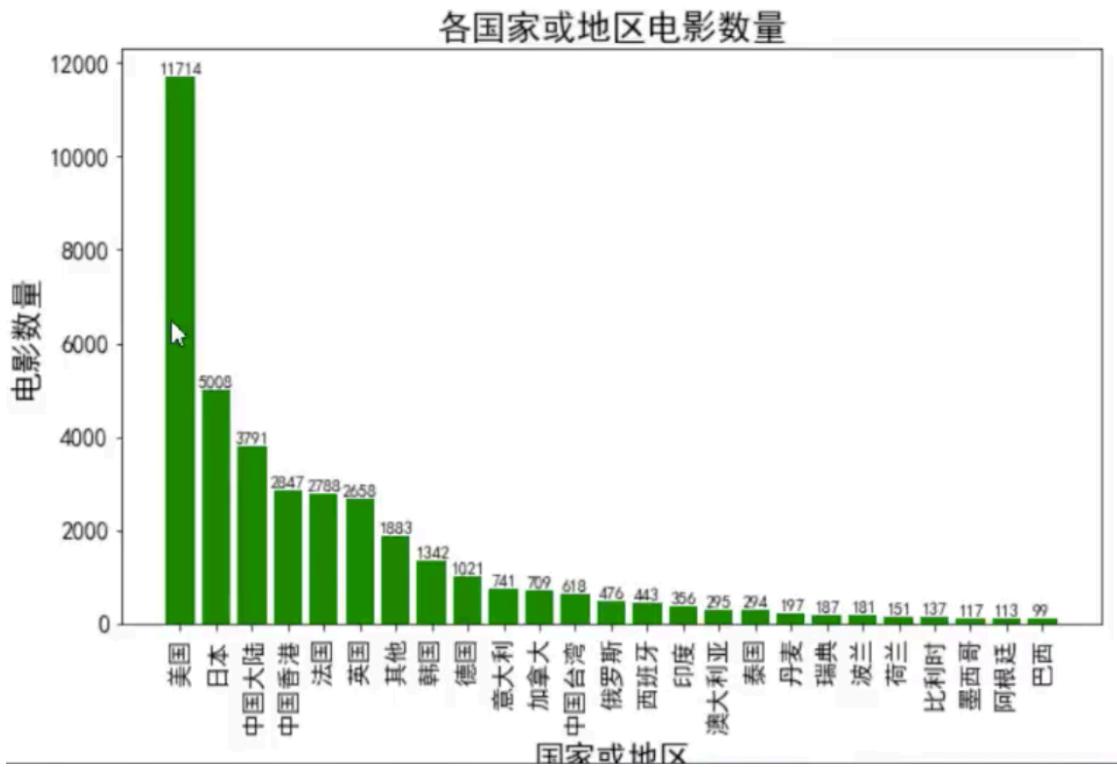
```
In [47]: plt.rcParams['font.sans-serif']=['SimHei']  
plt.rcParams['axes.unicode_minus']=False
```

```
In [ ]: df=pd.read_excel('movie_data3.xlsx')
```

(1) 绘制每个国家或地区的电影数量的柱状图

```
In [ ]: data=df['产地'].value_counts()
```

```
In [63]: x=data.index  
y=data.values  
  
plt.figure(figsize=(10, 6))  
plt.bar(x, y, color='g')  
  
plt.title('各国家或地区电影数量', fontsize=20)  
plt.xlabel('国家或地区', fontsize=18)  
plt.ylabel('电影数量', fontsize=18)  
plt.tick_params(labelsize=14) I  
plt.xticks(rotation=90)  
  
for a, b in zip(x, y):  
    plt.text(a, b+10, b, ha='center', va='bottom', fontsize=10)  
  
plt.show()
```



(2) 绘制每年上映的电影数量的曲线图 (1888–2015年)

```
In [65]: data=df['年代'].value_counts()
data=data.sort_index()[:-1]
data
```

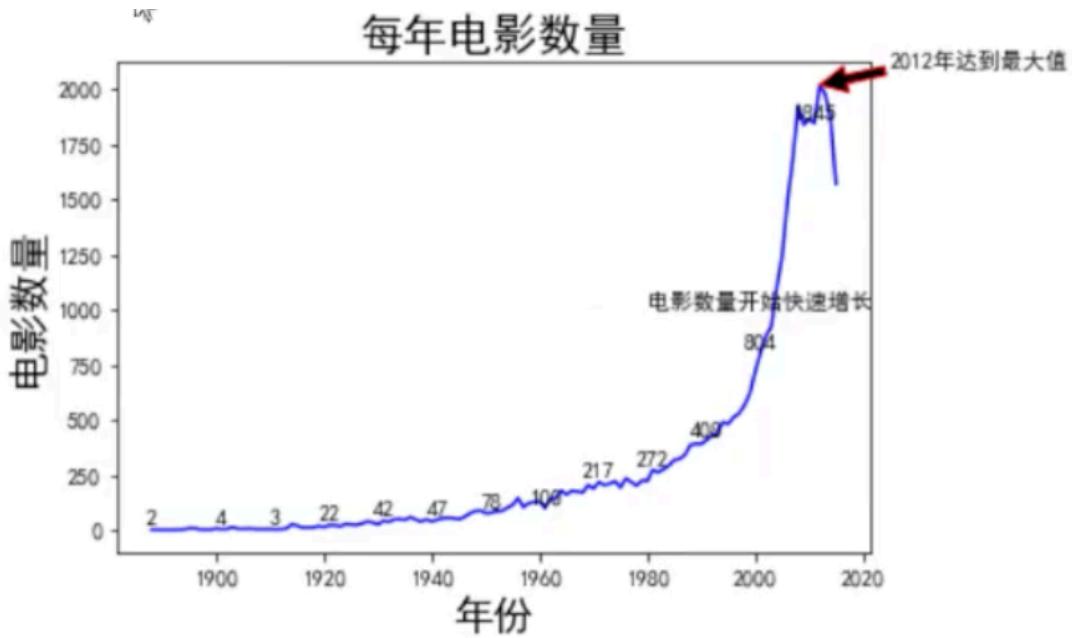
```
In [72]: x=data.index
y=data.values

plt.plot(x,y,color='b')
plt.title('每年电影数量', fontsize=20)
plt.ylabel('电影数量', fontsize=18)
plt.xlabel('年份', fontsize=18)

for a,b in zip(x[::10],y[::10]):
    plt.text(a,b+10,b,ha='center', va='bottom', fontsize=10)

plt.annotate('2012年达到最大值', xy=(2012,data[2012]), xytext=(2025,2100), arrowprops=dict(facecolor='black', edgecolor='red'))

plt.text(1950,1000,'电影数量开始快速增长')
plt.show()
```



上图总结——使用了 xlabel , ylabel , title , text 方法设置了文字，其中：

- xlabel: x 轴标注
- ylabel: y 轴标注
- title: 图形标题
- text: 在指定位置放入文字

输入特殊符号支持使用 Tex 语法，用 \$<some Tex code>\$ 隔开。

除了使用 text 在指定位置标上文字之外，还可以使用 annotate 函数进行注释，annotate主要有两个参数：

- xy: 注释位置
- xytext: 注释文字位置

(3) 根据电影的长度绘制饼图

函数原型:

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

参数: x (每一块)的比例，如果sum(x) > 1会使用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 是否阴影
labeldistance label绘制位置,相对于半径的比例, 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
radius 控制饼图半径

返回值: 如果没有设置autopct,返回(patches, texts)

如果设置autopct,返回(patches, texts, autotexts)

```
In [74]: data=pd.cut(df['时长'], [0, 60, 90, 110, 1000]).value_counts()
data
```

```
Out[74]: (90, 110]      13203
(0, 60]        9884
(60, 90]       7662
(110, 1000]    7417
Name: 时长, dtype: int64
```

```
In [81]: y=data.values
#y=y/sum(y)

plt.figure(figsize=(7,7))
plt.title('电影时长占比', fontsize=15)
patches, l_text, p_text=plt.pie(y, labels=data.index, autopct='%.1f %%', colors='bygr', startangle=90)

for i in p_text:
    i.set_size(15)
    i.set_color('w')
for i in l_text:
    i.set_size(15)
    i.set_color('r')
plt.legend()
plt.show()
```



(4)根据电影的评分绘制评率分布直方图

直方图 (Histogram) 又称质量分布图，是一种统计报告图，可以被归一化以显示“相对”频率。

hist的参数非常多，但常用的就这六个，只有第一个是必须的，后面可选

arr: 需要计算直方图的一维数组

bins: 直方图的柱数，可选项，默认为10

normed: 是否将得到的直方图向量归一化。默认为0

facecolor: 直方图颜色

edgecolor: 直方图边框颜色

alpha: 透明度

histtype: 直方图类型，'bar', 'barstacked', 'step', 'stepfilled'

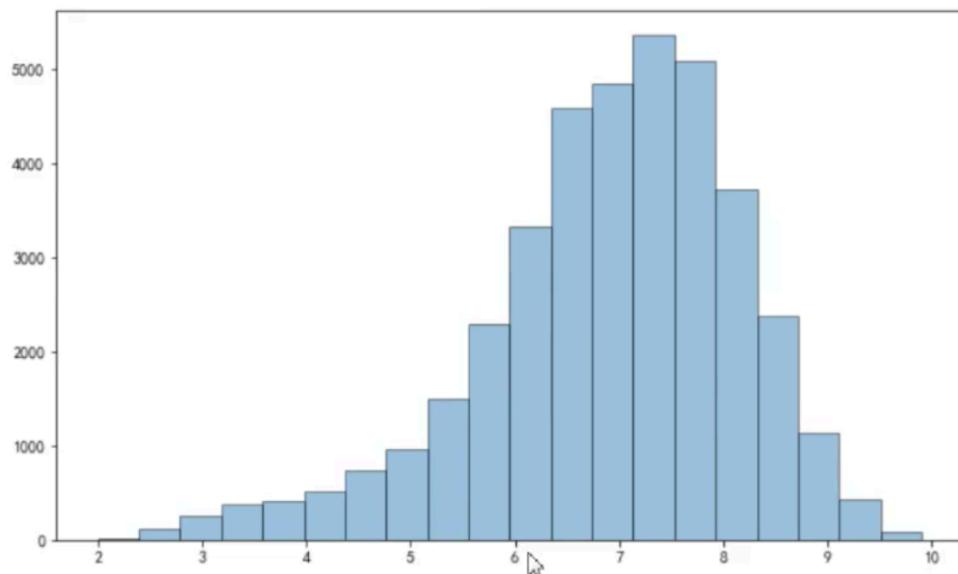
返回值：

n: 直方图向量，是否归一化由参数normed设定

bins: 返回各个bin的区间范围

patches: 返回每个bin里面包含的数据，是一个list

```
In [96]: plt.figure(figsize=(10, 6))
plt.hist(df['评分'], bins=20, edgecolor='k', alpha=0.5)
plt.show()
```



上图可以发现，电影的评分是服从一个右偏的正态分布的。

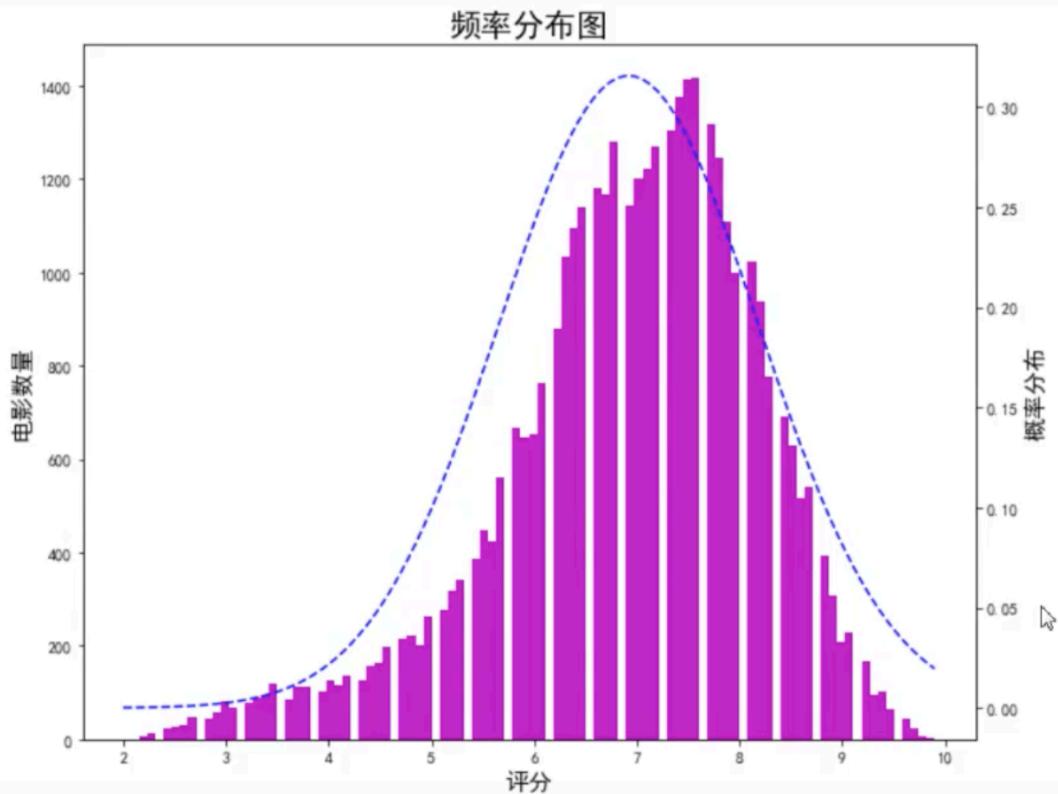
- 双轴图的画法

```
In [7]: import matplotlib.mlab as mlab
```

```
In [ ]: fig=plt.figure(figsize=(10,8))
ax1=fig.add_subplot(111)
n, bins, patches=ax1.hist(df['评分'], bins=100, color='m')

ax1.set_ylabel('电影数量', fontsize=15)
ax1.set_xlabel('评分', fontsize=15)
ax1.set_title('频率分布图', fontsize=20)

y=mlab.normpdf(bins, df['评分'].mean(), df['评分'].std())
ax2=ax1.twinx()
ax2.plot(bins, y, 'b--')
ax2.set_ylabel('概率分布', fontsize=15)
plt.show()
```



(5) 根据电影市场和电影评分绘制散点图
散点图通常用于比较跨类别的聚合数据

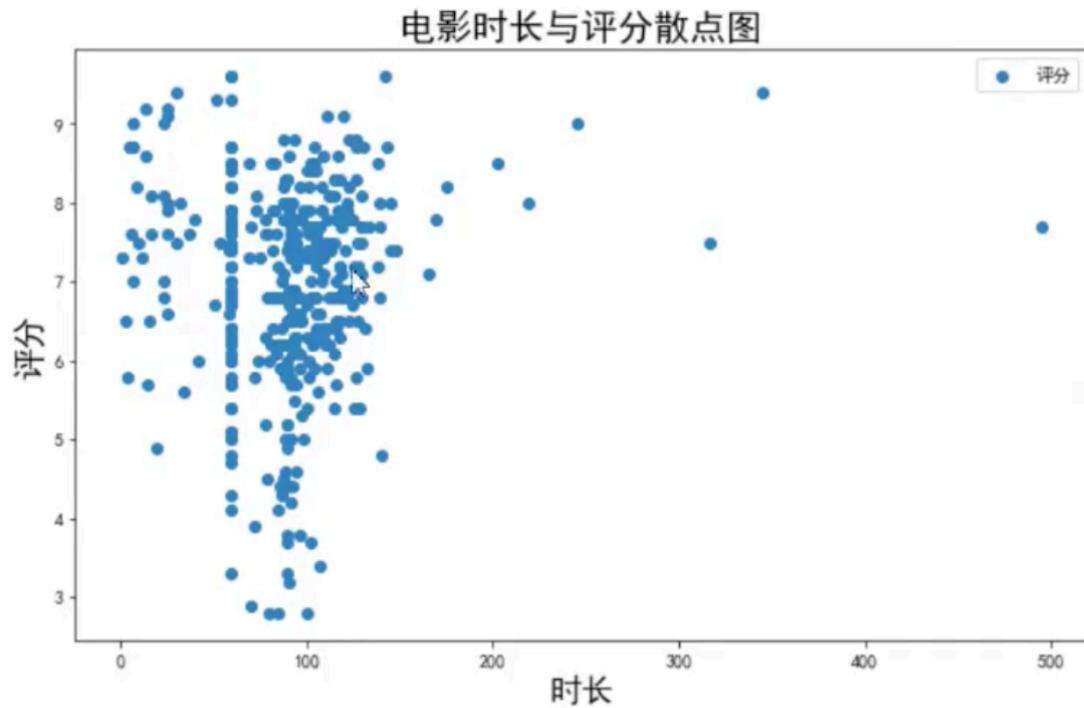
```
In [*:]: x=df['时长']
y=df['评分']

plt.figure(figsize=(10, 8))
plt.scatter(x, y)
plt.legend()
plt.title('电影时长与评分散点图', fontsize=20)
plt.xlabel('时长', fontsize=18)
plt.ylabel('评分', fontsize=18)
plt.show()
```

选择原来的1%

```
In [9]: x=df['时长'][::100]
y=df['评分'][::100]

plt.figure(figsize=(10, 6))
plt.scatter(x, y)
plt.legend()
plt.title('电影时长与评分散点图', fontsize=20)
plt.xlabel('时长', fontsize=18)
plt.ylabel('评分', fontsize=18)
plt.show()
```



- marker 属性

设置散点的形状

| marker | description | 描述 |
|--------|----------------|-----------|
| "." | point | 点 |
| "，" | pixel | 像素 |
| "o" | circle | 圆 |
| "v" | triangle_down | 倒三角形 |
| "^" | triangle_up | 正三角形 |
| "<" | triangle_left | 左三角形 |
| ">" | triangle_right | 右三角形 |
| "1" | tri_down | tri_down |
| "2" | tri_up | tri_up |
| "3" | tri_left | tri_left |
| "4" | tri_right | tri_right |
| "8" | octagon | 八角形 |
| "s" | square | 正方形 |
| "p" | pentagon | 五角 |
| "*" | star | 星星 |
| "h" | hexagon1 | 六角 1 |
| "H" | hexagon2 | 六角 2 |
| "+" | plus | 加号 |
| "x" | x | x 号 |
| "D" | diamond | 钻石 |
| "d" | thin_diamond | 细钻 |
| " " | vline | v 线 |
| "_" | hline | H 线 |

(6) 绘制各个地区的评分箱型图

是一种用作显示一组数据分散情况资料的统计图，常见于品质管理。

一般计算过程

- (1) 计算上四分位数 (Q3) , 中位数 , 下四分位数 (Q1)
- (2) 计算上四分位数和下四分位数之间的差值 , 即四分位数差 (IQR , interquartile range) $Q3-Q1$
- (3) 绘制箱线图的上下范围 , 上限为上四分位数 , 下限为下四分位数。在箱子内部中位数的位置绘制横线
- (4) 大于上四分位数1.5倍四分位数差的值 , 或者小于下四分位数1.5倍四分位数差的值 , 划为异常值 (outliers)
- (5) 异常值之外 , 最靠近上边缘和下边缘的两个值处 , 画横线 , 作为箱线图的触须
- (6) 极端异常值 , 即超出四分位数差3倍距离的异常值 , 用实心点表示 ; 较为温和的异常值 , 即处于1.5倍-3倍四分位数差之间的异常值 , 用空心点表示
- (7) 为箱线图添加名称 , 数轴等

参数详解

```
plt.boxplot(x, notch=None, sym=None, vert=None,  
            whis=1.5, positions=None, widths=None,  
            patch_artist=None, meanline=None, showmeans=None,  
            showcaps=None, showbox=None, showfliers=None,  
            boxprops=None, labels=None, flierprops=None,  
            medianprops=None, meanprops=None,  
            capprops=None, whiskerprops=None)
```

x : 指定要绘制箱线图的数据 ;

notch : 是否是凹口的形式展现箱线图 , 默认非凹口 ;

sym : 指定异常点的形状 , 默认为+号显示 ;

vert : 是否需要将箱线图垂直摆放 , 默认垂直摆放 ;

whis : 指定上下须与上下四分位的距离 , 默认为1.5倍的四分位差 ;

positions : 指定箱线图的位置 , 默认为[0,1,2...] ;

widths : 指定箱线图的宽度 , 默认为0.5 ;

patch_artist : 是否填充箱体的颜色 ;

meanline : 是否用线的形式表示均值 , 默认用点来表示 ;

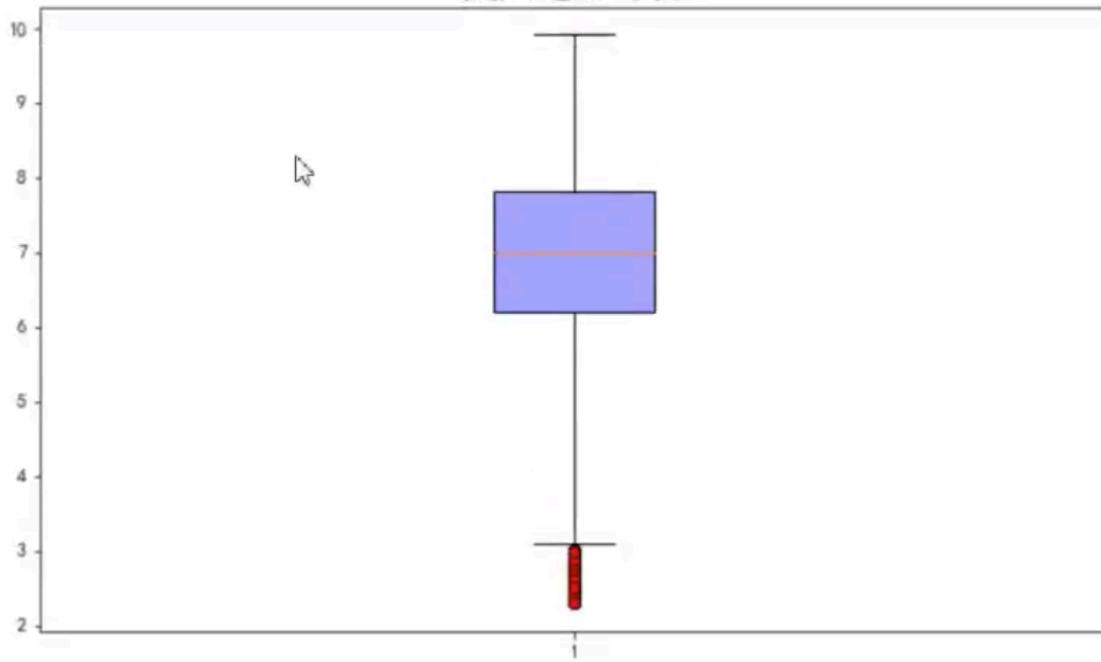
meanline : 是否用线的形式表示均值，默认用点来表示；
showmeans : 是否显示均值，默认不显示；
showcaps : 是否显示箱线图顶端和末端的两条线，默认显示；
showbox : 是否显示箱线图的箱体，默认显示；
showfliers : 是否显示异常值，默认显示；
boxprops : 设置箱体的属性，如边框色，填充色等；
labels : 为箱线图添加标签，类似于图例的作用；
flierprops : 设置异常值的属性，如异常点的形状、大小、填充色等；
medianprops : 设置中位数的属性，如线的类型、粗细等；
meanprops : 设置均值的属性，如点的大小、颜色等；
capprops : 设置箱线图顶端和末端线条的属性，如颜色、粗细等；
whiskerprops : 设置须的属性，如颜色、粗细、线的类型等

美国电影评分的箱线图

```
In [21]: data=df[df.产地=='美国']['评分']

plt.figure(figsize=(10,6))
plt.boxplot(data,whis=2,flierprops={'marker':'o','markerfacecolor':'r','color':'k'},
            patch_artist=True,boxprops={'color':'k','facecolor':'#9999ff'})
plt.title('美国电影评分',fontsize=20)
plt.show()
```

美国电影评分



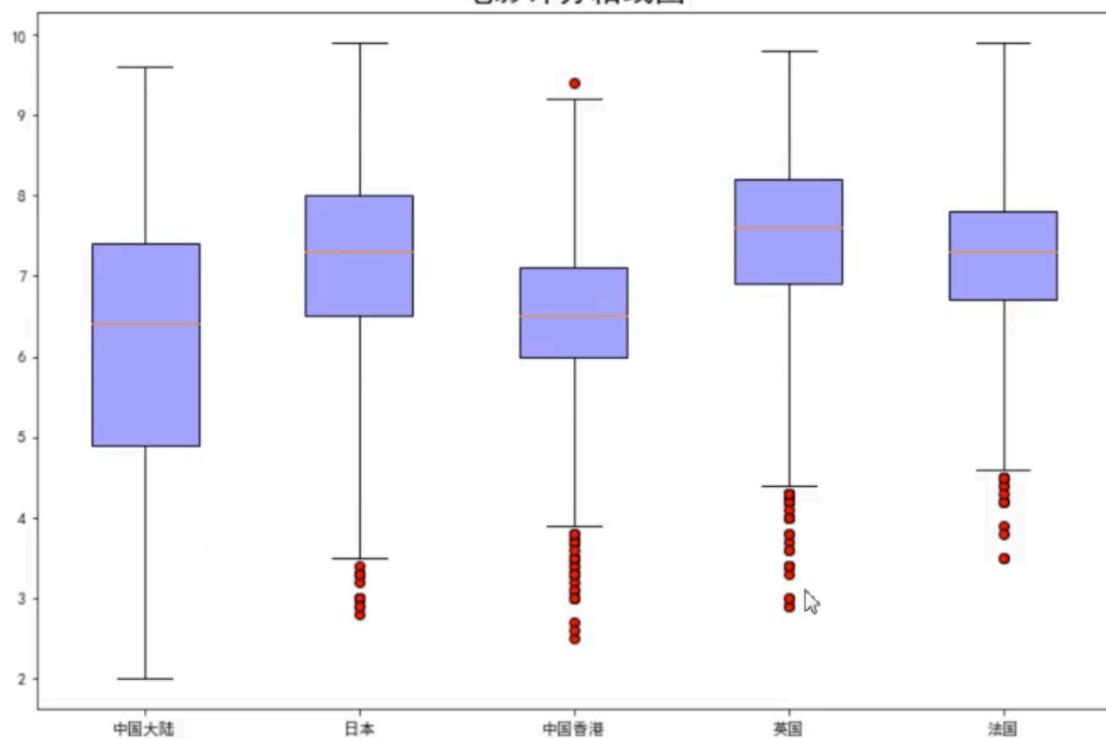
多组数据箱线图

```
In [22]: data1=df[df.产地=='中国大陆']['评分']
data2=df[df.产地=='日本']['评分']
data3=df[df.产地=='中国香港']['评分']
data4=df[df.产地=='英国']['评分']
data5=df[df.产地=='法国']['评分']

plt.figure(figsize=(12,8))
plt.boxplot([data1,data2,data3,data4,data5], labels=['中国大陆','日本','中国香港','英国','法国'],
            whis=2, flierprops={'marker':'o', 'markerfacecolor':'r', 'color':'k'},
            patch_artist=True, boxprops={'color':'k','facecolor':'#9999ff'})

plt.title('电影评分箱线图', fontsize=20)
plt.show()
```

电影评分箱线图



通过vert属性可以把图旋转过来（上面的红框）；
(下面的红框) 设置背景色和颜色透明度

```
plt.figure(figsize=(12, 8))
plt.boxplot([data1, data2, data3, data4, data5], labels=['中国大陆', '日本', '中国香港', '英国', '法国'],
            whis=2, flierprops={'marker':'o', 'markerfacecolor':'r', 'color':'k'},
            patch_artist=True, boxprops={'color':'k', 'facecolor':'#9999ff'},
            vert=False)
ax=plt.gca()
ax.patch.set_facecolor('gray')
ax.patch.set_alpha(0.3)

plt.title('电影评分箱线图', fontsize=20)
plt.show()
```

(7) 相关系数矩阵图- -热力图

准备数据

```
In [ ]: data=df[['投票人数', '评分', '时长']]
```

Pandas本身也封装了画图函数

画出各个属性之间的散点图，对角线是分布图

```
In [*]: %pylab inline
result=pd.scatter_matrix(data[::100],diagonal='kde',color='k',alpha=0.3,figsize=(10,10))
Populating the interactive namespace from numpy and matplotlib
```

seaborn是一个精简的Python库，可以创建具有统计意义的图表，能理解pandas的DataFrame类型
用corr取相关系数，abs取绝对值

```
In [ ]: import seaborn as sns
corr=data.corr()
corr=abs(corr)

fig=plt.figure(figsize=(10,8))
ax=fig.add_subplot(figsize=(10,8))
```

热力图绘制

```
ax=sns.heatmap(corr,vmax=1,vmin=0,annot=True,annot_kws={'size':13,'weight':'bold'},linewidths=0.05)
plt.xticks(fontsize=15)
plt.yticks(fontsize=15)
plt.show()
```

