介绍

学习目标：

1. 了解Matplotlib是什么
2. 知道为什么要进行数据可视化
3. 知道常见的图表类型

Matplotlib是什么？

Matplotlib是一个Python 绘图库，它可以在各种平台上以各种硬拷贝格式和交互式环境生成出具有出版品质的图形。Matplotlib可用于Python脚本，Python和IPython shell，Jupyter笔记本，Web应用程序服务器和四个图形用户界面工具包。

Matplotlib试图让简单的事情变得更简单，让无法实现的事情变得可能实现。 只需几行代码即可生成绘图，直方图， 功率谱，条形图，错误图，散点图等。

为了简单绘图，pyplot模块提供了类似于MATLAB的界面，特别是与IPython结合使用时。对于高级用户，您可以通过面向对象的界面或MATLAB用户熟悉的一组函数完全控制线条样式，字体属性，轴属性等。

点击查看：官网

为什么要进行数据可视化？

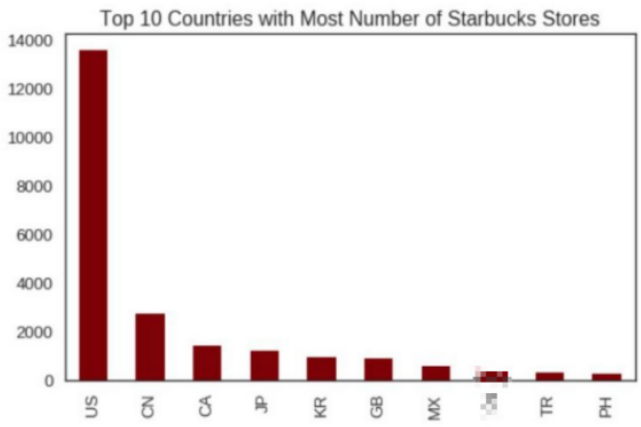
可视化是在整个数据挖掘的关键辅助工具，可以清晰的理解数据，从而调整我们的分析方法。

- **能将数据进行可视化，更直观的呈现**

- **使数据更加客观、更具说服力**

例如下面两个图为数字展示和图形展示：

```
US    13608
CN     2734
CA     1468
JP     1237
KR      993
GB      901
MX      579
[illegible]  394
TR      326
PH      298
Name: Country, dtype: int64
```

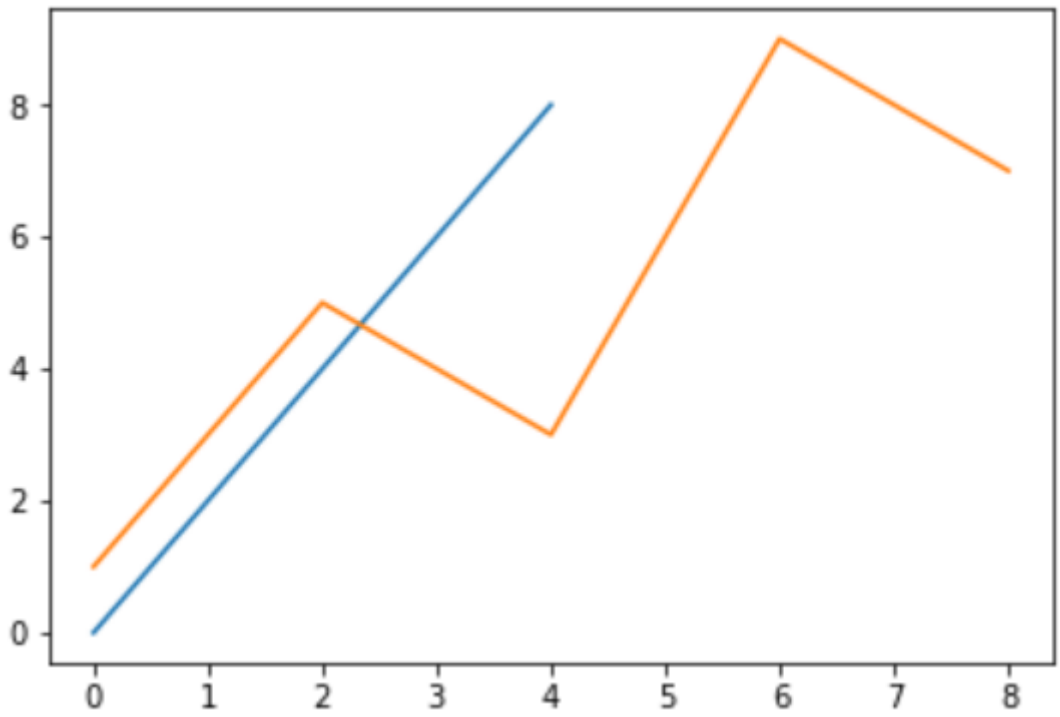


常见的图形种类

折线图

折线图：以折线的上升或下降来表示统计数量的增减变化的统计图

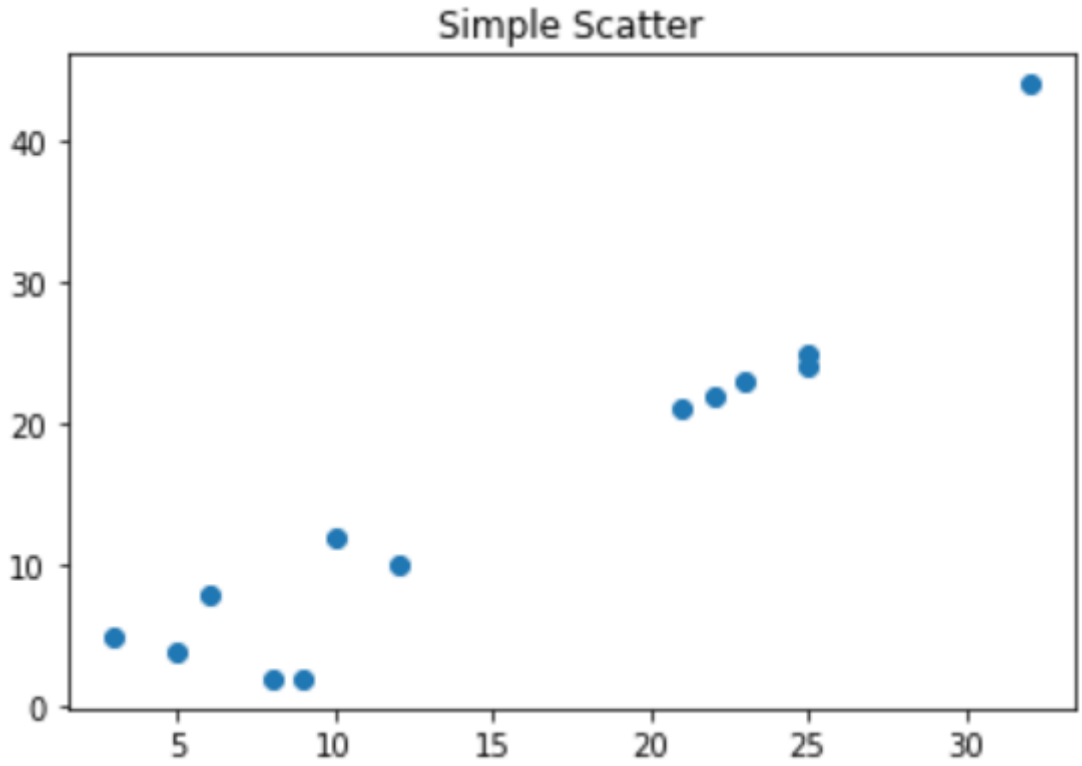
特点：**能够显示数据的变化趋势，反映事物的变化情况。（变化）**



散点图

用两组数据构成多个坐标点，考察坐标点的分布，判断两变量之间是否存在某种关联或总结坐标点的 分布模式。

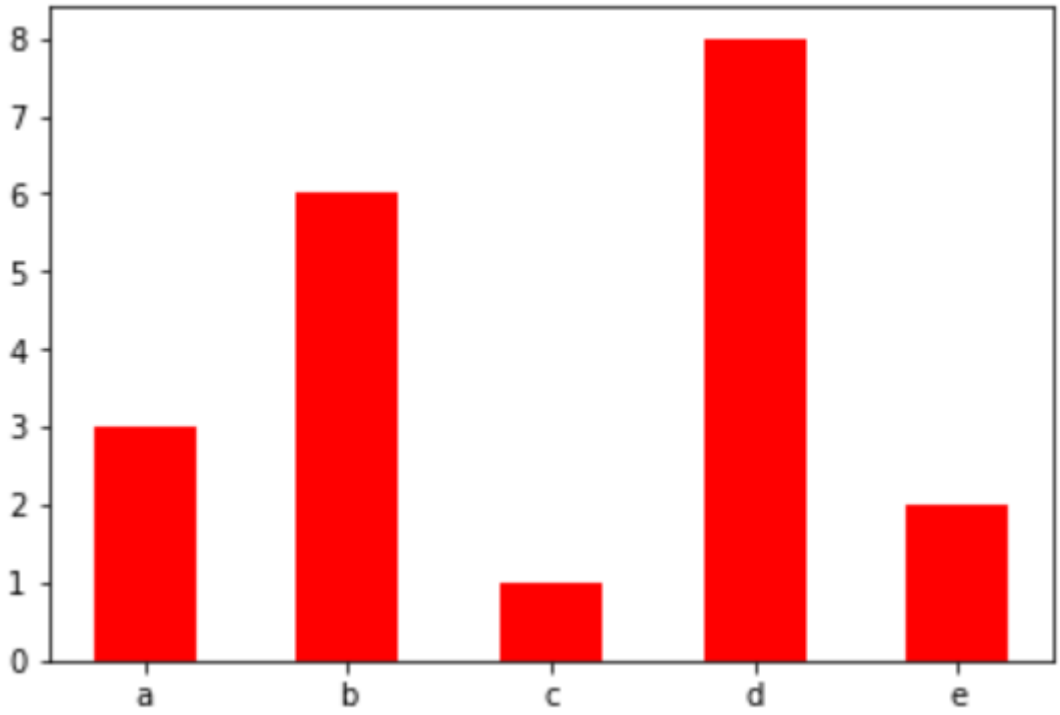
特点：判断变量之间是否存在数量关联趋势，展示离群点（分布规律）



柱状图

柱状图：排列在工作表的列或行中的数据可以绘制到柱状图中。

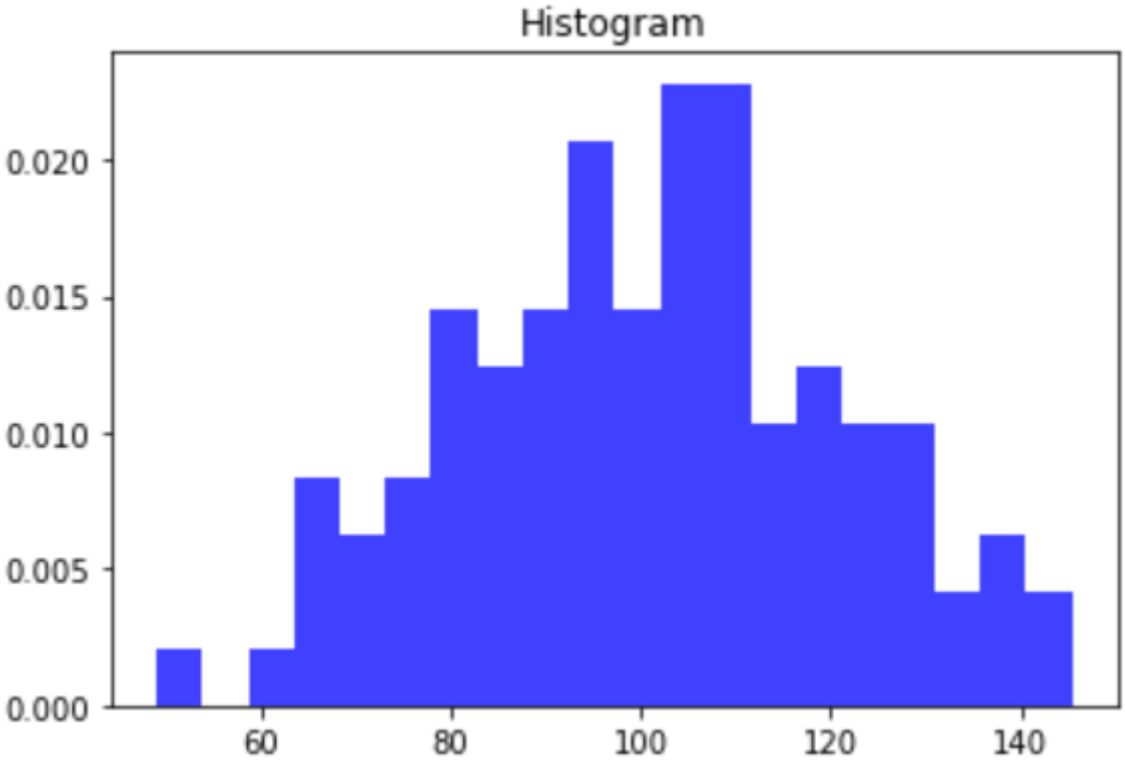
特点：绘制连离散的数据，能够一眼看出各个数据的大小，比较数据之间的差别。（统计/对比）



直方图

直方图：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范围，纵轴表示分布情况。

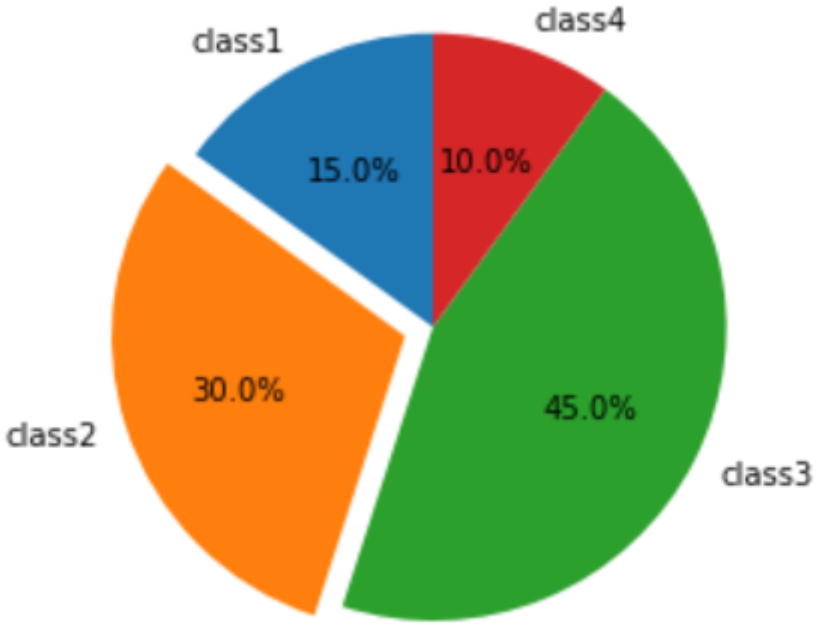
特点：绘制连续性的数据，展示一组或多组数据的分步情况（统计）



饼图

饼图：用于表示不同分类的占比情况，通过弧度大小来对比各种分类。

特点：分类数据的占比情况（占比）



安装

安装

安装Matplotlib非常简单，只需要执行以下指令即可。

```
pip install matplotlib
```

验证

安装完成之后，可以初步使用一下Matplotlib，验证一下是否安装没有问题。

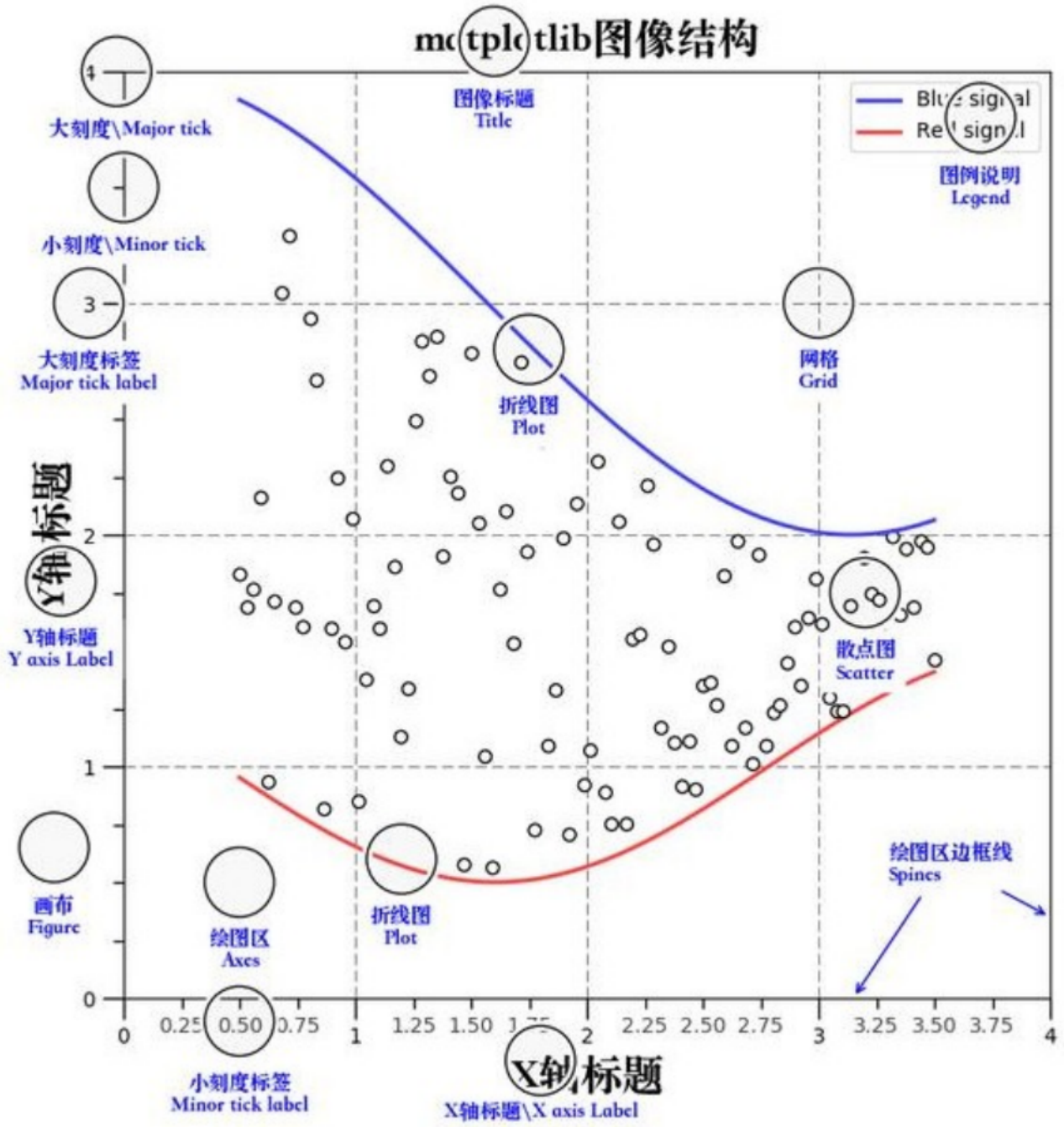
```
# 导入模块
import matplotlib.pyplot as plt

# 传入x和y，通过plot画图
plt.plot([1, 0, 9], [4, 5, 6])
# 在执行程序的时候展示图形
plt.show()
```

使用

学习目标：学会使用Matplotlib绘制常见的几种图形

基本概念

Matplotlib是一个绘图工具，为了方便理解后续的API，所以在绘图之前，我们先来了解一下绘图中的基本概念。



折线图

绘制折线图的API： `plt.plot(点的坐标)`

绘制基本的折线图

```
# 1. 绘制基本的折线图
from matplotlib import pyplot as plt

x = range(1,8) # x轴的位置
y = [17, 17, 18, 15, 11, 11, 13]
# 传入x和y, 通过plot画折线图
plt.plot(x,y)
plt.show()
```

折线的颜色和形状设置

```
"""
color='red' : 折线的颜色
alpha=0.5   : 折线的透明度(0-1)
linestyle='--' : 折线的样式
linewidth=3     : 折线的宽度—粗细
"""

from matplotlib import pyplot as plt
x = range(1,8)                                  # x轴的位置
y = [17, 17, 18, 15, 11, 11, 13]
# 传入x和y, 通过plot画折线图
plt.plot(x, y, color='red',alpha=0.5,linestyle='--',linewidth=3)
plt.show()
```

折点的样式

```
"""
折点形状选择:

character   description
================    ================================
``'-'``     solid line style
``'--'``    dashed line style
``'-.'``    dash-dot line style
```

```
``':'``     dotted line style
``'.'``     point marker
``','``     pixel marker
``'o'``     circle marker
``'v'``     triangle_down marker
``'^'``     triangle_up marker
``'<'``     triangle_left marker
``'>'``     triangle_right marker
``'1'``     tri_down marker
``'2'``     tri_up marker
``'3'``     tri_left marker
``'4'``     tri_right marker
``'s'``     square marker
``'p'``     pentagon marker
``' '``     star marker
``'h'``     hexagon1 marker
``'H'``     hexagon2 marker
``'+'``     plus marker
``'x'``     x marker
``'D'``     diamond marker
``'d'``     thin_diamond marker
``'|'``     vline marker
``'_'``     hline marker

"""

from matplotlib import pyplot as plt
x = range(1,8) # x轴的位置
y = [17, 17, 18, 15, 11, 11, 13]
# 传入x和y, 通过plot画折线图
plt.plot(x, y, marker='o')
plt.show()
```

设置保存图片的大小

```
from matplotlib import pyplot as plt
import random
x = range(2,26,2) # x轴的位置
y = [random.randint(15, 30) for i in x]

# 设置图片的大小
'''
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
'''
```

```
# 设置画布对象，figsize中对应的单位是英寸，dpi是每英寸有多少像素点
plt.figure(figsize=(20,8),dpi=80)

plt.plot(x,y)    # 传入x和y，通过plot画图
#plt.show()
#  保存(注意：要放在绘制的下面,并且plt.show()会释放figure资源，如果在显示图像之后保存图片将只能保存空图片。)
plt.savefig('./t1.png')

#    图片的格式也可以保存为svg这种矢量图格式，这种矢量图放在网页中放大后不会有锯齿
# plt.savefig('./t1.svg')
```

绘制x轴和y轴的刻度

```
from matplotlib import pyplot as plt
x = range(2,26,2) # x轴的位置
y = [random.randint(15, 30) for i in x]
plt.figure(figsize=(20,8),dpi=80)

# 设置x轴和y轴刻度的API
# plt.xticks(x)
# plt.xticks(range(1,25)) # 设置y轴的刻度
# plt.yticks(y)
# plt.yticks(range(min(y),max(y)+1))

# 构造x轴刻度标签
x_ticks_label = ["{}:00".format(i) for i in x]
rotation = 45 # 让字旋转45度
plt.xticks(x,x_ticks_label,rotation = 45)
# 设置y轴的刻度标签
y_ticks_label = ["{}℃".format(i) for i in range(min(y),max(y)+1)]
plt.yticks(range(min(y),max(y)+1),y_ticks_label)

# 绘 图
plt.plot(x,y)
plt.show()
```

设置显示中文

默认情况下，matplotlib只显示英文,无法显示中文。

如果需要显式中文的话，则需要引入字体管理器。

C:\Windows\\Fonts\\STSONG.TTF 注意如果无法显示汉字，说明字体路径本机没有，更换一下

```
from matplotlib import pyplot as plt
from matplotlib import font_manager

x = range(2,26,2) # x轴的位置
y = [random.randint(15, 30) for i in x]
plt.figure(figsize=(20,8),dpi=80)

# 构造x轴刻度标签
x_ticks_label = ["{}:00".format(i) for i in x]
rotation = 45 # 让字旋转45度
plt.xticks(x,x_ticks_label,rotation = 45)
# 设置y轴的刻度标签
y_ticks_label = ["{}℃".format(i) for i in range(min(y),max(y)+1)]
plt.yticks(range(min(y),max(y)+1),y_ticks_label,rotation = 0)

# 设置坐标轴标签与字体样式
my_font =
font_manager.FontProperties(fname='c:\\windows\\fonts\\msyh.ttc',size=
18)

# 设置标题
plt.title("上海天气",fontproperties=my_font)

# 设置坐标轴标签
plt.xlabel("时间",fontproperties=my_font)
plt.ylabel("温度",fontproperties=my_font)

# 绘 图
plt.plot(x,y)
plt.show()
```

一图多线

需求：

假设大家在30岁的时候，根据自己的实际情况，统计出来你和你同事各自从11岁到30岁每年交的新朋友的数量，如列表y1和y2，请在一个图中绘制出该数据的折线图，从而分析每年交朋友的数量走势。

y1 = [1, 0, 1, 1, 2, 4, 3, 4, 4, 5, 6, 5, 4, 3, 3, 1, 1, 1, 1, 1] y2 = [1, 0, 3, 1, 2, 2, 3, 4, 3, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1]

```
from matplotlib import pyplot as plt
```

```
from matplotlib import font_manager

y1 = [1, 0, 1, 1, 2, 4, 3, 4, 4, 5, 6, 5, 4, 3, 3, 1, 1, 1, 1, 1]
y2 = [1, 0, 3, 1, 2, 2, 3, 4, 3, 2, 1, 2, 1, 1, 1, 1, 1, 1, 1, 1]
x = range(11,31)
plt.figure( figsize=(20, 8), dpi=80 )

plt.plot(x,y1,color='red',alpha=1,linestyle='-',linewidth=2, label='小王')
plt.plot(x,y2,color='blue',alpha=1,linestyle='-',linewidth=2, label='小张')
# 引入字体
my_font = font_manager.FontProperties(fname='c:\\windows\\fonts\\msyh.ttc',size=16)
#
# 设置刻度
xtick_labels = ['{}岁'.format( i ) for i in x]
plt.xticks( x, xtick_labels, fontproperties=my_font, rotation=45 )
# 添加图例
plt.legend(prop=my_font)

plt.show()
```

一个画布多个子图

这个了解一下，后面学了Numpy之后，结合起来一起用。

```
import matplotlib.pyplot as plt
# 引入numpy, 需要先安装numpy的包
import numpy as np

# 设置x的值为1到99
x  =  np.arange(1,  100)

# 创建父图，设置画布
fig=plt.figure(figsize=(20,10),dpi=80)

fig,axes=plt.subplots(2,2)
ax1=axes[0,0]
ax2=axes[0,1]
ax3=axes[1,0]
ax4=axes[1,1]

# 子图1
ax1.plot(x, x)
```

```
# 子图2
ax2.plot(x, -x)

# 子图3
ax3.plot(x, x**2)

# 子图4
ax4.plot(x, np.log(x))

plt.show()
```

```
# 第二种方式
# 使用 plt.subplot() 添加子图

import matplotlib.pyplot as plt
import numpy as np

x  =  np.arange(1,  100)

# 创建子图
# 2,2,1 ==> 两行两列，放在第一个位置
plt.subplot(2,2,1)
plt.plot(x,x)

plt.subplot(2,2,2)
plt.plot(x, -x)

plt.subplot(2,2,3)
plt.plot(x, x ** 2)

plt.subplot(2,2,4)
plt.plot(x,np.log(x))

plt.show()
```

散点图

使用`plt.scatter()`绘制散点图

```
'''
题干:3月份每天最高气温
a =
[11,17,16,11,12,11,12,6,6,7,8,9,12,15,14,17,18,21,16,17,20,14,15,15,15
,19,21,22,22,22,23]
```

```
'''

from matplotlib import pyplot as plt
from matplotlib import font_manager

y =
[11,17,16,11,12,11,12,6,6,7,8,9,12,15,14,17,18,21,16,17,20,14,15,15,15
,19,21,22,22,22,23]
x = range( 1, 32 )

# 设置图形大小
plt.figure( figsize=(20, 8), dpi=80 )

# 使用scatter绘制散点图
plt.scatter( x, y, label='3月份' )

# 调整x轴的刻度
my_font =
font_manager.FontProperties(fname='c:\\windows\\fonts\\msyh.ttc',size=
16)

xticks_labels = ['3月{}日'.format( i ) for i in x]
plt.xticks( x[::3], _xticks_labels[::3], fontproperties=my_font,
rotation=45 )

# 设置坐标轴标签
plt.xlabel( ' 日 期 ', fontproperties=my_font )
plt.ylabel( '温度', fontproperties=my_font )
# 设置图例
plt.legend( prop=my_font )

plt.show()
```

条形图

使用`plt.bar()`绘制条形图

示例： 假设你获取到了2019年内地电影票房前20的电影（列表a)和电影票房数据（列表b)，请展示该数据：

a = ['流浪地球','疯狂的外星人','飞驰人生','大黄蜂','熊出没·原始时代','新喜剧之王'] b = [38.13,19.85,14.89,11.36,6.47,5.93]

```
import  matplotlib.pyplot as plt
from matplotlib import font_manager
```

```
a = ['流浪地球','疯狂的外星人','飞驰人生','大黄蜂','熊出没·原始时代','新喜剧之王']
b = [38.13,19.85,14.89,11.36,6.47,5.93]

# 引入字体
my_font = font_manager.FontProperties(fname='c:\\windows\\fonts\\msyh.ttc',size=16)

# 绘制画布
plt.figure(figsize=(20, 8), dpi=80)

# 绘制条形图
rects = plt.bar(range(len(b)),b,width=0.3,color='r')

# 绘制x轴标签
plt.xticks(range(len(a)),a,fontproperties=my_font,rotation=45)

# 给条形图添加标签
for rect in rects:
    height = rect.get_height()
    plt.text(rect.get_x() + rect.get_width() / 2, height + 0.05, '%.2f' % height, ha='center')

plt.show()
```

直方图

现有250部电影的时长，希望统计出这些电影时长的分布状态(比如时长为100分钟到120分钟电影的数量，出现的频率)等信息，你应该如何呈现这些数据?

采用直方图，使用 `plt.hist()` 来绘制直方图

```
time = [131, 98, 125, 131, 124, 139, 131, 117, 128, 108, 135, 138,
131, 102, 107, 114, 119, 128, 121, 142, 127, 130, 124, 101, 110, 116,
117, 110, 128, 128, 115, 99, 136, 126, 134, 95, 138, 117, 111, 78,
132, 124, 113, 150, 110, 117, 86, 95, 144, 105, 126, 130, 126, 130,
126, 116, 123, 106, 112, 138, 123, 86, 101, 99, 136, 123, 117, 119,
105, 137, 123, 128, 125, 104, 109, 134, 125, 127, 105, 120, 107, 129,
116, 108, 132, 103, 136, 118, 102, 120, 114, 105, 115, 132, 145, 119,
121, 112, 139, 125, 138, 109, 132, 134, 156, 106, 117, 127, 144, 139,
139, 119, 140, 83, 110, 102, 123, 107, 143, 115, 136, 118, 139, 123,
112, 118, 125, 109, 119, 133, 112, 114, 122, 109, 106, 123, 116, 131,
127, 115, 118, 112, 135, 115, 146, 137, 116, 103, 144, 83, 123, 111,
110, 111, 100, 154, 136, 100, 118, 119, 133, 134, 106, 129, 126, 110,
111, 109, 141, 120, 117, 106, 149, 122, 122, 110, 118, 127, 121, 114,
125, 126, 114, 140, 103, 130, 141, 117, 106, 114, 121, 114, 133, 137,
92, 121, 112, 146, 97, 137, 105, 98, 117, 112, 81, 97, 139, 113, 134,
106, 144, 110, 137, 137, 111, 104, 117, 100, 111, 101, 110, 105, 129,
137, 112, 120, 113, 133, 112, 83, 94, 146, 133, 101, 131, 116, 111,
84, 137, 115, 122, 106, 144, 109, 123, 116, 111, 111, 133, 150]

import matplotlib.pyplot as plt

# 画布
plt.figure(figsize=(20, 8), dpi=80)

# 绘制直方图
# bins=20 表示将数据分为20个区间，也就是20组，当然，这个数据也可以计算出来
plt.hist(time,bins=20,color='green')

# 添加x轴标签
plt.xlabel('播放时长',fontproperties=my_font)
# 添加y轴标签
plt.ylabel('播放次数',fontproperties=my_font)

# 修改x轴刻度显示
plt.xticks(range(min(time),max(time) + 1,  5), [str(i) + '分钟' for i
in range(min(time), max(time) + 1, 5)], fontproperties=my_font)

plt.show()
```

饼图

使用`plt.pie()`绘制饼图

```
"""
参数解释：
    explode:  设置各部分突出多少
```

```
    label：设置各部分标签
    labeldistance:设置标签文本距圆心位置，1.1表示1.1倍半径
    autopct：设置圆内文本
    shadow：设置是否有阴影
    startangle：起始角度，默认从0开始逆时针转
    pctdistance：设置圆内文本距圆心距离返回值
"""

import matplotlib.pyplot as plt
import matplotlib
label_list = ["第一部分", "第二部分", "第三部分"]    # 各部分标签
size = [55, 35, 10]                                 # 各部分大小
color = ["red", "green", "blue"]                    # 各部分颜色
explode = [0, 0.05, 0]                              # 各部分突出值

plt.pie(size,explode=explode, colors=color,
labels=label_list,labeldistance=1.1,
autopct="%1.1f%%",shadow=False,startangle=90, pctdistance=0.6)

plt.axis("equal")  # 设置横轴和纵轴大小相等，这样饼才是圆的
plt.show()
```