Python 探索性数据分析

金林

jinlin@zuel.edu.cn

中南财经政法大学统计与数学学院

2020-01

简介

1. 在进行任何统计分析之前，都需要对数据进行探索性分析（Exploratory Data Analysis,EDA）以了解资料的性质和数据的特点。
2. 当面对一组陌生的数据时，进行探索性统计分析有助于我们掌握数据的基本情况。
3. 探索性数据分析是通过分析数据集以决定选择哪种方法适合统计推断的过程。
4. 对于一维数据，它们是否近似地服从正态分布？是否呈现拖尾或截尾分布？其分布是对称的，还是呈偏态的？分布是单峰、双峰、还是多峰的？
5. 实现这一分析的主要过程是计算基本统计量和绘制基本统计图。

基本描述统计量

- python 提供了很多对数据进行基本分析的函数，
- 下表所列是 python 对变量（序列或数据框）进行基本数据分析的函数，其中描述统计量 describle 可对数据做一些基本描述，默认为计算定量数据的基本统计量。

定性数据	用途	定量数据	用途
value_counts()	一维频数表	mean()	均值
crosstab()	二维列联表	median()	中位数
pivot_table()	多维透视表	quantile()	分位数
		std()	标准差

代码

```
import numpy as np
import pandas as pd
BSdata = pd.read_csv('../data/BSdata.csv')
BSdata.describe()
```

```
BSdata[['性别','开设','课程','软件']].describe()
```

定性数据汇总分析

- 统计学中把取值范围是有限个值或一个数值的变量称为离散变量，其中表示分类情况的数据又称为定性数据。

1. 频数：绝对数 python 中的.value.counts() 函数可对定性数据计算频数。

```
T1=BSdata.性别.value_counts();T1
```

这是分类变量，来源于频数分析，说明在 52 名学生中有男生 27 人、女生 25 人。

2. 频率：相对数频数/总数为定性数据的频率。

```
T1/sum(T1)*100
```

这是性别的频率分析，说明在 52 名学生中男生占 51.92%，女生占 48.08%。

定量数据汇总分析

- 对于数值型数据，经常要分析它的集中趋势和离散程度，用来描述集中趋势的量主要有均值、中位数；
- 描述离散程度的量主要有方差、标准差。
- python 只需要一个命令就可以简单地得到这些结果，计算均值、中位数、方差、标准差的命令分别是 mean()、media()、var()、std()。
- 方差、标准差对异常值很敏感，这时我们可以用稳健的极差、四分位间距来描述离散程度。
- python 还提供了函数 quantile()——对数据计算分位数，describe()——求出分位数。

定量数据汇总分析

1. 均数（算术平均数）指一组数据的除以这组数据的个数所得到的商，它反映一组数据的总体水平。对于正态分布数据，通常计算其均值，来表示其集中趋势或平均水平。

```
BSdata.身高.mean()
```

2. 中位数：指一组数据按大小顺序排列，处于中间位置的一个数据（或中间两个数据的平均值），它反映了一组数据的集中趋势。对非正态分布数据，通常计算其中位数，来表示其平均水平。

```
BSdata.身高.median()
```

3. 极差：指一组数据中最大数据与最小数据的差，在统计中常用极差来刻画一组数据的离散程度。它反映的是变量分布的变异范围和离散程度，在总体中任何两个单位的数值之差都不能超过极差。

```
BSdata.身高.max()-BSdata.身高.min()
```

定量数据汇总分析

4. 方差：指各数据与平均数之差的平方的平均数，它表示数据的离散程度和数据的波动大小。

```
BSdata.身高.var()
```

5. 标准差：方差的算术平方根。作用等同于方差，但单位与原数据单位是一致的。对正态分布数据，通常计算其标准差，来反映其变异水平。

```
BSdata.身高.std()
```

方差或标准差是表示一组数据的波动性的指标，因此，方差或标准差可以判断一组数据的稳定性——方差或标准差越大，数据越不稳定；方差或标准差越小，数据越稳定。

定量数据汇总分析

6. 四分位间距（IQR）：python 提供了函数 quantile()，对定量数据计算分位数，IQR=quantile（x,0.75）-quantile(x,0.25)

```
BSdata.身高.quantile(0.75)-BSdata.身高.quantile(0.25)
```

7. 偏度：skew()

```
BSdata.身高.skew()
```

8. 峰度：kurt()

```
BSdata.身高.kurt()
```

自编计算基本统计量函数

- 要发挥 python 的优势，通常可构建一些数据分析函数来进行基本的数据分析。

```
def stats(x):
    stat=[x.count(),x.min(),x.quantile(.25),x.mean(),x.median(),
          x.quantile(.75),x.max(),x.max()-x.min(),x.var(),x.std(),x.skew
    (),x.kurt()]
    stat=pd.Series(stat,index=['Count','Min', 'Q1(25%)','Mean','Median'
    ,
                   'Q3(75%)','Max','Range','Var','Std','Skew','Kurt'])
    return(stat)

stats(BSdata.身高)
```

```
stats(BSdata.支出)
```

- 这些函数还可以不断完善，例如它只能计算向量或变量数据，无法计算矩阵或数据框的数据，大家可自行编写一个计算矩阵或数据框的基本统计量函数。

- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

常用的绘图函数

1. matplotlib 是 python 的基本绘图包，是一个图形框架。
2. 它是 python 最著名的绘图库，提供了一整套和 Matlab 相似的命令 API，十分适合交互式地进行制图。
3. 在绘制中文图形时，须作一些基本设置。

```
import matplotlib.pyplot as plt                    #基本绘图包
plt.rcParams['font.sans-serif']=['KaiTi'];    #SimHei黑体
plt.rcParams['axes.unicode_minus']=False;     #正常显示图中负号
# plt.figure(figsize=(6,5));                  #图形大小
```

常见统计作图函数

- 常用的统计图函数如下表所列

定性数据	用途	定量数据	用途
bar()	条图	plot()	折线图
pie()	饼图	hist()	直方图

- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

条图 (bar)

bar() 在对分类数据作条形图时，须先对原始数据分组，否则作出的不是分类数据的条形图。

```
X=['A','B','C','D','E','F','G']
Y=[1,4,7,3,2,5,6]
plt.bar(X,Y) # 条图
```

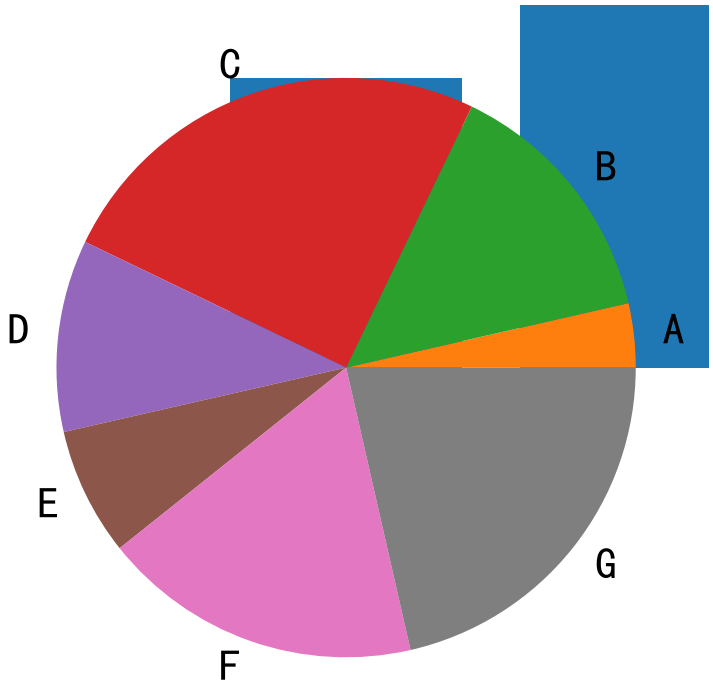
```
plt.savefig(”abc”, format=”pdf”)
```

饼图（pie）

使用命令 pie()，注意：和条形图一样，对原始数据作饼图前要先分组。

```
1 plt.pie(Y,labels=X)  # 饼图
```

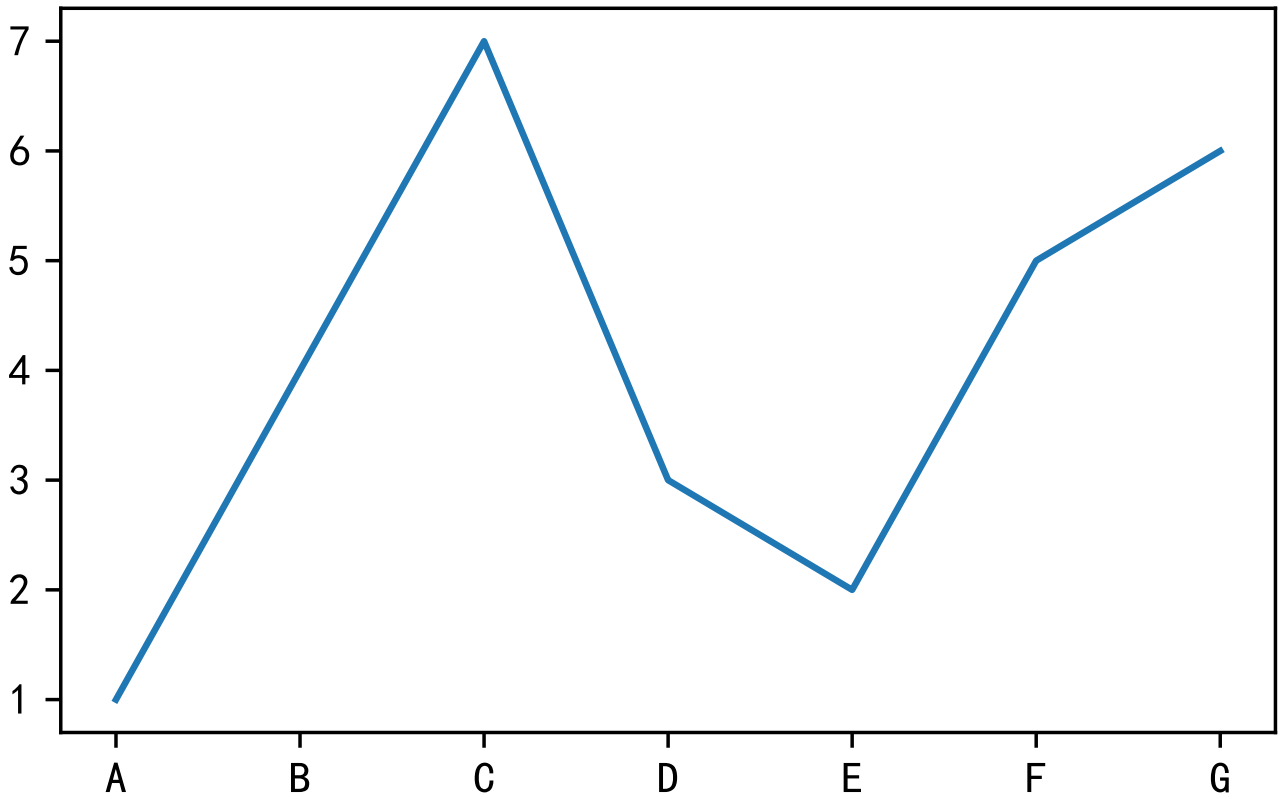
```
1 plt.show()
```



- 常用绘图函数
- 定性数据作图
- **定量数据的基本统计图**
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

线图（plot）

```
1 plt.plot(X,Y)  #线图 plot
```



直方图（hist）

```
plt.hist(BSdata.身高)  # 频数直方图
```

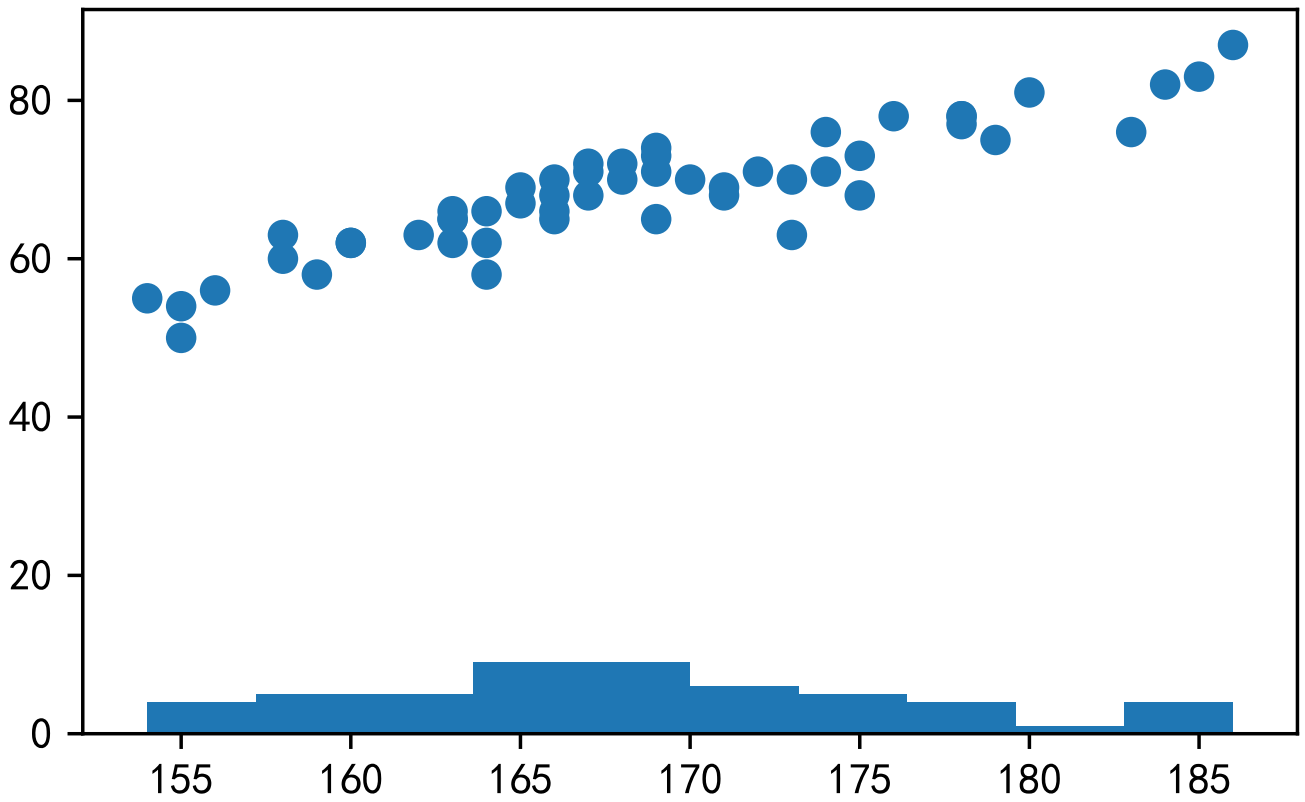
```
plt.hist(BSdata.身高,density=True) # 频率直方图
```


```
plt.hist(BSdata.身高,density=True) # 频率直方图
```


matplotlib 里用来作直方图的函数是 hist() 也可以用频率作直方图，只要把 density 参数设置为 True 就可以了，默认为 False。

散点图 (scatter)

```
1 plt.scatter(BSdata.身高, BSdata.体重);  # 散点图
```



- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

简介

- 我们可以通过设置不同的图形参数对图形进行调整和优化。
- python 中的每个绘图函数，都有许多参数设置选项，大多数函数的部分选项是一样的，下面给出一些主要的共同选项及其默认值。

标题、标签、标尺及颜色

1. 在使用 matplotlib 模块画坐标图时，往往需要对坐标轴设置很多参数，这些参数包括横纵坐标轴范围、坐标轴刻度大小、坐标轴名称等。
2. 在 matplotlib 中包含了很多函数，用来对这些参数进行设置。
   1. plt.xlim、plt.ylim：设置横、纵坐标轴范围；
   2. plt.xlabel、plt.ylabel：设置坐标轴名称；
   3. plt.xticks、plt.yticks：设置坐标轴刻度；
   4. colors：控制图形的颜色，'red'：设置为红色。

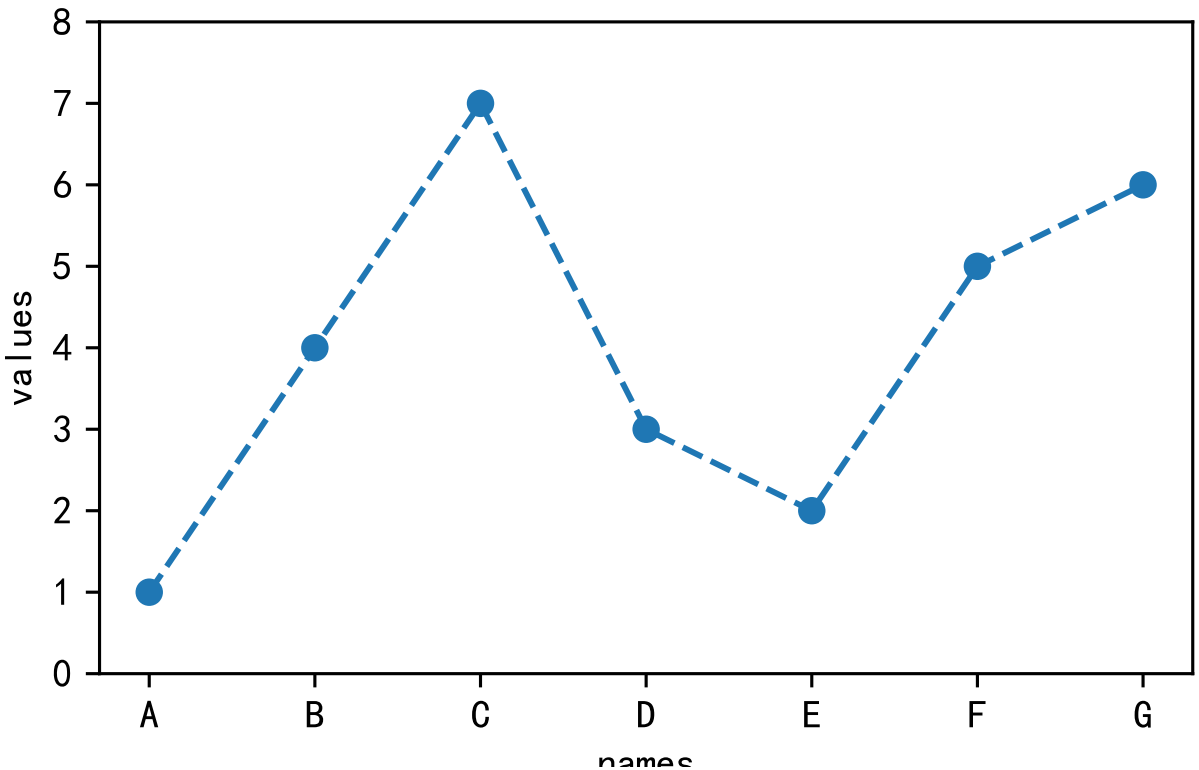
```
plt.ylim(0,8);
```

```
plt.xlabel('names');plt.ylabel('values');
plt.xticks(range(len(X)), X)
```

线型和符号

1. linestyle：控制连线的线型（-：实线，–：虚线，.：点线）；
2. marker：控制符号的类型，例如，‘o’绘制实心圆点图。

```
1 plt.plot(X,Y,linestyle='--',marker='o')
```

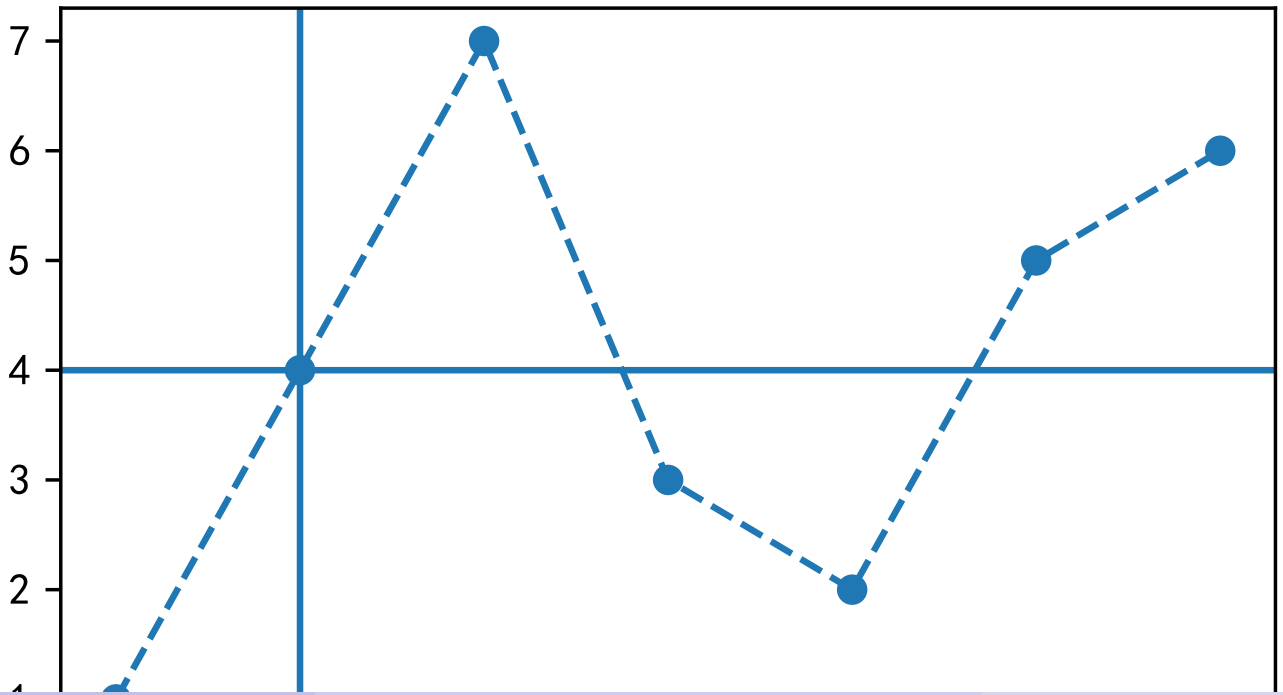


- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

简介

1. 使用高级绘图函数可以画出一幅新图，而低级绘图函数只能作用于已有的图形之上。
2. 垂线：在纵坐标 y 处画垂直线（plt.axvline）；
3. 水平线：在横坐标 x 处画水平线（plt.axhline）。

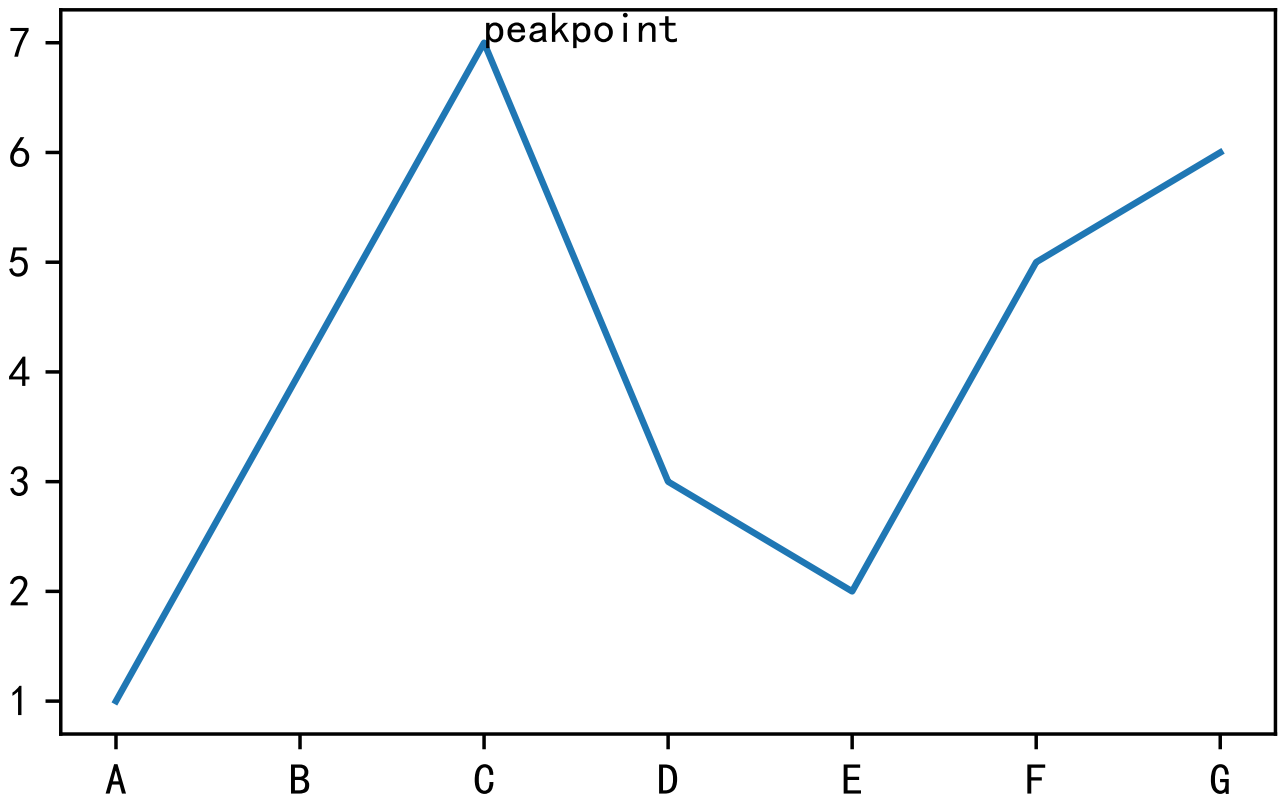
```
1 plt.plot(X,Y,'o--')
2 plt.axvline(x=1)
3 plt.axhline(y=4)
```



添加文字

text (x,y,labels,...) , 在 (x,y) 处添加用 labels 指定的文字。

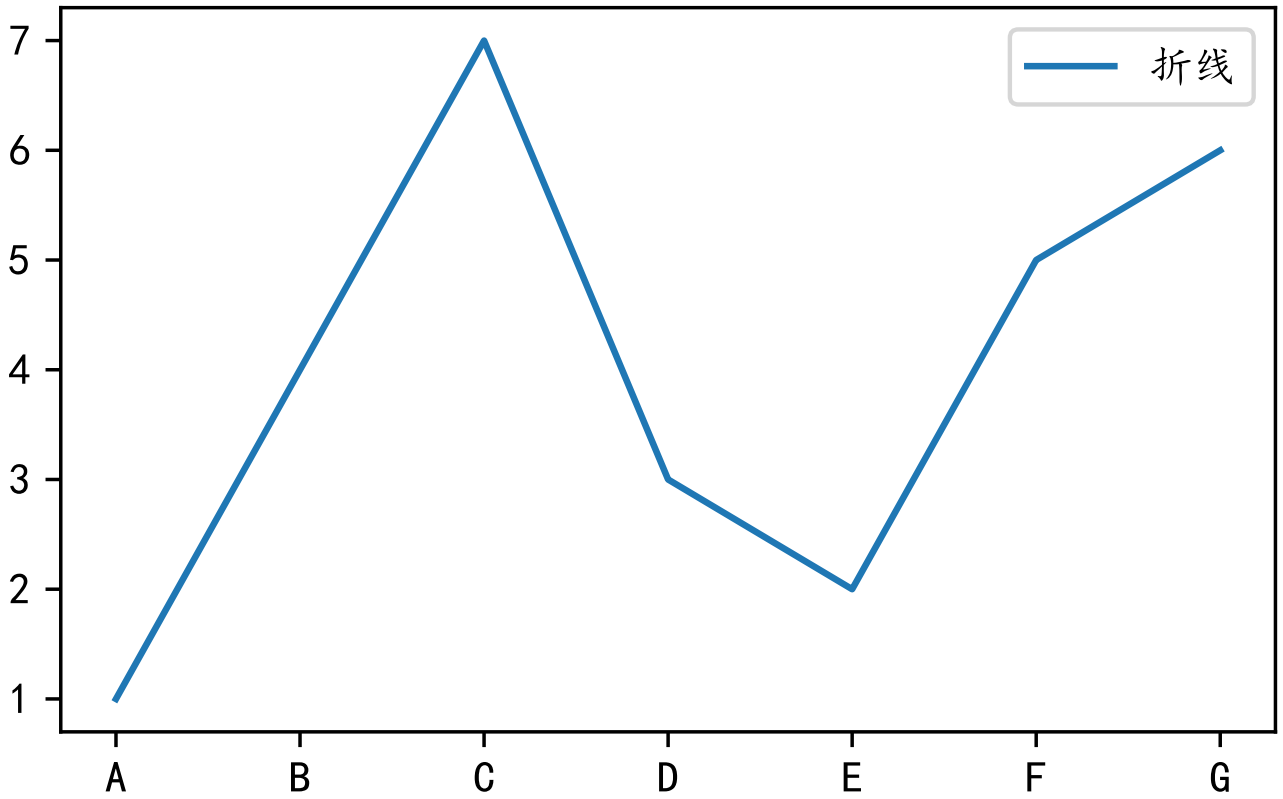
```
1 plt.plot(X,Y);
2 plt.text(2,7,'peakpoint')
```



图例

绘制图形后，可使用 legend 函数给图形加图例。

```
1 plt.plot(X,Y,label=u'折线')
2 plt.legend()
```



误差条图

```
s=[0.1,0.4,0.7,0.3,0.2,0.5,0.6]
plt.bar(X,Y,yerr=s,error_kw={'capsize':5})
```

- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

多图

1. 在 matplotlib 中，一个 figure 对象可以包含多个子图（Axes），可以使用 subplot 快速绘制，其调用形式为

   subplot (numRows, numCols, plotNum)

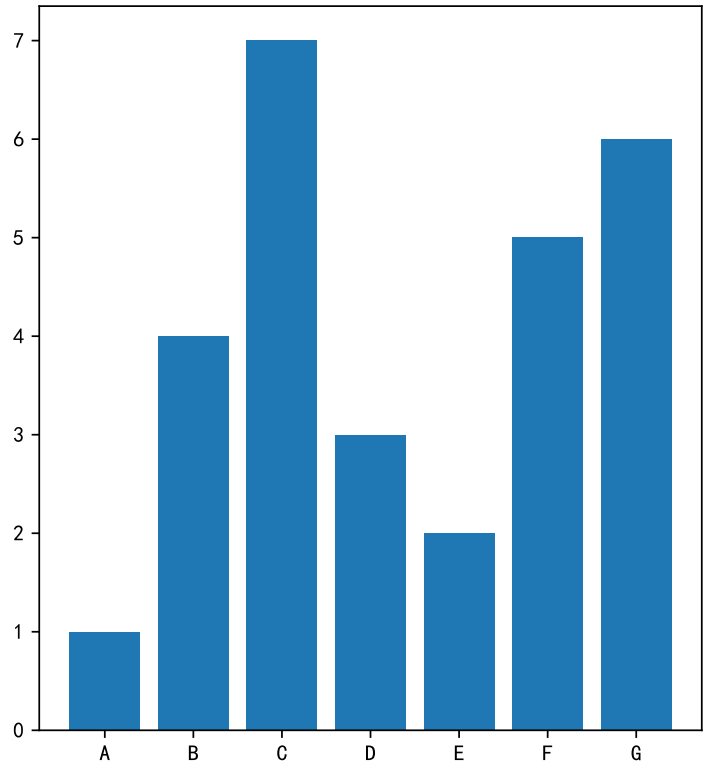
2. 图表的整个绘图区域被分成 numRows 行和 numCols 列。
3. 然后按照从左到右、从上到下的顺序对每个子区域进行编号，左上子区域的编号为 1，plotNum 参数指定创建的 Axes 对象所在的区域。

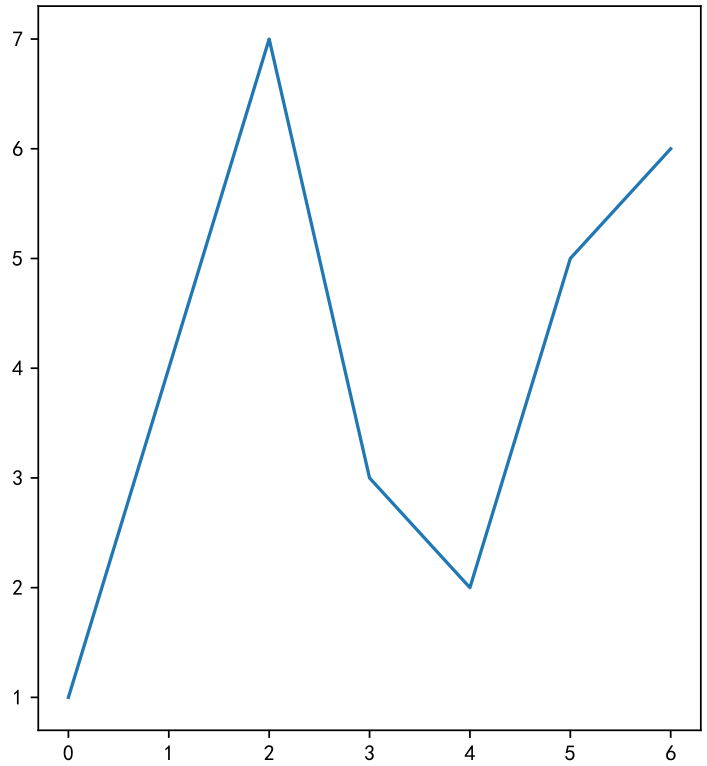
多图

```
'''一行绘制两个图形'''
```

```
plt.figure(figsize=(12,6));
plt.subplot(1,2,1); plt.bar(X,Y);
```

```
plt.subplot(1,2,2); plt.plot(Y)
```



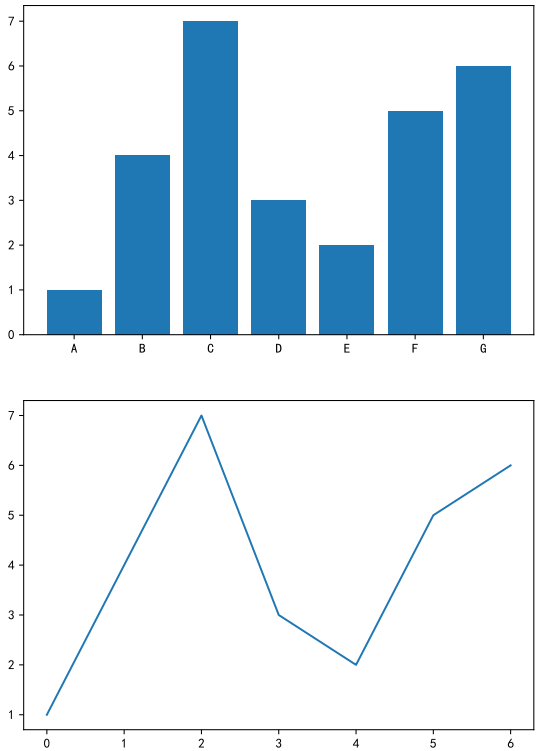


多图

```
'''一列绘制两个图形'''
```

```
plt.figure(figsize=(7,10));
plt.subplot(2,1,1); plt.bar(X,Y);
```

```
plt.subplot(2,1,2); plt.plot(Y)
```

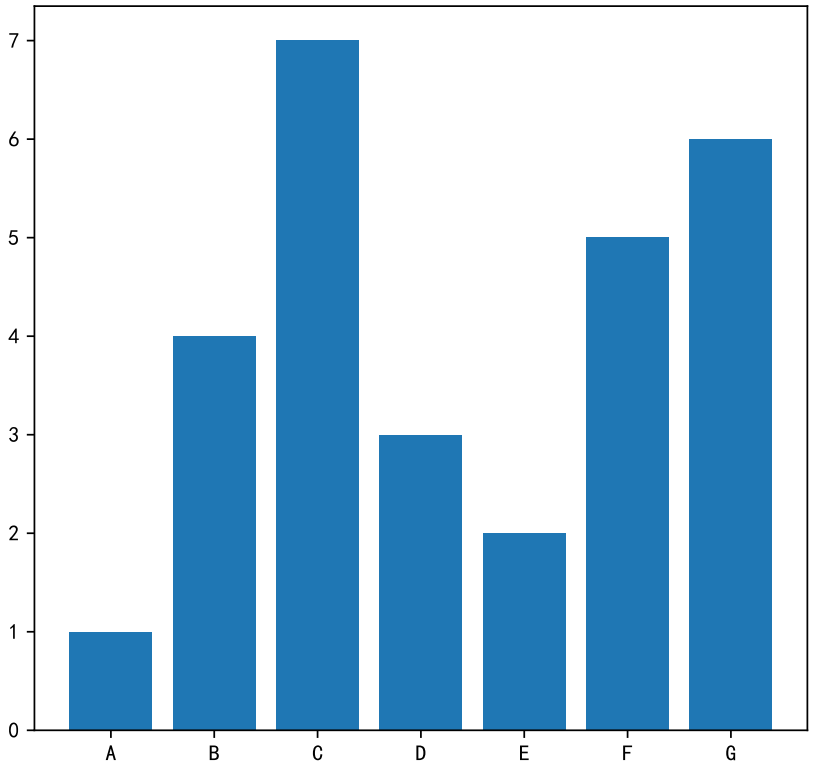


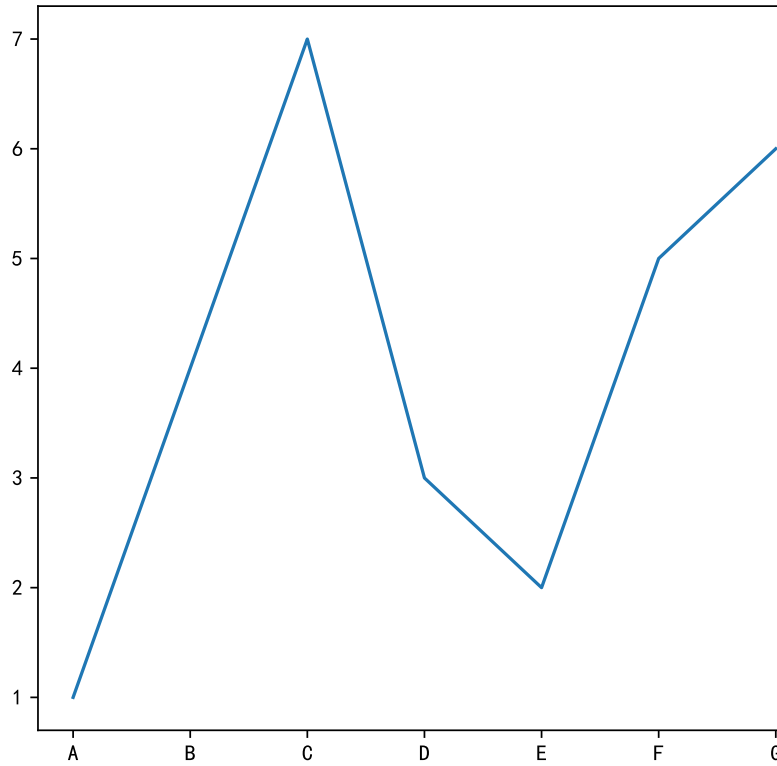
多图

```
'''一页绘制两个图形'''
```

```
fig,ax = plt.subplots(1,2,figsize=(14,6))
ax[0].bar(X,Y)
```

```
ax[1].plot(X,Y)
```



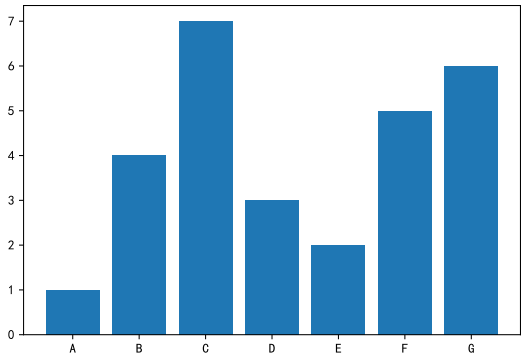


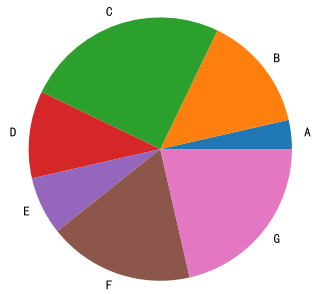
多图

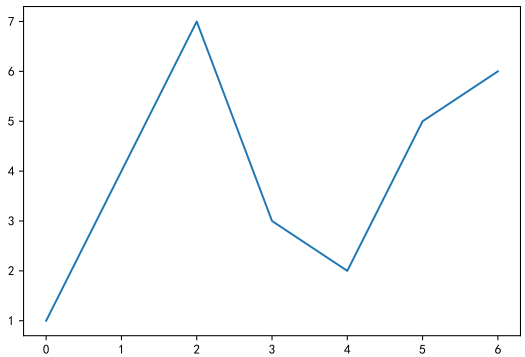
```
'''一页绘制四个图形'''
```

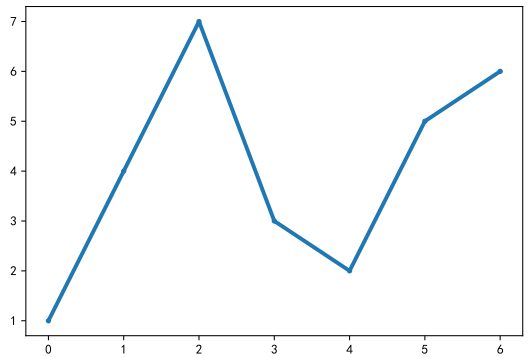
```
fig,ax=plt.subplots(2,2,figsize=(15,10))
ax[0,0].bar(X,Y); ax[0,1].pie(Y,labels=X)
```

```
ax[1,0].plot(Y); ax[1,1].plot(Y,'.-',linewidth=3)
```









- 常用绘图函数
- 定性数据作图
- 定量数据的基本统计图
- 图形参数设置
- 低级绘图命令
- 多图
- 基于 pandas 的绘图

简介

1. 在 pandas 中，数据框有行标签、列标签及分组信息等，即要制作一张完整的图表，原本需要很多行 matplotlib 代码，现在只需一两条简洁的语句即可。
2. pandas 有许多能够利用 DataFrame 对象数据组织特点来创建标准图标的高级绘图方法。
3. 对于数据框 DataFrame 绘图，其每列都为一个绘图图线，会将每列作为一个图线绘制到一张图片当中，并用不同的线条颜色及不同的图例标签来表示。

基本格式

```
DataFrame.plot(kind='line')
kind:#图类型
'line':(default)#折线图
'bar':(default)#垂直条图
'barh':(default)#水平条图
'hist':(default)#直方图
'box':(default)#箱线图
'kde':(default)#核密度估计图，对柱状图添加概率密度线，可'density'
'area':(default)#面积图
'pie':(default)#饼图
'scatter':(default)#散点图
```

定量数据

```
BSdata['体重'].plot(kind='line');
BSdata['体重'].plot(kind='hist');
```

```
BSdata['体重'].plot(kind='box');
```

```
BSdata['体重'].plot(kind='density',title='Density')
```

```
BSdata[['身高','体重','支出']].plot(subplots=True,layout=(1,3),kind='
    box')
```

```
BSdata[['身高','体重','支出']].plot(subplots=True,layout=(1,3),kind='
    density')
```

```
BSdata[['身高','体重','支出']].plot(subplots=True,layout=(3,1),kind='
    density')
```

定性数据

```
T1=BSdata['开设'].value_counts();T1
```

```
pd.DataFrame({'频数':T1,'频率':T1/T1.sum()*100})
```

```
T1.plot(kind='bar'); #T1.sort_values().plot(kind='bar');
T1.plot(kind='pie')
```

- 一维频数分析
- 二维集聚分析
- 多维透视分析

定性数据频数分布

1. pivot-table

```
BSdata['开设'].value_counts()
#BSdata.pivot_table(values='学号',index='开设',aggfunc=len)
```

定性数据频数分布

② 自定义计数汇总函数

由于 python 自带的 value_counts() 函数只能统计定性数据的个数，无法计算其频率，于是我们自定义一个函数 tab() 来进行统计和绘图。

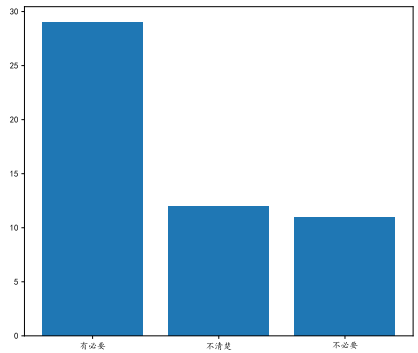
```
def tab(x,plot=False): #计数频数表
    f=x.value_counts();f
    s=sum(f);
    p=round(f/s*100,3);p
    T1=pd.concat([f,p],axis=1);
    T1.columns=['例数','构成比'];
    T2=pd.DataFrame({'例数':s,'构成比':100.00},index=['合计'])
    Tab=T1.append(T2)
    if plot:
      fig,ax = plt.subplots(2,1,figsize=(8,15))
      ax[0].bar(f.index,f); # 条图
      ax[1].pie(p,labels=p.index,autopct='%1.2f%%');  # 饼图
    return(round(Tab,3))


tab(BSdata.开设,True)
```

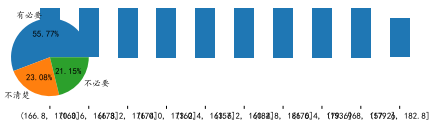
定量数据频数分布

1 身高频数表与条图

```
pd.cut(BSdata.身高,bins=10).value_counts()
```

```
pd.cut(BSdata.身高,bins=10).value_counts().plot(kind='bar')
```



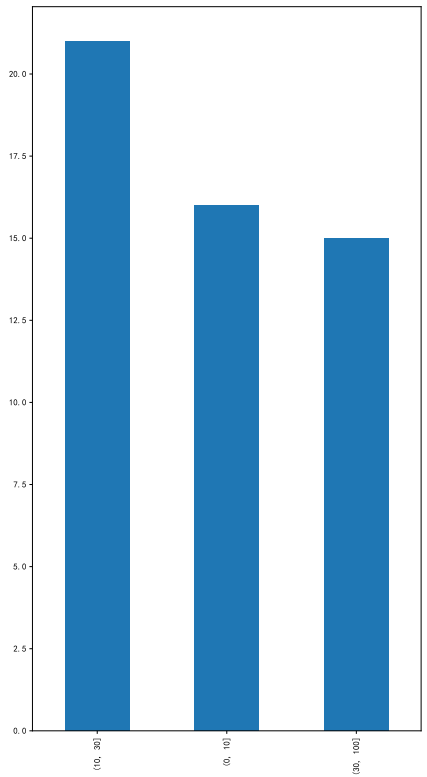


定量数据频数分布

2 支出频数表

```
1 pd.cut(BSdata.支出,bins=[0,10,30,100]).value_counts()
```

```
1 pd.cut(BSdata.支出,bins=[0,10,30,100]).value_counts().plot(kind='bar')
```



定量数据频数分布

1. 自定义计量频率分析函数

由于 python 自带的 hist 函数不是以频数表的形式显示的，于是自定义一个函数 freq 来进行统计和绘图。

```
def freq(X,bins=10): #计量频数表与直方图
    H=plt.hist(X,bins);
    a=H[1][:-1];a
    b=H[1][1:];b
    f=H[0];f
    p=f/sum(f)*100;p
    cp=np.cumsum(p);cp
    Freq=pd.DataFrame([a,b,f,p,cp])
    Freq.index=['[下限','上限)','频数','频率(%)','累计频数(%)']
    return(round(Freq.T,2))

freq(BSdata.体重)
```

- 一维频数分析
- 二维集聚分析
- 多维透视分析

定性数据的列联表

1. 二维列联表 python 的 crosstab() 函数可以把双变量分类数据整理成二维表形式。

```
pd.crosstab(BSdata.开设,BSdata.课程)
```

2. 行和列的合计可使用参数 margins=True。

```
pd.crosstab(BSdata.开设,BSdata.课程,margins=True)
```

定性数据的列联表

1. 对于二维表，我们经常要计算某个数据占行、列的比例或占总和的比例，也就是边缘概率。
2. python 可以很简单地计算这些比例，使用 normalize 参数，normalize=‘index’ 表示各数据占行的比例；normalize=‘columns’ 表示各数据占列的比例；normalize=‘all’，表示各数据占总和的构成比例。例如：

```
pd.crosstab(BSdata.开设,BSdata.课程,margins=True,normalize='index')
```

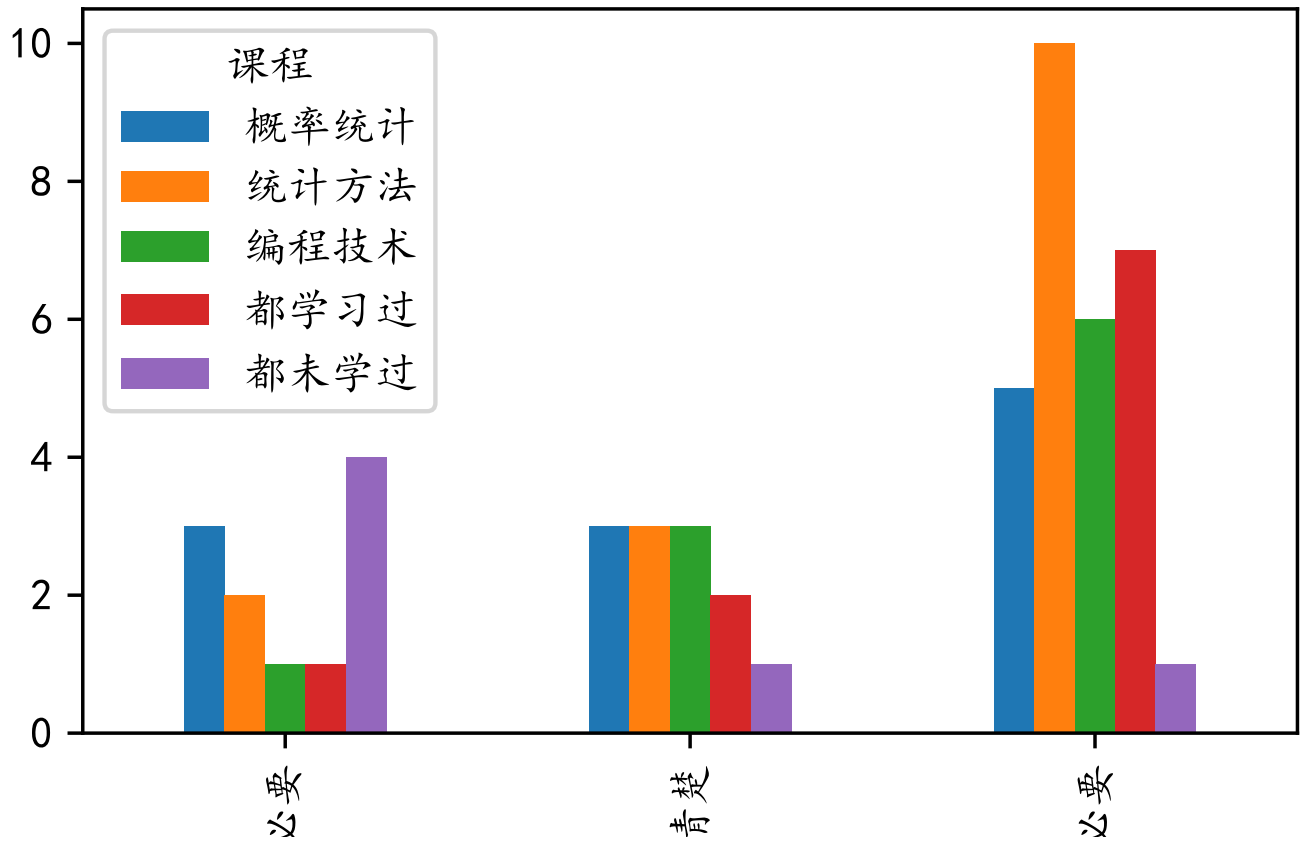
```
pd.crosstab(BSdata.开设,BSdata.课程,margins=True,normalize='columns')
```

```
pd.crosstab(BSdata.开设,BSdata.课程,margins=True,normalize='all').round
    (3)
```

复式条图

```
T2=pd.crosstab(BSdata.开设,BSdata.课程);T2
```

```
T2.plot(kind='bar');
```



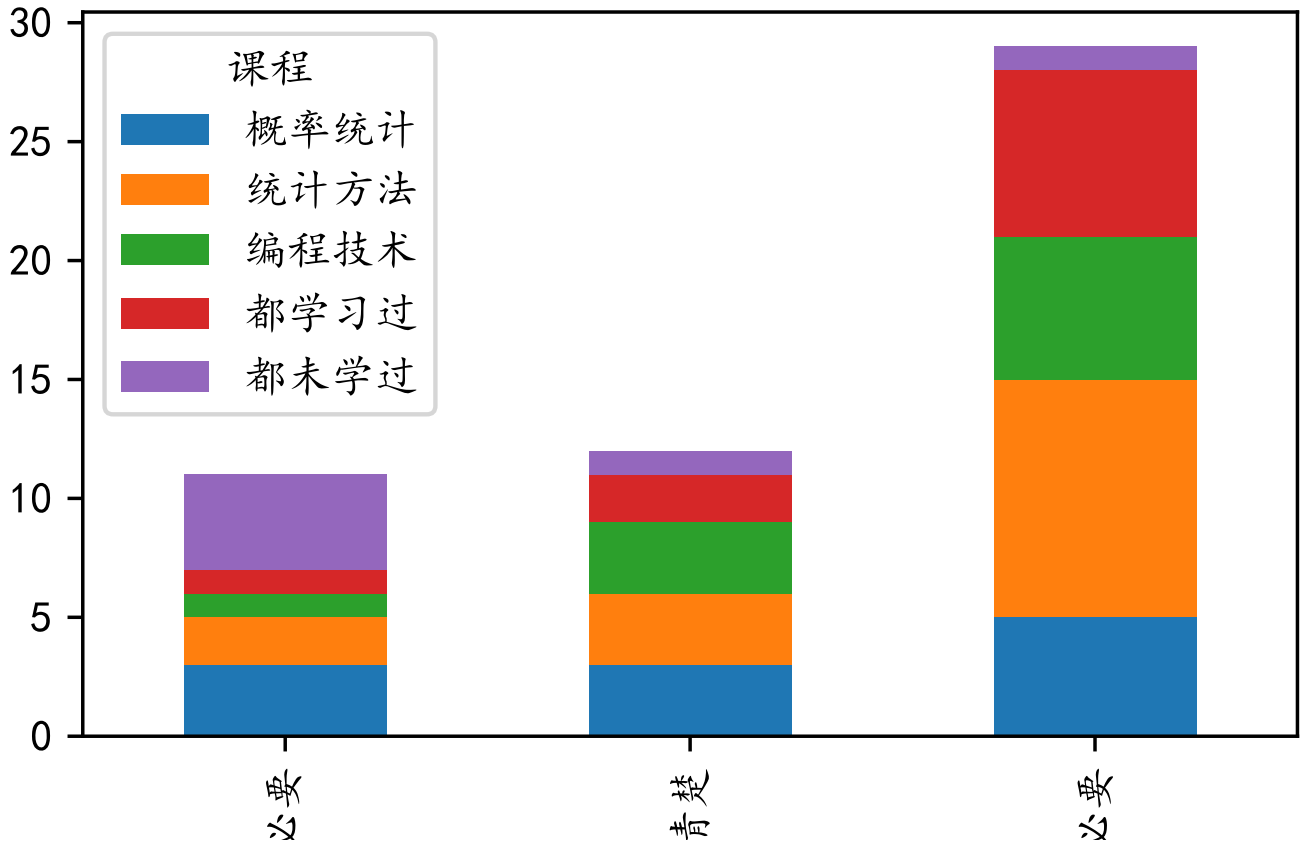
复式条图

1. 条图用等宽直条的长短来表示相互独立的各指标数值大小，该指标可以是连续性变量的某汇总指标，也可以是分类变量的频数或构成比。
2. 各组直条间的间距应相等，其宽度一般与直条的宽度相等或为直条宽度的一半。
3. python 作条形图的函数是 bar()，不过在作条形图前须对数据进行分组。
4. 我们继续以前面的分类数据为例作条形图，粗略分析变量的分布情况。

复式条图

5. stacked 参数设置为 False 时，作出的是分段式条形图；为 True 时，作出的是并列式条形图，默认为 False。

```
1 T2.plot(kind='bar',stacked=True);
```



定量数据的集聚表

1. pandas 提供灵活高效的 groupby 功能，使得用户能以一种自然的方式对数据集进行切片、切块、摘要等操作；
2. 根据一个或多个键（可以是函数、数组或 DataFrame 列名）拆分 pandas 对象；
3. 计算分组摘要统计，如计数、平均值、标准差，以及用户自定义函数；
4. 对 DataFrame 的列应用各种各样的函数。

按列分组

1. 注意：以下使用 groupby() 函数生成的是一个中间分组变量，为 GroupBy 类型.

```
BSdata.groupby(['性别'])
```

```
type(BSdata.groupby(['性别']))
```

按分组统计

在分组结果的基础上应用 size()/sum()/count() 等统计函数，可分别统计分组数量、不同列的分组和、不同列的分组数量。

```
BSdata.groupby(['性别'])['身高'].mean()
```

```
BSdata.groupby(['性别'])['身高'].size()
```

```
BSdata.groupby(['性别','开设'])['身高'].mean()
```

应用 agg() 函数

1. 对于分组的某一列或多列，应用 agg（func）可以对分组后的数据应用 func 函数，
2. 也可以推广到同时作用于多个列和使用多个函数上。

```
1 BSdata.groupby(['性别'])['身高'].agg([np.mean, np.std])
```

应用 apply() 函数

1. apply() 不同于 agg() 的地方在于：前者应用于 dataFrame 的各个列，后者仅作用于指定的列。

```
BSdata.groupby(['性别'])['身高','体重'].apply(np.mean)
```

```
BSdata.groupby(['性别','开设'])['身高','体重'].apply(np.mean)
```

- 一维频数分析
- 二维集聚分析
- 多维透视分析

定性数据透视分析：pivot-table

1. 对定性数据，前面介绍了 value_counts() 函数生成一维表，用 crosstab() 函数生成二维表，其实 pivot_table() 函数可以生成任意维统计表。
2. 使用 pandas 中 pivot_table 命令的各种列联表，可以实现 Excel 等电子表格的透视表功能，且更为灵活。

```
BSdata.pivot_table(index=['性别'],values=['学号'],aggfunc=len)
```

```
BSdata.pivot_table(values=['学号'],index=['性别','开设'],aggfunc=len)
```

```
BSdata.pivot_table(values=['学号'],index=['开设'],columns=['性别'],
    aggfunc=len)
```

定量数据透视分析

```
BSdata.pivot_table(index=['性别'],values=[”身高”],aggfunc=np.mean)
```

```
BSdata.pivot_table(index=['性别'],values=[”身高”],aggfunc=[np.mean,np.
    std])
```

```
BSdata.pivot_table(index=[”性别”],values=[”身高”,”体重”])
```

复合数据透视分析

```
BSdata.pivot_table('学号', ['性别','开设'], '课程', aggfunc=len, margins
    =True, margins_name='合计')
```

```
BSdata.pivot_table(['身高','体重'],['性别',”开设”],aggfunc=[len,np.mean,
    np.std] )
```