## 2.1.3 基本统计量与公共健康数据的相关性分析

### 2.1.3.1 基本统计量：标准误、中心极限定理、t分布、统计显著性、效应量和置信区间 [1][2]

#### 1）标准误

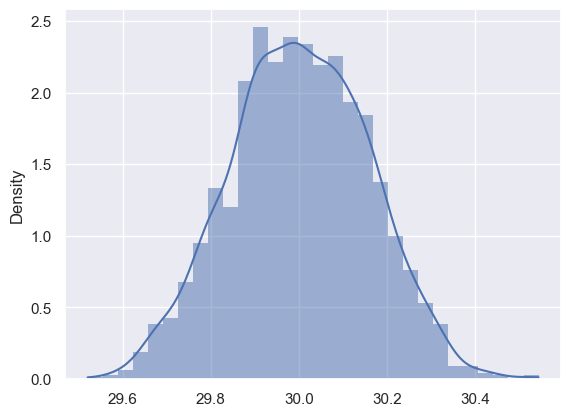
定义：标准误是某一统计量（例如均值、两个均值只差、相关系数等）抽样分布（sampling distribution）的标准差（即样本均值的标准差，而不是样本的标准差），度量了从同一总体中抽取相同容量样本的预期随机差异。在下述代码中，从服从平均值为30，标准差为5的正态分布中，随机提取2000个样本（sample），各个样本容量为1000，并计算每一样本的均值，查看这2000个样本的均值分布，即均值抽样分布（sampling distribution of the mean）。普通分布具有均值和标准差，在均值的抽样分布中，均值则称为均值的期望值（expected value of the mean），这是因为样本均值的最佳猜测与总体均值一致。下述代码中计算了均值抽样分布的均值几乎与总体（population）一样，证明了这一点。均值抽样分布的标准差则称为标准误。标准差是分布中的单个取值与分布均值之间的平均差异或平均离差，均值的标准误提供了同样的信息，只是单个样本均值与其期望值之间的平均差异，可以理解为对样本均值代表实际总体均值的确信程度，有助于确定样本统计值（例如样本均值）与总体参数（例如总体均值）之间的差异是否有意义。同时计算了均值抽样分布标准计分的K-S test正态性检验，每次代码运行产生的数据是符合上述条件下的随机值，其p-value是变化的，但通常高于0.5，即>0.05，因此可以判断均值抽样分布符合正态分布。

均值标准误的计算公式：

​或 ，式中σ为总体标准差， s为标准差的样本估计， n为样本容量。因为通常σ总体的标准差未知，因此用标准差的样本估计，计算标准误。

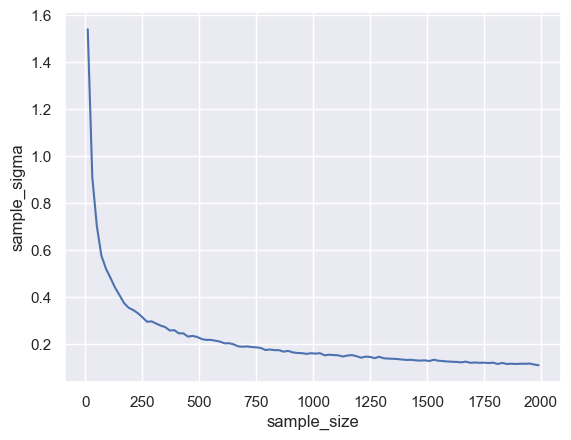
import numpy as np  
from scipy import stats  
import seaborn as sns  
import math  
  
sns.set()  
mu,sigma=30,5  
sample\_size=1000  
sample\_mus=np.array([np.random.normal(mu, sigma, sample\_size).mean() for i in range(2000)]) # 从服从平均值为30，标准差为5的正态分布中，随机提取2000个样本，每个样本容量为1000的样本，并计算每一样本的均值  
bins=30  
sns.histplot(sample\_mus,bins=bins,kde=True,stat="density", linewidth=0) # 查看2000个样本均值分布  
print("sample\_mus mu:%.2f,sigma:%.2f"%(sample\_mus.mean(),sample\_mus.std()))  
kstest\_test=stats.kstest(stats.zscore(sample\_mus),cdf='norm')  
print("sampling distribution of the mean - K-S test statistic:%.2f,p-value:%.2f"%kstest\_test)  
print("standard error of mean:",sample\_mus.std()/math.sqrt(sample\_size) ) # 计算标准误  
print("standard error of mean\_scipy.stats.sem():",stats.sem(sample\_mus,ddof=0)) # 使用scipy.stats库直接计算标准误

sample\_mus mu:30.00,sigma:0.16  
sampling distribution of the mean - K-S test statistic:0.02,p-value:0.41  
standard error of mean: 0.0049183131322453714  
standard error of mean\_scipy.stats.sem(): 0.0034777725678095507



单个样本的容量越大，就越接近于总体，均值抽样分布的均值越趋近于总体均值，而其标准差即标准误越小，表明样本均值的统计量（或计算其它的统计量）能够代表该总体的统计量，对总体的估计也就越准确。利用下述代码分析上述观点，计算多个不用样本容量均值抽样分布的标准误，随着样本容量的增加，标准误迅速的减小后逐步趋于缓和，说明单个样本容量越大，越接近总体，对总体的估计越准确。

import matplotlib.pyplot as plt   
  
sample\_mu\_list=[]  
sample\_sigma\_list=[]  
sample\_size\_list=list(range(10,2000,20))  
for sample\_size in sample\_size\_list:  
 sample\_size\_mu=np.array([np.random.normal(mu, sigma, sample\_size).mean() for i in range(2000)])  
 sample\_mu\_list.append(sample\_size\_mu.mean())  
 sample\_sigma\_list.append(sample\_size\_mu.std())  
ax=sns.lineplot(x=sample\_size\_list,y=sample\_sigma\_list)   
ax.set(xlabel='sample\_size',ylabel='sample\_sigma')  
plt.show()



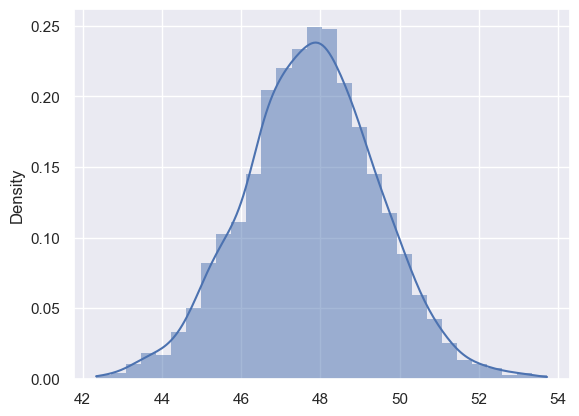
#### 2）中心极限定理

中心极限定义（central limit theorem）表明，只要样本容量足够大，即使样本取值的总体分布不是正态分布，均值的抽样分布也服从正态分布。或理解为不管样本总体服从什么分布，当样本数量足够大时，样本的均值以正态分布的形式围绕总体均值波动。为验证中心极限，使用百度POI数据，提取美食的价格数据。在正态分布与概率密度函数一节，已经通过正态性检验确定了该美食价格数据集为非正态分布，下述代码重现了K-S test正态性检验，p-value=5.153632014592587e-26，小于0.05，拒绝原假设，即不符合正态分布。从该数据集中随机抽取单个样本容量为350，总共2000个样本，计算均值抽样分布后的K-S test正态性检验，其值p-value通常大于0.5，表明均值抽样分布为正态分布，满足中心极限定义。

很多统计量都依赖于从正态分布中得到概率，而中心极限定理确定了足够多的样本均值服从正态分布（即使总体分布不是正态的），即均值抽样分布为正态分布，因此这一定理使得众多统计量的概率计算成为可能。

import util\_misc  
import pandas as pd  
  
poi\_gpd=pd.read\_pickle('./data/poisInAll/poisInAll\_gdf.pkl') # 读取已经存储为.pkl格式的POI数据，其中包括geometry字段，为GeoDataFrame地理信息数据，可以通过poi\_gpd.plot()迅速查看数据。  
delicacy\_price=poi\_gpd.xs('poi\_0\_delicacy',level=0).detail\_info\_price # 提取美食价格数据  
delicacy\_price\_df=delicacy\_price.to\_frame(name='price').astype(float)   
delicacy\_price\_df\_clean=delicacy\_price\_df.dropna()  
\_,delicacyPrice\_outliersDrop=util\_misc.is\_outlier(delicacy\_price\_df\_clean.price,threshold=3.5) # 将异常值处理函数放置于util\_misc.py文件中，直接调用。见异常值处理一节  
  
delicacy\_price\_array\_dropna=delicacyPrice\_outliersDrop.to\_numpy().reshape(-1)  
print("original data mean:%.2f"%delicacy\_price\_array\_dropna.mean())  
  
print("original data K-S test:",stats.kstest(stats.zscore(delicacy\_price\_array\_dropna),cdf='norm')) # 计算标准化后的K-S test正态性检验  
  
delicacy\_price\_sample\_mus=np.array([np.random.choice(delicacy\_price\_array\_dropna,350).mean() for i in range(2000)])  
sns.histplot(delicacy\_price\_sample\_mus,bins=bins,kde=True,stat="density", linewidth=0)  
print("sample\_mus mu:%.2f,sigma:%.2f"%(delicacy\_price\_sample\_mus.mean(),delicacy\_price\_sample\_mus.std()))  
kstest\_test=stats.kstest(stats.zscore(delicacy\_price\_sample\_mus),cdf='norm')  
print("sample\_mus K-S test statistic:%.2f,p-value:%.2f"%kstest\_test)

original data mean:47.70  
original data K-S test: KstestResult(statistic=0.11464960111854455, pvalue=5.153632014592587e-26)  
sample\_mus mu:47.76,sigma:1.69  
sample\_mus K-S test statistic:0.01,p-value:0.76



#### 3）t分布 (Student’s t-distribution)

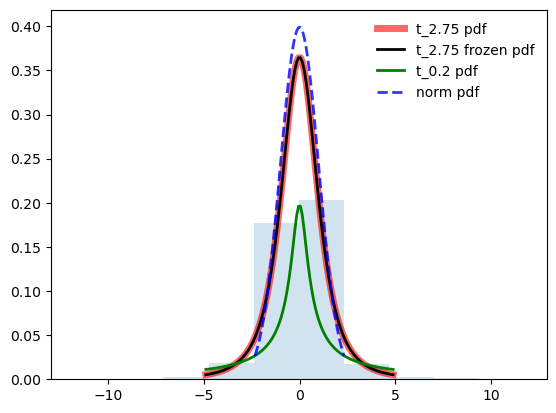
如果要使用正态分布，通过z-score（标准计分）获得精确概率，至少需要满足两个条件，一个是，总体标准差已知；另一个是，大样本（即样本容量足够大）。如果上述条件均不满足，则需要考虑样本容量，这就用到t分布。t分布的形状受样本容量影响，大样本条件下，t分布与正态分布的形状几何完全一样，而随样本容量减少，t分布的形状变得中间更平坦，两端更粗厚，即均值周围的取值变得更少，而远离均值，位于分布尾部的取值变得更多。以SciPy库[scipy.stats.t](https://docs.scipy.org/doc/scipy-0.13.0/reference/generated/scipy.stats.t.html)①给出的案例来观察t分布，其代码方法与正态分布基本相同，可以互相比较查看之间的差异。

t分布中，增加了一个新的参数自由度（degree of freedom，df），通常用符号ν来表示，当以样本的统计量来估计总体的参数时，对N个随机样本而言，其自由度为N-1。

数据集为正态分布，给定一个取值，计算该取值的z-score后，就可以获得该取值的概率。而在t分布下，要获取对应取值的概率，则需要计算t值（t统计量），其公式定义为：​，式中μ为总体均值，为样本均值， ​为标准误的样本估计。知道t值，及自由度（样本容量N-1），可以查t值表得到该取值的概率。当然，目前直接使用SciPy等库计算，无需再计算t值再对应查表。

from scipy.stats import norm  
from scipy.stats import t  
import matplotlib.pyplot as plt  
import numpy as np  
  
fig, ax=plt.subplots(1, 1)  
df=2.75 # 配置自由度  
mean, var, skew, kurt=t.stats(df, moments='mvsk') # 查看服从t分布的相关统计量  
print('mean, var, skew, kurt=',(mean, var, skew, kurt))  
  
x=np.linspace(t.ppf(0.01, df),t.ppf(0.99, df), 100) # 获取服从自由度df，概率位于1%到99%的100个取值  
ax.plot(x, t.pdf(x, df),'r-', lw=5, alpha=0.6, label='t\_%.2f pdf'%df)  
  
rv=t(df) # 指定固定自由度的随机变量（random variable）序列  
ax.plot(x, rv.pdf(x), 'k-', lw=2, label='t\_%.2f frozen pdf'%df)  
  
rv\_df\_02=t(0.2) # 指定固定自由度为0.2的随机变量序列  
ax.plot(x, rv\_df\_02.pdf(x), 'g-', lw=2, label='t\_0.2 pdf')  
  
vals=t.ppf([0.001, 0.5, 0.999], df) # 返回概率为0.1%、50%和99.9%的取值  
print("验证累积分布函数CDF返回值与其PPF返回值是否相等或近似：",np.allclose([0.001, 0.5, 0.999], t.cdf(vals, df)))  
  
rv\_1000=t.rvs(df, size=1000) # 获取服从自由度df的1000个随机取值  
ax.hist(rv\_1000, density=True, histtype='stepfilled', alpha=0.2)  
  
# 比较标准正态分布  
x\_nd=np.linspace(norm.ppf(0.01),norm.ppf(0.99), 100)  
ax.plot(x\_nd, norm.pdf(x\_nd), 'b--', lw=2, alpha=0.8, label='norm pdf')  
ax.legend(loc='best', frameon=False)  
plt.show()

mean, var, skew, kurt= (array(0.), array(3.66666667), array(nan), array(inf))  
验证累积分布函数CDF返回值与其PPF返回值是否相等或近似： True



# 如果需要计算服从t分布的概率则指定df，及loc（均值，默认为0）和scale(标准差，默认为1)  
print("用.cdf计算值小于或等于113的概率为：",t.cdf(113,df=999,loc=100,scale=12))   
print("用.sf计算值大于或等于113待概率为：",t.sf(113,df=999,loc=100,scale=12))   
print("可以观察到.cdf（<=113）概率结果+.sf(>=113)概率结果为：",t.cdf(113,df=999,loc=100,scale=12)+t.sf(113,df=999,loc=100,scale=12))  
print("用.ppf找到给定概率值为98%的数值为：",t.ppf(0.81766,df=999,loc=100,scale=12))

用.cdf计算值小于或等于113的概率为： 0.8605390547558804  
用.sf计算值大于或等于113待概率为： 0.13946094524411956  
可以观察到.cdf（<=113）概率结果+.sf(>=113)概率结果为： 1.0  
用.ppf找到给定概率值为98%的数值为： 110.88276310459135

#### 4）统计显著性

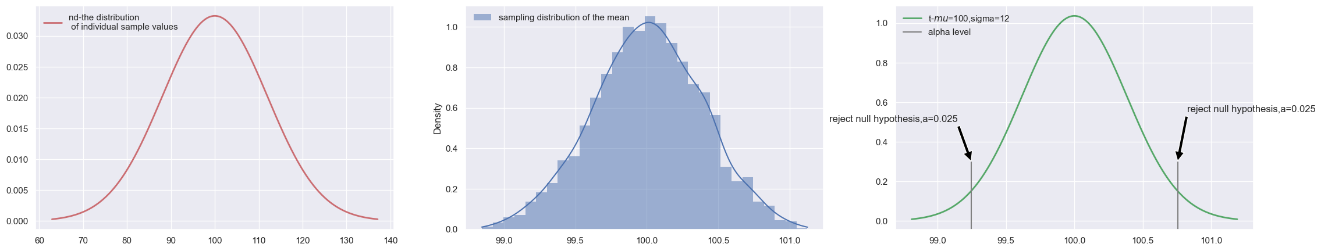
使用SciPy库建立服从均值为100，标准差为12的正态分布。绘制样本容量为1000的单个样本的随机取值分布；同时绘制样本容量为1000，样本数为2000的均值抽样分布。计算2000个样本均值抽样分布的标准差，将其作为绘制t分布的标准差，均值保持不变为100，自由度为1999（即样本容量为2000，等于均值抽样分布的样本数量）。观察均值抽样分布与对应的t分布（同均值和标准差，及自由度为均值抽样分布的样本数），曲线形状基本保持一致，再一次验证中心极限定理。在这样本容量为1000的2000个样本中，可以依据t分布估计样本均值概率小于0.05（5%）的取值为 99.387，大于0.95（95%）的取值为100.641。就此我们能够估计取值小于99.387或者大于100.641的样本均值出现的概率（机率）小于5%（0.05）。

如果我们不是根据概率获取对应的取值（即样本均值），而是估计取值的概率，例如估计样本均值小于或者等于99.0的概率为多少，计算结果为0.00439。这个概率值是否能确定样本统计量（例如样本均值）与总体参数（例如总体均值）之间某种差异（例如样本均值为99.0，而总体均值为100，100-99.0=1.0的差异），是否仅仅由随机抽样误差或偶然因素导致？惯例是取概率P值p-value=0.05作为一个水平的界限。这一水平就为，当原假设为真时所得到的样本观察结果或更极端结果出现的概率，其P值很小，说明原假设发生的概率很小，根据小概率原理，0.00439<0.05，有理由拒绝原假设。P值越小，拒绝原假设的理由越充分，就是说样本均值为99.0与总体均值之间的差异不是偶然的，得出结论认为99.0的样本均值实际上异于总计均值，从该样本中观察到的相关情况不能代表总体中的实际现象。

对于上述过程，可转换为假设检验的推断过程，提出一种假设并确立一种准则，用于决定保留或拒绝假设。假设样本均值不异于总体均值为原假设即零假设（null hypothesis，H0），零假设一般指总体中的效应不存在（总体均值与样本均值不会不同），符号表示为：，其中μ为总体均值，为样本均值。如果假设样本均值异于总体均值，则为对立假设（一种替代假设），符号表示为：。样本均值趋向于等同总体均值，但是因为毕竟不是总体，之间总会存在差异，引起这个差异的原因就是随机抽样误差或偶然因素，这正是均值抽样分布呈正态性的内在原因。其p-value<=0.05的概率对应着显著性水平（α，alpha level），例如样本均值99.0的概率为0.00439，小于显著性水平α，拒绝零假设，样本均值异于总体均值，其结果是统计显著的（指在原假设为真的条件下，用于检验的样本统计量的值落在了拒绝域内，做出了拒绝原假设的决定）。显著性水平常用值的除了0.05，还有0.01。

在上述计算过程中，计算有概率小于0.025，和大于0.975的两种情况，如果同时包括两种情况，则为双尾检测，仅有一种情况，但是概率需要小于0.05或者大于0.95的情况下为单尾检测。

import matplotlib.pyplot as plt  
df,loc,scale,sample\_size=1999,100,12,1000  
  
# 绘制样本容量为1000的单个样本的随机取值分布  
x=np.linspace(norm.ppf(0.001, loc=loc,scale=scale),norm.ppf(0.999, loc=loc,scale=scale), sample\_size)  
fig, axs=plt.subplots(1,3,figsize=(27,5))  
axs[0].plot(x, norm.pdf(x,loc=loc,scale=scale),'r-', lw=2, alpha=0.8, label='nd-the distribution \n of individual sample values')  
axs[0].legend(loc='upper left', frameon=False)  
  
# 绘制样本容量为1000，样本数为2000的均值抽样分布  
samples=np.array([norm.rvs(loc=loc,scale=scale,size=sample\_size).mean() for i in range(df+1)])  
bins=30  
sns.histplot(samples,bins=bins,ax=axs[1],label='sampling distribution of the mean',kde=True,stat="density",linewidth=0)  
axs[1].legend(loc='upper left', frameon=False)  
  
# 计算2000个样本均值抽样分布的标准差，作为绘制t分布的标准差，均值保持保持保持不变为100，自由度为1999（即样本容量为2000，等于均值抽样分布的样本数量）  
samples\_std=samples.std()  
t\_x=np.linspace(t.ppf(0.001, df=df,loc=loc,scale=samples\_std),t.ppf(0.999, df=df,loc=loc,scale=samples\_std), df+1) # 获取服从自由度df，概率位于1%到99%的100个取值  
t\_rv=t(df=df,loc=loc,scale=samples\_std) # 指定固定自由度  
axs[2].plot(t\_x, t\_rv.pdf(t\_x), 'g-', lw=2, label='t-$mu$=%d,sigma=%d'%(loc,scale))  
  
# 显著性水平为0.05时，曲线区间绘制  
pValue\_5percent=t.ppf(0.025,df=df,loc=loc,scale=samples\_std)  
pValue\_95percent=t.ppf(0.975,df=df,loc=loc,scale=samples\_std)  
axs[2].axvline(pValue\_5percent, 0,0.3,c='gray',label='alpha level')  
axs[2].axvline(pValue\_95percent, 0,0.3,c='gray')  
axs[2].annotate('reject null hypothesis,a=%.3f'%0.025,xy=(pValue\_5percent,0.3),xytext=(pValue\_5percent-0.1, 0.5),arrowprops=dict(facecolor='black', shrink=0.01),horizontalalignment='right',)  
axs[2].annotate('reject null hypothesis,a=%.3f'%0.025,xy=(pValue\_95percent,0.3),xytext=(pValue\_95percent+0.07, 0.55),arrowprops=dict(facecolor='black', shrink=0.01),horizontalalignment='left',)  
axs[2].legend(loc='upper left', frameon=False)  
  
plt.show()  
  
print("用.ppf找到给定概率值为5%的数值为：",pValue\_5percent)  
print("用.ppf找到给定概率值为95%的数值为：",pValue\_95percent)  
print("用.cdf计算值小于或等于99的概率为：",t.cdf(99.0,df=df,loc=loc,scale=samples\_std))



用.ppf找到给定概率值为5%的数值为： 99.24538217329646  
用.ppf找到给定概率值为95%的数值为： 100.75461782670354  
用.cdf计算值小于或等于99的概率为： 0.004710985065683475

#### 5）效应量

效应量（effect size）是量化现象强度的数值。其绝对值越大表示效应越强，也就是现象越明显。对于均值抽样分布，效应量的计算公式为：，式d为效应量， 为样本均值，μ为总体均值， s为标准差的样本估计。效应量所代表的就是以标准差为单位所度量的差异，这个与标准计分（z-score）非常相似，标准计分是以标准差为单位，从感兴趣的点到均值之间有多少个标准差。关于效应量是否有意义时，所建议的效应量值的区间也所有差异，关键是检验什么以及所持观点，一般而言效应量小于0.20算小，在0.25~0.75之间算中等，超过0.80算大。对于样本均值99的效应量计算结果为-2.583，其绝对值大于0.8，表明样本均值99到总体均值100之间所差的100-99=1的差异，按照标准差为单位，效应量显著，即样本均值99远离总体均值100。

print("样本均值99的效应量：",(99-100)/samples\_std)

样本均值99的效应量： -2.5988670698883993

#### 6）置信区间

一个概率样本的置信区间（Confidence interval，CI），是对产生这个样本的总体的参数分布（Parametric Distribution）中的某一个位置参数值，以区间的形式给出估计。对于现实中均值抽样分布，实际上总体的均值（总体参数的实际值）并不知道，我们所拥有的只是样本数据，而通过样本数据是可以估计总体均值，给出总体均值分布的区间，这个区间即为置信区间。计算公式为：，式中CI95​为95%置信区间； CI99​为99%置信区间；为样本均值； ；为标准误； t95​为给定自由度条件下， a水平为0.05双尾检验所对应的t值； t99​为给定自由度条件下， a水平为0.01双尾检验所对应的t值。

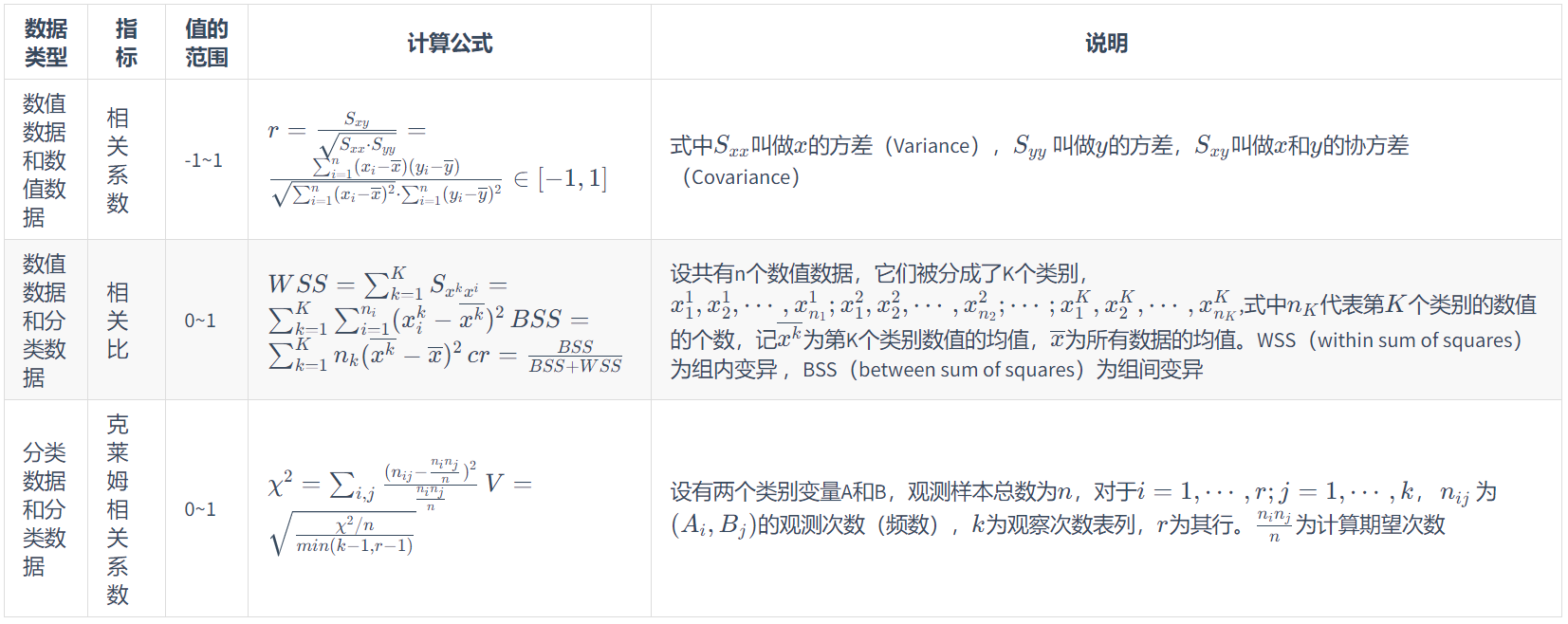
对于置信区间的计算可以直接使用SciPy库。

print("0.05\_置信区间：",stats.t.interval(0.95, len(samples)-1, loc=np.mean(samples), scale=stats.sem(samples)))  
print("0.01\_置信区间：",stats.t.interval(0.99, len(samples)-1, loc=np.mean(samples), scale=stats.sem(samples)))

0.05\_置信区间： (99.98972207782829, 100.02347805302809)  
0.01\_置信区间： (99.98441087421692, 100.02878925663946)

### 2.1.3.2 相关性

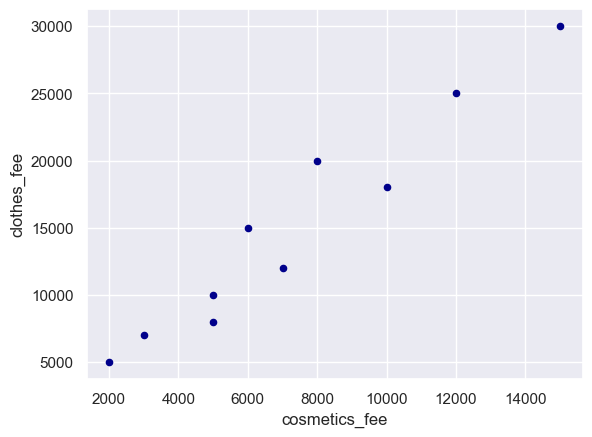
如果要判断两个变量之间是否相互联系，则需要计算相关系数。在变量之间相关性分析时，涉及到三种情况，包括数值数据与数值数据、数值数据与分类数据，及分类数据与分类数据。因此不同的变量类型之间的相关性计算指标有所差异：



对于数值数据和数值数据之间的相关性分析，使用相关系数计算（通常为Pearson’s r，即皮尔森相关系数 r，为线性相关）。先建立简单数据集，数据来源于《漫画统计学》询问10名20多岁女性的化妆品费和置装费[2]。待分析的两个变量为x和y值，打印散点图观察点的分布是否存在规律，可以比较明显的观察到点的分布似乎沿一条隐藏的倾斜直线分布，可以初步判断化妆品费和置装费之间存在关联。

皮尔森相关系数，又称积差相关系数、积矩相关系数（Pearson product-moment correlation coefficient，PPMCC或PCCs）

import pandas as pd  
  
dressUp\_cost={'name':["miss\_A","miss\_B","miss\_C","miss\_D","miss\_E","miss\_F","miss\_G","miss\_H","miss\_I","miss\_J"],  
 "cosmetics\_fee":[3000,5000,12000,2000,7000,15000,5000,6000,8000,10000],  
 "clothes\_fee":[7000,8000,25000,5000,12000,30000,10000,15000,20000,18000]  
 }  
dressUp\_cost\_df=pd.DataFrame.from\_dict(dressUp\_cost)  
dressUp\_cost\_df.plot.scatter(x='cosmetics\_fee',y='clothes\_fee',c='DarkBlue')  
plt.show()



使用上述公式计算相关系数。

人工计算的过程有助于理解概念及公式。

import math  
  
dressUp\_cost\_df["xSx\_mean"]=dressUp\_cost\_df.cosmetics\_fee.apply(lambda row: row-dressUp\_cost\_df.cosmetics\_fee.mean())   
dressUp\_cost\_df["ySy\_mean"]=dressUp\_cost\_df.clothes\_fee.apply(lambda row: row-dressUp\_cost\_df.clothes\_fee.mean())  
dressUp\_cost\_df["square\_xSx\_mean"]=dressUp\_cost\_df.xSx\_mean.apply(lambda row: math.pow(row,2))  
dressUp\_cost\_df["square\_ySy\_mean"]=dressUp\_cost\_df.ySy\_mean.apply(lambda row: math.pow(row,2))  
dressUp\_cost\_df["xSx\_meanMySy\_mean"]=dressUp\_cost\_df.xSx\_mean\*dressUp\_cost\_df.ySy\_mean  
r=dressUp\_cost\_df.xSx\_meanMySy\_mean.sum()/math.sqrt(dressUp\_cost\_df.square\_xSx\_mean.sum()\*dressUp\_cost\_df.square\_ySy\_mean.sum())  
print("Pearson's r:",r)  
  
dressUp\_cost\_df

Pearson's r: 0.968019612860768

|  | **name** | **cosmetics\_fee** | **clothes\_fee** | **xSx\_mean** | **ySy\_mean** | **square\_xSx\_mean** | **square\_ySy\_mean** | **xSx\_meanMySy\_mean** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **0** | miss\_A | 3000 | 7000 | -4300.0 | -8000.0 | 18490000.0 | 64000000.0 | 34400000.0 |
| **1** | miss\_B | 5000 | 8000 | -2300.0 | -7000.0 | 5290000.0 | 49000000.0 | 16100000.0 |
| **2** | miss\_C | 12000 | 25000 | 4700.0 | 10000.0 | 22090000.0 | 100000000.0 | 47000000.0 |
| **3** | miss\_D | 2000 | 5000 | -5300.0 | -10000.0 | 28090000.0 | 100000000.0 | 53000000.0 |
| **4** | miss\_E | 7000 | 12000 | -300.0 | -3000.0 | 90000.0 | 9000000.0 | 900000.0 |
| **5** | miss\_F | 15000 | 30000 | 7700.0 | 15000.0 | 59290000.0 | 225000000.0 | 115500000.0 |
| **6** | miss\_G | 5000 | 10000 | -2300.0 | -5000.0 | 5290000.0 | 25000000.0 | 11500000.0 |
| **7** | miss\_H | 6000 | 15000 | -1300.0 | 0.0 | 1690000.0 | 0.0 | -0.0 |
| **8** | miss\_I | 8000 | 20000 | 700.0 | 5000.0 | 490000.0 | 25000000.0 | 3500000.0 |
| **9** | miss\_J | 10000 | 18000 | 2700.0 | 3000.0 | 7290000.0 | 9000000.0 | 8100000.0 |

使用SciPy库直接计算，其结果保持一致。

from scipy import stats  
  
r\_=stats.pearsonr(dressUp\_cost\_df.cosmetics\_fee,dressUp\_cost\_df.clothes\_fee)  
print(  
 "pearson's r:",r\_[0],"\n",  
 "p\_value:",r\_[1]  
 )

pearson’s r: 0.968019612860768 p\_value: 4.402991448166131e-06

相关系数的强弱并没有严格的规定，通常需要根据具体的应用背景和目的确定，例如复杂多变因素影响的变量，0.9的值则是相当高的。对于相关系数值的意义不同的文献给出的参考意义也有所差别，这里给出《漫画统计学》里的划分：

| 相关系数的绝对值 | 若细分 | 若大体上划分 |
| --- | --- | --- |
| 1.0~0.9 | 非常强 | 相关 |
| 0.9~0.7 | 有点强 | 相关 |
| 0.7~0.5 | 有点弱 | 相关 |
| 未满0.5 | 非常弱 | 不相关 |

及Wikipedia给出的参考：

| 相关性 | 负 | 正 |
| --- | --- | --- |
| 无 | −0.09 to 0.0 | 0.0 to 0.09 |
| 弱 | −0.3 to −0.1 | 0.1 to 0.3 |
| 中 | −0.5 to −0.3 | 0.3 to 0.5 |
| 强 | −1.0 to −0.5 | 0.5 to 1.0 |

若相关系数接近±1，则相关性越强；如果接近0，则相关性越弱。如果为正，则为正相关；如果为负，则为负相关。

通过计算相关系数，获得结果r=0.968，说明抽样的10名20多岁女性在化妆品费和置装费这两个变量间存在强相关性，那么该值是否代表其抽样的总体（所有女性）中两个变量（化妆品费和置装费）间也存在非常强的相关关系？为了确认这个现象，需要检验相关系数是否统计显著。提出零假设，代表总体下两个变量（化妆品费和置装费）之间完全无关，即总体的相关系数为0。通常用t分布来检验相关系数是否统计显著，进行t检验。其t值的计算公式为：式中自由度为N−2，即样本对象数减去2，即N-2=10-2=8。使用t分布的累计分布函数t.cdf()或者t.sf()计算P值，因为双尾检查，因此再乘以2，其计算结果与stats.pearsonr()计算相关系数时给出的P值保持一致。因为p\_value=4.402991448104743e-06<0.05，因此拒绝原假设，即拒绝总体下两个变量（化妆品费和置装费）之间完全无关，就是说总体下两个变量是相关的。因此所计算的相关系数值为0.968说明了总体所有女性在化妆品费和置装费的消费上是强关联的，如果一个女性花费较多的钱在化妆品费上，那么她花费在置装费的钱也相对较多，反之亦然。

t\_value=r\*math.sqrt((10-2)/(1-math.pow(r,2)))  
print("p-value\_cdf:",(1-stats.t.cdf(t\_value,10-2))\*2)  
print("p-value\_sf:",stats.t.sf(t\_value,10-2)\*2)

p-value\_cdf: 4.402991448104743e-06 p-value\_sf: 4.402991448166121e-06

### 2.1.3.3 卡方分布与独立性检验

#### 1） 卡方分布（Chi-Square Distribution，χ²-distribution）[1][2][3]

若k个随机变量Z1​,……,Zk​，是相互独立，符合标准正态分布的随机变量（数学期望为0，方差为1），则随机变Z的平方和​被称为服从自由度为k的卡方分布，记作X∼x2(k)或X∼xk2​。卡方分布的概率密度函数（即计算卡方分布曲线的公式）为：式中x≥0，当x≤0时fk​(x)=0。Γ代表Gamma函数。在Python中绘制卡方分布，仍旧与正态分布、t分布一样，直接由SciPy库完成。

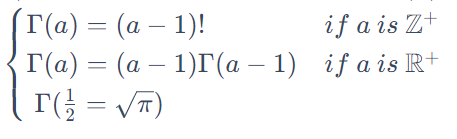
因为卡方分布表述的是多个事件（随机变量）的机率，每个事件符合标准正态分布，而标准正态分布表为记录对应横轴刻度的机率表，卡方分布表则是记录对应几率的横轴刻度表。

对于卡方分布的理解，可以结合比较均值抽样分布，二者具有类似的逻辑。可以表述为从同一总体中抽取相同容量样本平方和的分布，假设从服从平均值为30，标准差为5的正态分布中，随机提取2000个样本（事件，或随机变量，即），各个样本的容量为1000，计算每一样本的平方和，观察这2000个样本的卡方分布情况。从下述实验打印结果来看，趋近服从自由度为的分布。即可以通过一个检验统计量（平方和）来比较期望结果和实际结果之间的差别，然后得出观察频数极值的发生概率。因此以特定概率分布为某种情况建模时，事件长期结果较为稳定，能够清晰进行把握。但是如果期望与事实存在差异时，则可以应用卡方分布判断偏差是正常的小幅度波动还是建模上的错误。一是，可以检验一组给定数据与指定分布的吻合程度；二是，可以检验两个变量的独立性，即变量之间是否存在某种关系。

* *Γ函数*

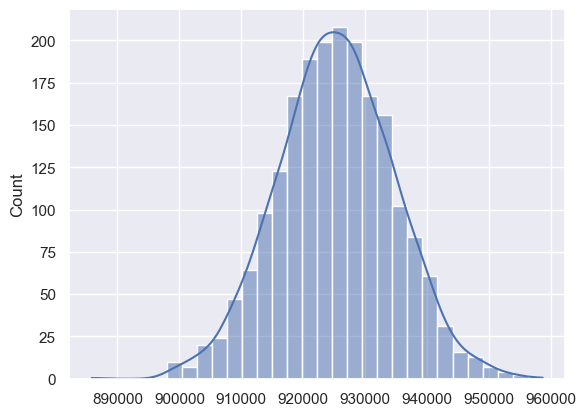
在数学中，Γ函数，也称为伽马函数（Gamma函数），是阶乘函数在实数与复数域上的扩展。如果*n*为正整数，则：Γ(*n*)=(*n*−1)!，即正整数的阶乘；对于实数部分为正的复数*z*，伽马函数定义为：。发现Γ函数的起因是数列插值问题，即找到一个光滑的曲线连接那些由*y*=(*x*−1)!所给定的点 (*x*,*y*)，并要求*x*为正整数。但是如果*x*由正整数拓展到实数，即可以计算2!,3!,…,，那么是否可以计算2.5！，并绘制 (*n*,*n*!)的平滑曲线？而Γ函数正是借由微积分的积分与极限表达阶乘。

伽马（Gamma）分布，假设*X*1​,*X*2​,…,*Xn*​为连续发生事件的等候时间，且这*n*次等侯时间为独立的，那么这*n*此等候时间之和*Y*(*Y*=*X*1​,*X*2​,…,*Xn*​)服从伽马分布，即*Y*∼*Gamma*(*α*,*β*)，式中*α*=*n*,*β*=*γ*，*α*是伽马分布中的母数，称为形状参数，*β*为尺度参数。*γ*是连续发生事件的平均发生频率。指数分布是伽马分布*α*=1的特殊情况。

令*X*∼Γ(*α*,*β*)，*λ*=*β*（即*X*∼Γ(*α*,*γ*)），则伽马分布的概率密度函数为：，式中伽马函数的特征为：

import numpy as np  
from scipy import stats  
import seaborn as sns  
import math  
  
sns.set()  
mu,sigma=30,5  
sample\_size=1000  
sample\_square=np.array([sum(np.square(np.random.normal(mu, sigma, sample\_size))) for i in range(2000)]) # 从服从平均值为30，标准差为5的正态分布中，随机提取2000个样本，每个样本容量为1000的样本，并计算每一样本的均值  
bins=30  
sns.histplot(sample\_square,bins=bins,kde=True,legend=False) # 查看2000个样本平方和的分布

<AxesSubplot:ylabel=‘Count’>

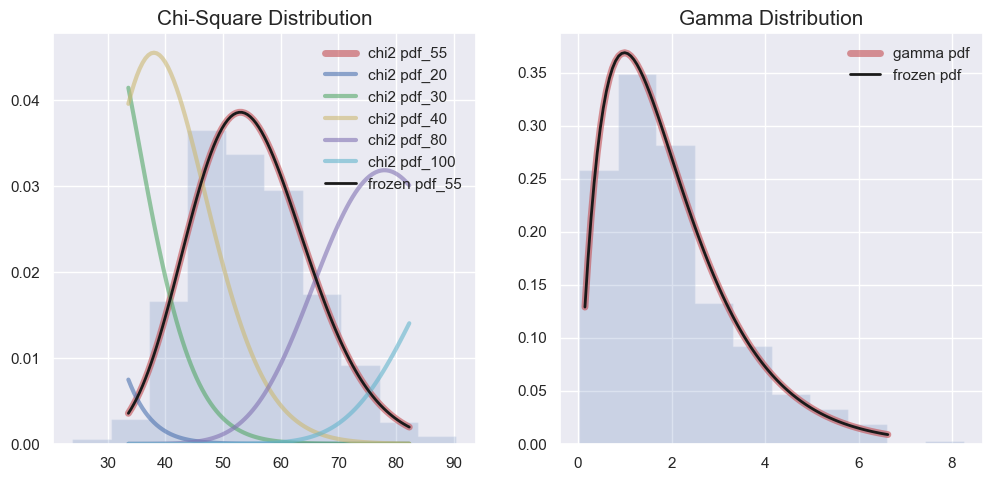


使用SciPy库计算打印卡方分布及伽马分布。

import matplotlib.pyplot as plt  
from scipy.stats import chi2  
import numpy as np  
  
fig, axs=plt.subplots(1,2,figsize=(18/1.5,8/1.5))   
  
# A-卡方分布  
df=55  
mean,var,skew,kurt=chi2.stats(df, moments='mvsk')  
# 打印卡方分布的概率密度函数 Display the probability density function (pdf)  
x=np.linspace(chi2.ppf(0.01, df),chi2.ppf(0.99, df), 100)  
axs[0].plot(x, chi2.pdf(x, df),'r-', lw=5, alpha=0.6, label='chi2 pdf\_55')  
  
df\_lst=[20,30,40,80,100]  
fmts=['b-','g-','y-','m-','c-']  
for i in range(len(df\_lst)):  
 axs[0].plot(x, chi2.pdf(x, df\_lst[i]),fmts[i], lw=3, alpha=0.6, label='chi2 pdf\_%d'%df\_lst[i])  
   
# 固定分布 Alternatively, freeze the distribution and display the frozen pdf  
rv=chi2(df)  
axs[0].plot(x, rv.pdf(x), 'k-', lw=2, label='frozen pdf\_55')  
  
vals=chi2.ppf([0.001, 0.5, 0.999], df)  
print("Chi\_2\_Check accuracy of cdf and ppf:",np.allclose([0.001, 0.5, 0.999], chi2.cdf(vals, df)))  
  
r=chi2.rvs(df, size=1000)  
axs[0].hist(r, density=True, histtype='stepfilled', alpha=0.2)  
axs[0].legend(loc='best', frameon=False)  
   
# B-Gamma分布   
from scipy.stats import gamma  
a=1.99323054838  
mean\_,var\_,skew\_,kurt\_=gamma.stats(a, moments='mvsk')  
# 打印Gamma分布的概率密度函数 Display the probability density function (pdf):  
x\_= np.linspace(gamma.ppf(0.01, a),gamma.ppf(0.99, a), 100)  
axs[1].plot(x\_, gamma.pdf(x\_, a),'r-', lw=5, alpha=0.6, label='gamma pdf')  
# Alternatively, freeze the distribution and display the frozen pdf:  
rv\_= gamma(a)  
axs[1].plot(x\_, rv\_.pdf(x\_), 'k-', lw=2, label='frozen pdf')  
  
vals = gamma.ppf([0.001, 0.5, 0.999], a)  
print("Gamma\_Check accuracy of cdf and ppf:",np.allclose([0.001, 0.5, 0.999], gamma.cdf(vals, a)))  
  
r\_=gamma.rvs(a, size=1000)  
axs[1].hist(r\_, density=True, histtype='stepfilled', alpha=0.2)  
axs[1].legend(loc='best', frameon=False)  
  
axs[0].set\_title(r'Chi-Square Distribution', fontsize=15)  
axs[1].set\_title(r'Gamma Distribution', fontsize=15)  
plt.show()

Chi\_2\_Check accuracy of cdf and ppf: True

Gamma\_Check accuracy of cdf and ppf: True



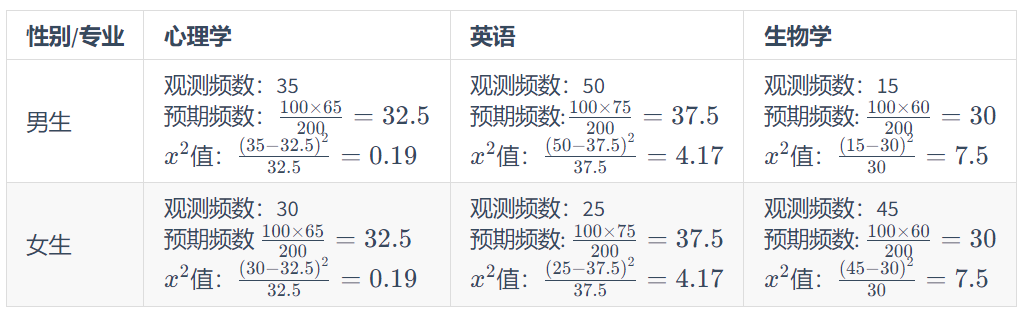
#### 2） （卡方）独立性检验

卡方检验（Chi-Squared Test，或 Test），是假设检验的一种，一种非参数假设检验，主要用于类别/分类变量（类别变量就是取值为离散值的变量，例如性别即为一个类别变量，有男女两类，又或者国籍、学科、植物等的类别等），在没有其它的限制条件或说明时，卡方检验一般指代的是皮尔森卡方（Pearson）检验。1900年，Pearson发表了著名的检验的论文，假设实验中从总体随机取样得到的个观察值被划分为个互斥的分类中，这样每个分类都有一个对应实际的观察次数（或观测频数，observed frequencies）xi​(i=1,2,…,k)。对实验中各个观察值落入第个分类的概率的分布提出零假设，获得对应所有第分类的理论期望次数（或预期频数，expected frequencies）及限制条件，。在上述零假设成立以及n趋向∞∞时，以下统计量的极限分布趋向x2分布，。*X*2值的计算公式通常表示为：，其中， O为各个单元格（对列联表而言）的观测值（观测频数），E为各个单元格的预期值（预期频数）。零假设中所有分类的理论期望次数mi​均为足够大且已知的情况，同时假设各分类的实际观察次xi​均服从正态分布，得出样本容量n足够大时， x2趋近服从自由度为 (k−1)的x2分布。

通常将用于卡方检验的数据以表格的形式给出并依据表格进行计算，这个表格即为列联表（contingency tabel）。以《白话统计学》性别与专业的修订数据为例，

| 性别/专业 | 心理学 | 英语 | 生物学 | 行合计 |
| --- | --- | --- | --- | --- |
| 男生 | 35 | 50 | 15 | 100 |
| 女生 | 30 | 25 | 45 | 100 |
| 列合计 | 65 | 75 | 60 | 200 |

利用表格每一单元格中的观测频数，及行、列和整个样本的合计频数，计算每个单元格的预期频数。男女两行中每一单元格的预期值都相等，是因为样本中的男女生人数相等。并根据上述的X2值的计算公式，计算X2，其和为：0.19+0.19+4.17+4.17+7.5+7.5=23.72。



注意到x2值较大是因为男女生在选择英语或生物专业时存在相对较大的差异。而心理专业的观测值和预期值之差相对较小，对整体x2值的贡献不大。获得观测的x2值，则需要查表（或程序）查找临界x2值，其自由df=(R−1)(C−1)=(2−1)×(3−1)=2，使用SciPy的chi2.ppf(q=1-0.05,df=2)计算可得0.05的α水平，自由度为2的条件下临界x2值为5.99，而观测的x2值为23.72，所以可以得出结论，男女生在专业选择上存在统计显著的差异。而因为观测的x2值足够大，在0.001的显著性水平上chi2.ppf(q=1-0.001,df=2)（临界值为13.815510557964274），也是统计显著的（即p<0.001）。

print("𝑝<0.05,df=2,Chi-Squared=%.3f"%chi2.ppf(q=1-0.05,df=2))  
print("𝑝<0.001,df=2,Chi-Squared=%.3f"%chi2.ppf(q=1-0.001,df=2))

𝑝<0.05,df=2,Chi-Squared=5.991

𝑝<0.001,df=2,Chi-Squared=13.816

使用SciPy的chi2\_contingency方法计算列联表，其计算结果与手工计算结果保持一致。

from scipy.stats import chi2\_contingency  
import numpy as np  
  
schoolboy=(35,50,15)  
schoolgirl=(30,25,45)  
statistical\_data=np.array([schoolboy,schoolgirl])  
chi2\_results=chi2\_contingency(statistical\_data)  
print("卡方值：%.3f \n P值：%.10f \n 自由度:%d \n 对应预期频数（期望值）：\n %s"%chi2\_results)

卡方值：23.718

P值：0.0000070748

自由度:2

对应预期频数（期望值）： [[32.5 37.5 30. ]

[32.5 37.5 30. ]]

#### 3） 协方差估计（Covariance Estimators）

统计学上常用的统计量包括平均值、方差、标准差等。平均值描述了样本集合的中间点；方差描述了一组数据与其平均值的偏离程度，方差越小，数据越集中，方差越大，数据越离散；标准差描述了样本集中各个样本点到均值的距离的平均值，同方差，描述数据集的集聚离散程度。这些统计量是针对一维数组，到处理高维时，用到协方差，度量多个随机变量关系的统计量，结果均为正则正相关，都为负则负相关，均趋近于0，则不相关。协方差是计算不同特征之间的统计量，不是不同样本之间的统计量。同时，协方差的大小，除了和变量之间的相关程度有关，也与变量本身的方差大小有关，因此引入相关系数，移除变量本身的影响。在协方差计算时可以使用协方差（矩阵）计算公式（查看方差和协方差部分），而有时并不使用全部的样本数据计算协方差矩阵，而是利用部分样本数据计算，这是就需要考虑样本计算得到的协方差矩阵是否和总体的协方差矩阵相同和近似。大多数情况下，估计总体的协方差矩阵必须在样本的性质（大小size，结构structure，同质性homogeneity）对估计质量有很大影响下进行，在sklearn.covariance模块中则提供了多个健壮的协方差估计算法，列表如下：

| 协方差估计方法 | 解释 |
| --- | --- |
| covariance.EmpiricalCovariance(\*[, …]) | 最大似然协方差估计 Maximum likelihood covariance estimator |
| covariance.EllipticEnvelope(\*[, …]) | 用于检测高斯分布数据集中异常值的对象 An object for detecting outliers in a Gaussian distributed dataset. |
| covariance.GraphicalLasso([alpha, mode, …]) | 带L1惩罚估计量的稀疏逆协方差估计 Sparse inverse covariance estimation with an l1-penalized estimator. |
| covariance.GraphicalLassoCV(\*[, alphas, …]) | 稀疏逆协方差w/交叉验证l1惩罚的选择 Sparse inverse covariance w/ cross-validated choice of the l1 penalty. |
| covariance.LedoitWolf(\*[, store\_precision, …]) | LedoitWolf估计量 LedoitWolf Estimator |
| covariance.MinCovDet(\*[, store\_precision, …]) | 最小协方差行列式(MCD):协方差的稳健估计 Minimum Covariance Determinant (MCD): robust estimator of covariance. |
| covariance.OAS(\*[, store\_precision, …]) | Oracle逼近收缩估计 Oracle Approximating Shrinkage Estimator |
| covariance.ShrunkCovariance(\*[, …]) | 协方差缩水（shrinkage）估计 Covariance estimator with shrinkage |
| covariance.empirical\_covariance(X, \*[, …]) | 计算最大似然协方差估计量 Computes the Maximum likelihood covariance estimator |
| covariance.graphical\_lasso(emp\_cov, alpha, \*) | l1-惩罚项协方差估计量 l1-penalized covariance estimator |
| covariance.ledoit\_wolf(X, \*[, …]) | 估计缩水的Ledoit-Wolf协方差矩阵 Estimates the shrunk Ledoit-Wolf covariance matrix. |
| covariance.oas(X, \*[, assume\_centered]) | 用Oracle近似缩水算法估计协方差 Estimate covariance with the Oracle Approximating Shrinkage algorithm. |
| covariance.shrunk\_covariance(emp\_cov[, …]) | 计算在对角线上缩水的协方差矩阵 Calculates a covariance matrix shrunk on the diagonal |

下述假设了一个协方差矩阵，并根据该协方差矩阵生产一组数据集，分布使用了sklearn.covariance提供的GraphicalLassoCV，EmpiricalCovariance，MinCovDet，及NumPy库提供的np.cov()方法进行计算比较观察，其结果相近，向真实假设的协方差矩阵值靠近。

import numpy as np  
from sklearn.covariance import GraphicalLassoCV,EmpiricalCovariance,MinCovDet  
  
# 假设的协方差矩阵，包含4个特征量  
true\_cov=np.array([[0.8, 0.0, 0.2, 0.0],  
 [0.0, 0.4, 0.0, 0.0],  
 [0.2, 0.0, 0.3, 0.1],  
 [0.0, 0.0, 0.1, 0.7]])  
np.random.seed(0)  
# 生成满足假设协方差矩阵的特征值  
X=np.random.multivariate\_normal(mean=[0, 0, 0, 0], cov=true\_cov,size=200)  
# A-使用GraphicalLassoCV方法  
cov=GraphicalLassoCV().fit(X)  
print("A-GraphicalLassoCV algorithm:\n{},estimated location(the estimated mean):{}".format(np.around(GraphicalLassoCV().fit(X).covariance\_, decimals=3),np.around(cov.location\_, decimals=3)))  
  
# B-EmpiricalCovariance  
print("A-EmpiricalCovariance algorithm:\n{}".format(np.around(EmpiricalCovariance().fit(X).covariance\_, decimals=3)))  
  
# C-MinCovDet  
print("A-MinCovDet:\n{}".format(np.around(MinCovDet().fit(X).covariance\_, decimals=3)))  
  
# D-np.cov  
print("A-np.cov:\n{}".format(np.around(np.cov(X.T), decimals=3)))

1. GraphicalLassoCV algorithm: [[0.816 0.051 0.22 0.017] [0.051 0.364 0.018 0.036] [0.22 0.018 0.322 0.094] [0.017 0.036 0.094 0.69 ]],estimated location(the estimated mean):[0.073 0.04 0.038 0.143] A-EmpiricalCovariance algorithm: [[0.816 0.059 0.228 0.009] [0.059 0.364 0.025 0.044] [0.228 0.025 0.322 0.103] [0.009 0.044 0.103 0.69 ]] A-MinCovDet: [[ 0.741 -0.005 0.162 0.089] [-0.005 0.305 0.024 0.061] [ 0.162 0.024 0.237 0.117] [ 0.089 0.061 0.117 0.55 ]] A-np.cov: [[0.82 0.059 0.229 0.009] [0.059 0.366 0.025 0.044] [0.229 0.025 0.324 0.103] [0.009 0.044 0.103 0.694]]

### 2.1.3.4 公共健康数据的地理空间分布与相关性分析

#### 1）公共健康数据的地理空间分布

公共健康数据为芝加哥社区选定的公共健康（卫生）指标（public health indicator）。该数据集包括27项重要的公共健康指标，这些指标为比率、百分比、或者出生率、死亡率、传染病、铅中毒及经济状况相关的指标。该数据集是按照社区进行统计，在字段中给出了社区名，因此可以将数据按照社区名配对到各个社区范围的地理空间数据上。芝加哥社区边界及公共健康数据均来源于[Chicago Data Portal，CDP](https://data.cityofchicago.org/)②。关于数据的细节描述可以参考CDP提供的文件。

在JupyterLab中打印数据的方式，包括print()和直接将要显示数据的变量放置于单个单元格（cell）两种方式。但是前者显示为固定宽度，不能按照屏幕宽度自动缩放，后者虽为100%宽度自动缩放显示，但是如果要将.ipynb的Jupyter文件输出为.md的Mardown文件，可能会产生乱码。因此，可以使用from IPython.display import HTML，将要显示的数据转换为HTML格式数据后，会解决上述两种问题。

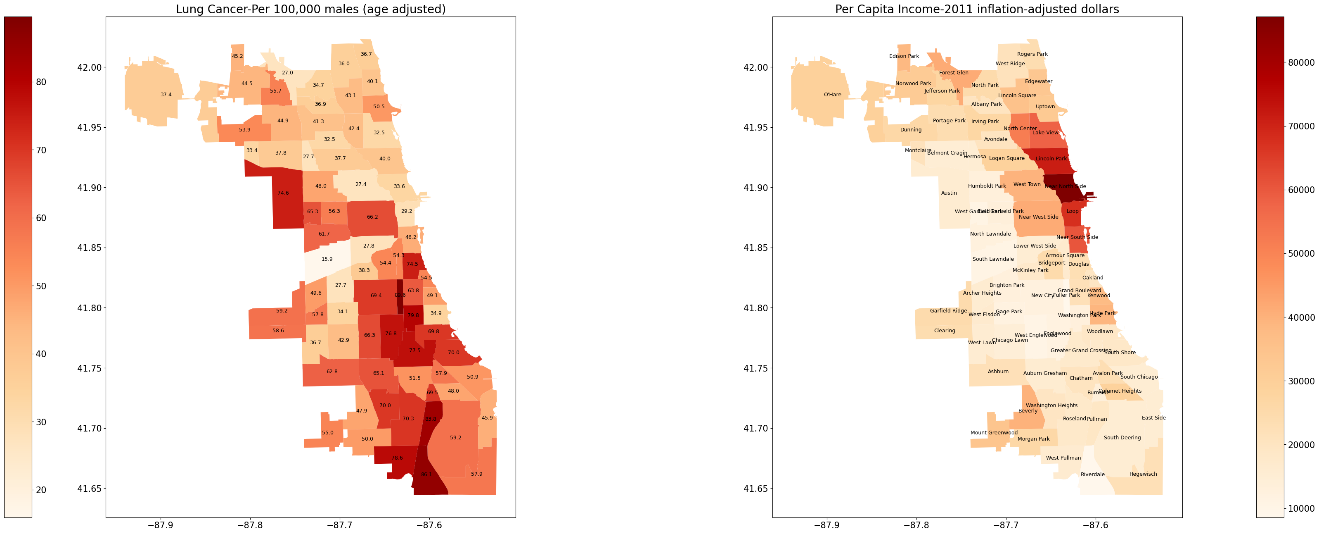
import pandas as pd  
  
dataFp\_dic={  
 "ublic\_Health\_Statistics\_byCommunityArea\_fp":r'./data/Public\_Health\_Statistics\_Selected\_public\_health\_indicators\_by\_Chicago\_community\_area.csv',  
 "Boundaries\_Community\_Areas\_current":r'./data/ChicagoCommunityAreas/ChicagoCommunityAreas.shp',   
}  
  
pubicHealth\_Statistic=pd.read\_csv(dataFp\_dic["ublic\_Health\_Statistics\_byCommunityArea\_fp"])  
  
# 中英对照表（字段映射表）  
PubicHealth\_Statistic\_columns={'Community Area':'社区',   
 'Community Area Name':'社区名',  
 'Birth Rate':'出生率',  
 'General Fertility Rate':'一般生育率',  
 'Low Birth Weight':'低出生体重',  
 'Prenatal Care Beginning in First Trimester':'产前3个月护理',   
 'Preterm Births':'早产',  
 'Teen Birth Rate':'青少年生育率',  
 'Assault (Homicide)':'攻击（杀人）',  
 'Breast cancer in females':'女性乳腺癌',  
 'Cancer (All Sites)':'癌症',   
 'Colorectal Cancer':'结肠直肠癌',  
 'Diabetes-related':'糖尿病相关',  
 'Firearm-related':'枪支相关',  
 'Infant Mortality Rate':'婴儿死亡率',   
 'Lung Cancer':'肺癌',  
 'Prostate Cancer in Males':'男性前列腺癌',  
 'Stroke (Cerebrovascular Disease)':'中风(脑血管疾病)',  
 'Childhood Blood Lead Level Screening':'儿童血铅水平检查',  
 'Childhood Lead Poisoning':'儿童铅中毒',  
 'Gonorrhea in Females':'女性淋病',   
 'Gonorrhea in Males':'男性淋病',   
 'Tuberculosis':'肺结核',  
 'Below Poverty Level':'贫困水平以下',   
 'Crowded Housing':'拥挤的住房',   
 'Dependency':'依赖',  
 'No High School Diploma':'没有高中文凭',   
 'Per Capita Income':'人均收入',  
 'Unemployment':'失业',  
 }  
  
def print\_html(df,row\_numbers=5):   
 '''  
 function - 在Jupyter中打印DataFrame格式数据为HTML  
   
 Params:  
 df - 需要打印的DataFrame或GeoDataFrame格式数据；DataFrame  
 row\_numbers - 打印的行数，如果为正，从开始打印如果为负，从末尾打印；int  
   
 Returns:  
 转换后的HTML格式数据；  
 '''  
 from IPython.display import HTML  
   
 if row\_numbers>0:  
 return HTML(df.head(row\_numbers).to\_html())  
 else:  
 return HTML(df.tail(abs(row\_numbers)).to\_html())  
print\_html(pubicHealth\_Statistic,6)

|  | Community Area | Community Area Name | Birth Rate | General Fertility Rate | Low Birth Weight | Prenatal Care Beginning in First Trimester | Preterm Births | Teen Birth Rate | Assault (Homicide) | Breast cancer in females | Cancer (All Sites) | Colorectal Cancer | Diabetes-related | Firearm-related | Infant Mortality Rate | Lung Cancer | Prostate Cancer in Males | Stroke (Cerebrovascular Disease) | Childhood Blood Lead Level Screening | Childhood Lead Poisoning | Gonorrhea in Females | Gonorrhea in Males | Tuberculosis | Below Poverty Level | Crowded Housing | Dependency | No High School Diploma | Per Capita Income | Unemployment |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| 0 | 1 | Rogers Park | 16.4 | 62.0 | 11.0 | 73.0 | 11.2 | 40.8 | 7.7 | 23.3 | 176.9 | 25.3 | 77.1 | 5.2 | 6.4 | 36.7 | 21.7 | 33.7 | 364.7 | 0.5 | 322.5 | 423.3 | 11.4 | 22.7 | 7.9 | 28.8 | 18.1 | 23714 | 7.5 |
| 1 | 2 | West Ridge | 17.3 | 83.3 | 8.1 | 71.1 | 8.3 | 29.9 | 5.8 | 20.2 | 155.9 | 17.3 | 60.5 | 3.7 | 5.1 | 36.0 | 14.2 | 34.7 | 331.4 | 1.0 | 141.0 | 205.7 | 8.9 | 15.1 | 7.0 | 38.3 | 19.6 | 21375 | 7.9 |
| 2 | 3 | Uptown | 13.1 | 50.5 | 8.3 | 77.7 | 10.3 | 35.1 | 5.4 | 21.3 | 183.3 | 20.5 | 80.0 | 4.6 | 6.5 | 50.5 | 25.2 | 41.7 | 353.7 | 0.5 | 170.8 | 468.7 | 13.6 | 22.7 | 4.6 | 22.2 | 13.6 | 32355 | 7.7 |
| 3 | 4 | Lincoln Square | 17.1 | 61.0 | 8.1 | 80.5 | 9.7 | 38.4 | 5.0 | 21.7 | 153.2 | 8.6 | 55.4 | 6.1 | 3.8 | 43.1 | 27.6 | 36.9 | 273.3 | 0.4 | 98.8 | 195.5 | 8.5 | 9.5 | 3.1 | 25.6 | 12.5 | 35503 | 6.8 |
| 4 | 5 | North Center | 22.4 | 76.2 | 9.1 | 80.4 | 9.8 | 8.4 | 1.0 | 16.6 | 152.1 | 26.1 | 49.8 | 1.0 | 2.7 | 42.4 | 15.1 | 41.6 | 178.1 | 0.9 | 85.4 | 188.6 | 1.9 | 7.1 | 0.2 | 25.5 | 5.4 | 51615 | 4.5 |
| 5 | 6 | Lake View | 13.5 | 38.7 | 6.3 | 79.1 | 8.1 | 15.8 | 1.4 | 20.1 | 126.9 | 13.0 | 38.5 | 1.8 | 2.2 | 32.5 | 17.0 | 24.4 | 179.2 | 0.4 | 81.8 | 357.6 | 3.2 | 10.5 | 1.2 | 16.5 | 2.9 | 58227 | 4.7 |

在数据合并时，公共健康数据以”Community Area”为对位列，社区边界数据以”area\_numbe”为对位列，使用df.merge()方法完成融合。打印两组数据，其中肺癌（Lung Cancer）为每100,000人所占的比例。GeoDataFrame格式数据在打印时如果需要在图上每个Polygon上标注数据增加地图的信息量，可以按照下述代码处理。

import geopandas as gpd  
from mpl\_toolkits.axes\_grid1 import make\_axes\_locatable  
import matplotlib.pyplot as plt  
  
community\_area=gpd.read\_file(dataFp\_dic["Boundaries\_Community\_Areas\_current"])  
print(community\_area.dtypes)   
community\_area.area\_numbe=community\_area.area\_numbe.astype('int64')  
print("\_"\*50)  
print("boundaries\_community.area\_numbe dtype:",community\_area.area\_numbe.dtypes)  
pubicHealth\_gpd=community\_area.merge(pubicHealth\_Statistic,left\_on='area\_numbe', right\_on='Community Area')  
  
fig, axs=plt.subplots(1,2,figsize=(40, 40))  
  
# 打印第1组数据  
divider=make\_axes\_locatable(axs[0])   
cax\_0=divider.append\_axes("left", size="5%", pad=0.1) # 配置图例参数  
pubicHealth\_gpd.plot(column='Lung Cancer',ax=axs[0],cax=cax\_0,legend=True,cmap='OrRd') # 如果提示如下错误，则需要按照信息安装对应的库,ImportError: The descartes package is required for plotting polygons in geopandas. You can install it using 'conda install -c conda-forge descartes' or 'pip install descartes'.  
axs[0].set\_title("Lung Cancer-Per 100,000 males (age adjusted)",fontsize=20)  
pubicHealth\_gpd.apply(lambda x: axs[0].annotate(text=x["Lung Cancer"], xy=x.geometry.centroid.coords[0], ha='center'),axis=1) #增加标注  
axs[0].tick\_params(axis='both', labelsize=15)  
cb\_ax\_0=axs[0].figure.axes[2]  
cb\_ax\_0.tick\_params(labelsize=15)  
  
# 打印第2组数据  
divider=make\_axes\_locatable(axs[1])  
cax\_1=divider.append\_axes("right", size="5%", pad=0.1)  
pubicHealth\_gpd.plot(column='Per Capita Income',ax=axs[1],cax=cax\_1,legend=True,cmap='OrRd')  
axs[1].set\_title("Per Capita Income-2011 inflation-adjusted dollars",fontsize=20)  
pubicHealth\_gpd.apply(lambda x: axs[1].annotate(text=x["Community Area Name"], xy=x.geometry.centroid.coords[0], ha='center'),axis=1)  
axs[1].tick\_params(axis='both', labelsize=15)  
cb\_ax\_1=axs[1].figure.axes[3]  
cb\_ax\_1.tick\_params(labelsize=15)  
  
plt.show()

area float64 area\_num\_1 object area\_numbe object comarea float64 comarea\_id float64 community object perimeter float64 shape\_area float64 shape\_len float64 geometry geometry dtype: object \_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_\_ boundaries\_community.area\_numbe dtype: int64



#### 2）公共健康数据的相关性分析

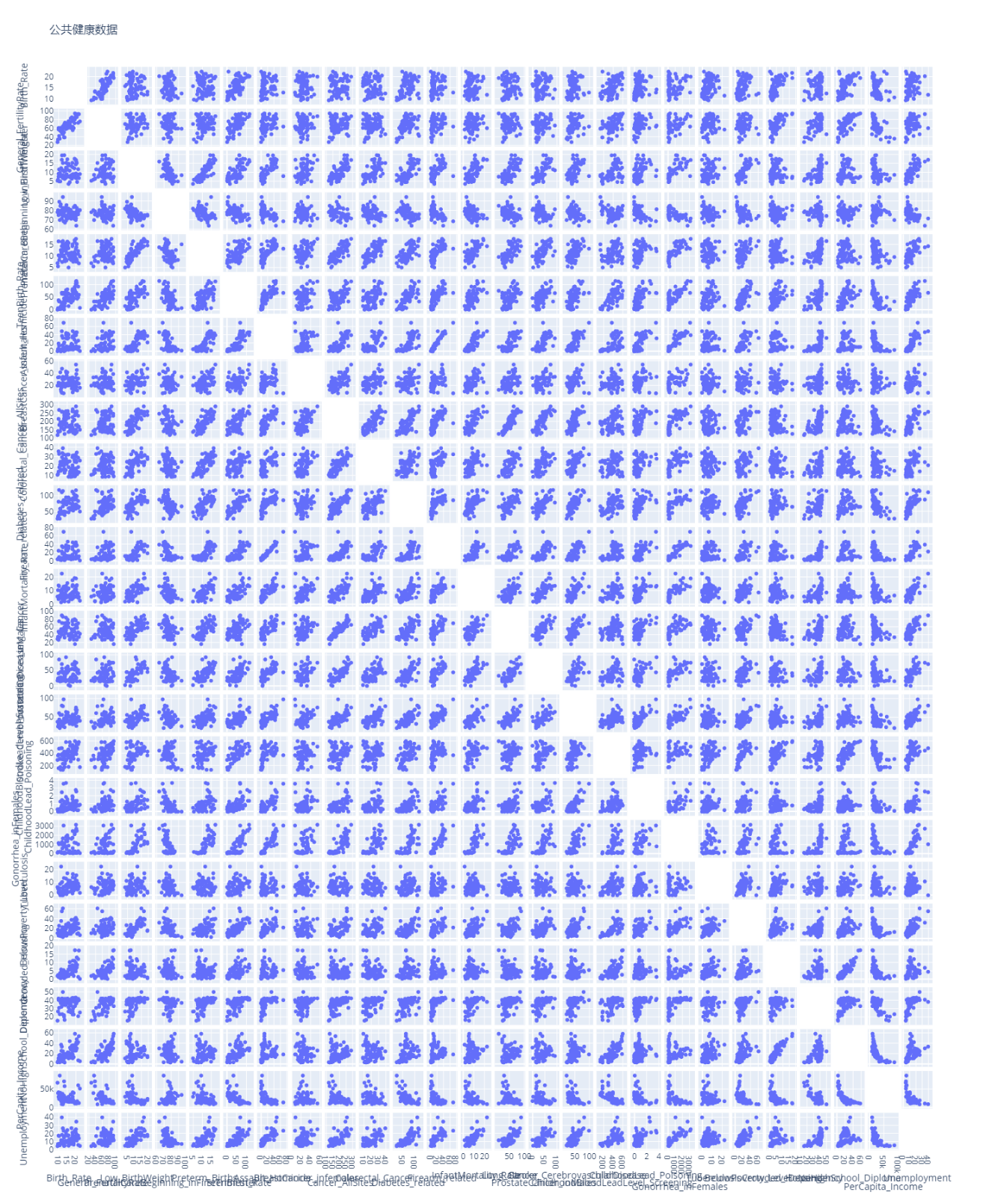
观察上图，似乎肺癌的发生比例与收入多少有关联，收入高的区域发生肺癌的机率低，相反，收入低的区域，发生肺癌的机率高。粗略的观察只能够给出一个初步不是十分确定的判断结果，如果需要以数据分析的方式证明之间是否存在关联，则需要作相关性分析。

pubicHealth\_Statistic\_mapping={'Community Area':'CommunityArea',   
 'Community Area Name':'CommunityArea\_Name',  
 'Birth Rate':'Birth\_Rate',  
 'General Fertility Rate':'General\_FertilityRate',  
 'Low Birth Weight':'Low\_BirthWeight',  
 'Prenatal Care Beginning in First Trimester':'PrenatalCareBeginning\_inFirstTrimester',   
 'Preterm Births':'Preterm\_Births',  
 'Teen Birth Rate':'TeenBirth\_Rate',  
 'Assault (Homicide)':'Assault\_Homicide',  
 'Breast cancer in females':'BreastCancer\_infemales',  
 'Cancer (All Sites)':'Cancer\_AllSites',   
 'Colorectal Cancer':'Colorectal\_Cancer',  
 'Diabetes-related':'Diabetes\_related',  
 'Firearm-related':'Firearm\_related',  
 'Infant Mortality Rate':'InfantMortality\_Rate',   
 'Lung Cancer':'Lung\_Cancer',  
 'Prostate Cancer in Males':'ProstateCancer\_inMales',  
 'Stroke (Cerebrovascular Disease)':'Stroke\_CerebrovascularDisease',  
 'Childhood Blood Lead Level Screening':'ChildhoodBloodLeadLevel\_Screening',  
 'Childhood Lead Poisoning':'ChildhoodLead\_Poisoning',  
 'Gonorrhea in Females':'Gonorrhea\_inFemales',   
 'Gonorrhea in Males':'Gonorrhea\_inMales',   
 'Tuberculosis':'Tuberculosis',  
 'Below Poverty Level':'BelowPoverty\_Level',   
 'Crowded Housing':'Crowded\_Housing',   
 'Dependency':'Dependency',  
 'No High School Diploma':'NoHighSchool\_Diploma',   
 'Per Capita Income':'PerCapita\_Income',  
 'Unemployment':'Unemployment',  
 }  
  
pubicHealth\_rename=pubicHealth\_gpd.rename(columns=pubicHealth\_Statistic\_mapping)  
  
pubicHealth\_extract\_columns=[  
 'Birth\_Rate','General\_FertilityRate', 'Low\_BirthWeight',  
 'PrenatalCareBeginning\_inFirstTrimester', 'Preterm\_Births',  
 'TeenBirth\_Rate', 'Assault\_Homicide', 'BreastCancer\_infemales',  
 'Cancer\_AllSites', 'Colorectal\_Cancer', 'Diabetes\_related',  
 'Firearm\_related', 'InfantMortality\_Rate', 'Lung\_Cancer',  
 'ProstateCancer\_inMales', 'Stroke\_CerebrovascularDisease',  
 'ChildhoodBloodLeadLevel\_Screening', 'ChildhoodLead\_Poisoning',  
 'Gonorrhea\_inFemales', 'Gonorrhea\_inMales', 'Tuberculosis',   
 'BelowPoverty\_Level', 'Crowded\_Housing', 'Dependency',  
 'NoHighSchool\_Diploma', 'PerCapita\_Income', 'Unemployment'   
 ]  
  
pubicHealth\_extract=pubicHealth\_rename[pubicHealth\_extract\_columns].select\_dtypes(include=np.number)  
publicHealth\_correlation=pubicHealth\_extract.corr()  
print\_html(publicHealth\_correlation)

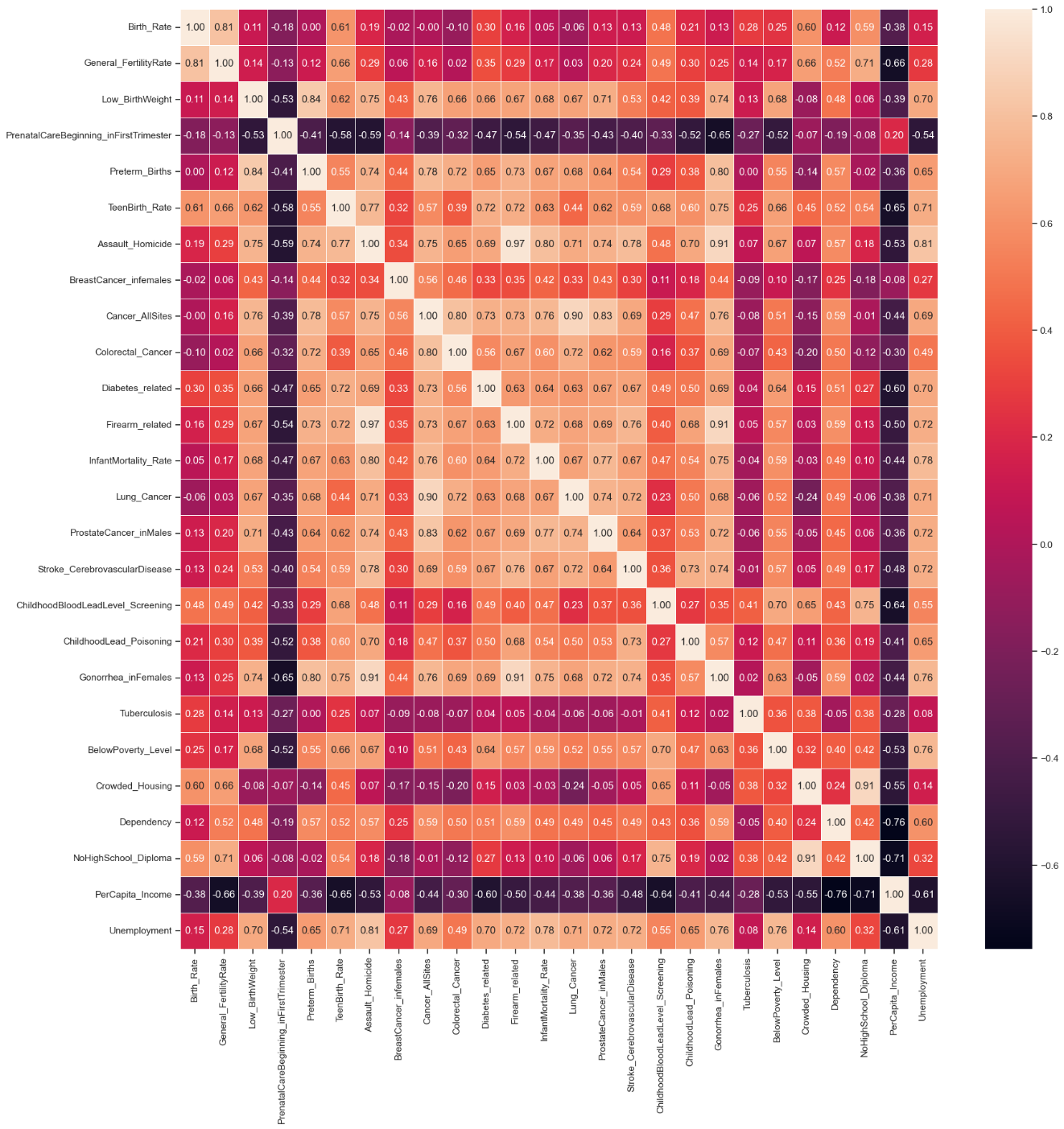
|  | Birth\_Rate | General\_FertilityRate | Low\_BirthWeight | PrenatalCareBeginning\_inFirstTrimester | Preterm\_Births | TeenBirth\_Rate | Assault\_Homicide | BreastCancer\_infemales | Cancer\_AllSites | **Colorectal\_Cancer** | **Diabetes\_related** | **Firearm\_related** | **InfantMortality\_Rate** | **Lung\_Cancer** | **ProstateCancer\_inMales** | **Stroke\_CerebrovascularDisease** | **ChildhoodBloodLeadLevel\_Screening** | **ChildhoodLead\_Poisoning** | **Gonorrhea\_inFemales** | **Tuberculosis** | **BelowPoverty\_Level** | **Crowded\_Housing** | **Dependency** | **NoHighSchool\_Diploma** | **PerCapita\_Income** | **Unemployment** |
| --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- | --- |
| Birth\_Rate | 1.000000 | 0.810334 | 0.108179 | -0.178847 | 0.004334 | 0.612710 | 0.188536 | -0.022201 | -0.001289 | -0.097507 | 0.303831 | 0.162722 | 0.051716 | -0.063567 | 0.130114 | 0.126423 | 0.481725 | 0.210016 | 0.126671 | 0.279755 | 0.249764 | 0.603219 | 0.115771 | 0.594249 | -0.377653 | 0.149439 |
| General\_FertilityRate | 0.810334 | 1.000000 | 0.142189 | -0.134292 | 0.122235 | 0.655528 | 0.293843 | 0.058089 | 0.164279 | 0.022977 | 0.352294 | 0.285562 | 0.165320 | 0.026357 | 0.196524 | 0.241311 | 0.490693 | 0.303624 | 0.250984 | 0.144972 | 0.173846 | 0.655826 | 0.524558 | 0.714008 | -0.661131 | 0.275589 |
| Low\_BirthWeight | 0.108179 | 0.142189 | 1.000000 | -0.532546 | 0.843100 | 0.622936 | 0.747493 | 0.425041 | 0.759791 | 0.657992 | 0.664107 | 0.671880 | 0.677792 | 0.671295 | 0.713754 | 0.525439 | 0.416265 | 0.388319 | 0.737727 | 0.125231 | 0.681049 | -0.080532 | 0.479581 | 0.056962 | -0.394564 | 0.702227 |
| PrenatalCareBeginning\_inFirstTrimester | -0.178847 | -0.134292 | -0.532546 | 1.000000 | -0.413383 | -0.579170 | -0.591605 | -0.142970 | -0.385923 | -0.315996 | -0.470152 | -0.540972 | -0.466319 | -0.353354 | -0.430259 | -0.395817 | -0.328470 | -0.516579 | -0.653985 | -0.270147 | -0.516145 | -0.074951 | -0.192405 | -0.078027 | 0.200858 | -0.542603 |
| Preterm\_Births | 0.004334 | 0.122235 | 0.843100 | -0.413383 | 1.000000 | 0.549843 | 0.742019 | 0.435020 | 0.780207 | 0.722153 | 0.650790 | 0.732584 | 0.672087 | 0.675119 | 0.642100 | 0.539608 | 0.289144 | 0.379146 | 0.800491 | 0.001197 | 0.550906 | -0.140244 | 0.571350 | -0.023655 | -0.364807 | 0.646550 |

通过上述相关性计算，已经计算了两两之间的相关系数。为了更加清晰的查看相关系数，可以同时打印散点图（通常在计算相关系数前打印散点图，查看两个变量间的线性关系），或查看相关系数变化的热力图。

import plotly.graph\_objects as go  
  
fig = go.Figure(data=go.Splom(  
 dimensions=[dict(label=k,values=list(v.values())) for k,v in pubicHealth\_extract.to\_dict().items()],  
 diagonal\_visible=False, # remove plots on diagonal  
 ))  
fig.update\_layout(  
 title='公共健康数据',  
 dragmode='select',  
 width=1800,  
 height=1800,  
 hovermode='closest',  
)  
  
fig.show()

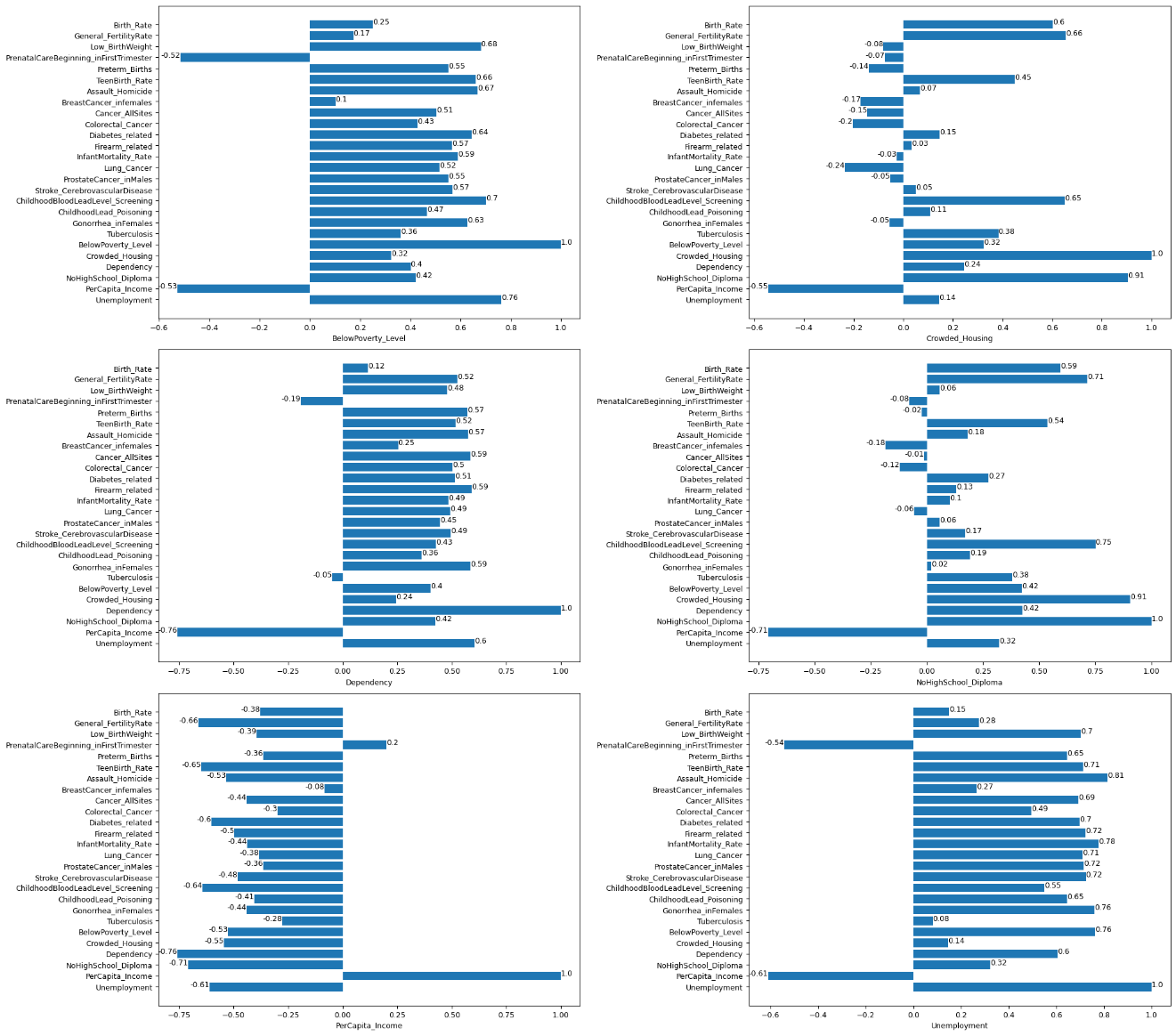


import seaborn as sns  
  
plt.figure(figsize=(20, 20))  
sns.heatmap(publicHealth\_correlation,annot=True, fmt=".2f", linewidths=.5,)  
plt.show()



不管是散点图还是热力图，以及相关系数矩阵，在观察多个两两之间相关性时，都不是很容易观察所关注的对象。因此可以根据分析的目的将需要关注的变量提取出来，只关注该变量与其它所有变量的相关系数。

import numpy as np  
  
economic\_factors=['BelowPoverty\_Level', 'Crowded\_Housing', 'Dependency', 'NoHighSchool\_Diploma', 'PerCapita\_Income', 'Unemployment']  
publicHealth\_indicator=publicHealth\_correlation[economic\_factors]  
publicHealth\_indicator\_columns=publicHealth\_indicator.T.columns.to\_numpy()  
  
plt.rcdefaults()  
plt.rcParams.update({'font.size':9})  
  
nrows=3  
ncols=2  
fig, axs=plt.subplots(nrows=nrows,ncols=ncols,figsize=(10\*2, 10\*2))   
y\_pos=np.arange(len(publicHealth\_indicator\_columns))  
  
i=0  
for idx in [(row,col) for row in range(nrows) for col in range(ncols)]:  
 axs[idx].barh(y\_pos, publicHealth\_indicator[economic\_factors[i]].to\_numpy(), align='center')   
 axs[idx].set\_yticks(y\_pos)  
 axs[idx].set\_yticklabels(publicHealth\_indicator\_columns)  
 axs[idx].invert\_yaxis() # labels read top-to-bottom  
 axs[idx].set\_xlabel(economic\_factors[i])  
 # axs.set\_title(title\_str)  
 for index, value in enumerate(publicHealth\_indicator[economic\_factors[i]].to\_numpy()):  
 if value>=0:  
 axs[idx].text(value, index, str(round(value,2)),horizontalalignment='left')  
 else:  
 axs[idx].text(value, index, str(round(value,2)),horizontalalignment='right')  
 i+=1  
   
plt.subplots\_adjust(top=0.92, bottom=0.08, left=0.10, right=0.95, hspace=0.1,wspace=0.4)  
plt.show()



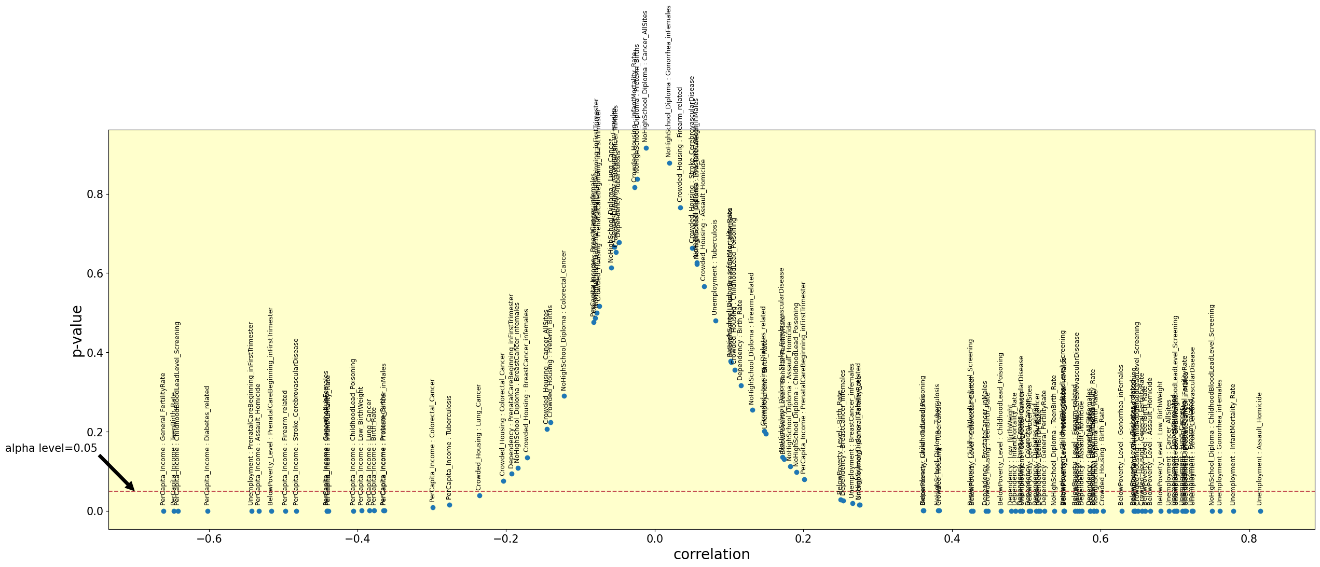
虽然计算了两两公共健康指标之间的相关系数，但这并不代表该样本所计算得出的相关性能够反映总体是否也存在样本所得出的现象，这个时候需要关注假设检验计算得出的P值，将P值不满足显著性水平为0.05的两两变量剔除掉，所得到的结果即为能够反映总体相关性的部分。首先用所关注的变量作为index索引，为所有的经济变量*economic\_factors=['BelowPoverty\_Level', 'Crowded\_Housing', 'Dependency', 'NoHighSchool\_Diploma', 'PerCapita\_Income', 'Unemployment']*，方便后续的数据分析。

from scipy import stats  
  
disease\_columns=[   
 'Birth\_Rate','General\_FertilityRate', 'Low\_BirthWeight',  
 'PrenatalCareBeginning\_inFirstTrimester', 'Preterm\_Births',  
 'TeenBirth\_Rate', 'Assault\_Homicide', 'BreastCancer\_infemales',  
 'Cancer\_AllSites', 'Colorectal\_Cancer', 'Diabetes\_related',  
 'Firearm\_related', 'InfantMortality\_Rate', 'Lung\_Cancer',  
 'ProstateCancer\_inMales', 'Stroke\_CerebrovascularDisease',  
 'ChildhoodBloodLeadLevel\_Screening', 'ChildhoodLead\_Poisoning',  
 'Gonorrhea\_inFemales', 'Tuberculosis',  
 ]  
pubicHealth\_pearsonr={}  
  
for factor in economic\_factors:  
 disease\_temp={}  
 for disease in disease\_columns:  
 desease\_series=pd.to\_numeric(pubicHealth\_extract[disease],errors='coerce')  
 factor\_series=pd.to\_numeric(pubicHealth\_extract[factor],errors='coerce')  
 mask=pd.notna(desease\_series)&pd.notna(factor\_series)  
   
 disease\_temp[disease]=stats.pearsonr(pd.to\_numeric(factor\_series[mask],errors='ignore'),pd.to\_numeric(desease\_series[mask],errors='ignore'))  
 pubicHealth\_pearsonr[factor]=disease\_temp  
pubicHealth\_pearsonr\_df=pd.DataFrame.from\_dict(pubicHealth\_pearsonr, orient='index').stack().to\_frame(name='corr\_pV')  
  
pubicHealth\_pearsonr\_df['correlation']=pubicHealth\_pearsonr\_df.corr\_pV.apply(lambda row:row[0])  
pubicHealth\_pearsonr\_df['p\_value']=pubicHealth\_pearsonr\_df.corr\_pV.apply(lambda row:row[1])  
  
print\_html(pubicHealth\_pearsonr\_df)

|  |  | corr\_pV | correlation | **p\_value** |
| --- | --- | --- | --- | --- |
| BelowPoverty\_Level | Birth\_Rate | (0.24976406537114698, 0.028475743860025216) | 0.249764 | 2.847574e-02 |
| General\_FertilityRate | (0.17384625982353086, 0.13051281178487212) | 0.173846 | 1.305128e-01 |
| Low\_BirthWeight | (0.6810485083790098, 9.384671983490509e-12) | 0.681049 | 9.384672e-12 |
| PrenatalCareBeginning\_inFirstTrimester | (-0.5161448655124206, 1.5503295893976403e-06) | -0.516145 | 1.550330e-06 |
| Preterm\_Births | (0.5509056684724956, 2.086303329322895e-07) | 0.550906 | 2.086303e-07 |

以相关系数的值为横坐标，以相关系数对应的P值为纵坐标，以0.05作为显著性水平，打印散点图，可以较为直观的查看哪些两两公共健康指标的相关系数能够反映或者不能够反映总体的现象。

pubicHealth\_pearsonr\_resetIdx=pubicHealth\_pearsonr\_df.reset\_index()  
significance\_level=0.05  
fig=plt.figure(figsize=(30, 10))  
ax=fig.add\_subplot(111, facecolor='#FFFFCC')  
X=pubicHealth\_pearsonr\_resetIdx.correlation  
Y=pubicHealth\_pearsonr\_resetIdx.p\_value  
ax.plot(X, Y, 'o')  
ax.axhline(y=significance\_level,color='r',linestyle='--',label='significance\_level')  
i=0  
for x,y in zip (X,Y):  
 label=pubicHealth\_pearsonr\_resetIdx.loc[i].level\_0+' : '+pubicHealth\_pearsonr\_resetIdx.loc[i].level\_1  
 plt.annotate(label, # this is the text  
 (x,y), # this is the point to label  
 textcoords="offset points", # how to position the text  
 xytext=(0,10), # distance from text to points (x,y)  
 ha='center', # horizontal alignment can be left, right or center  
 rotation=90  
 )  
   
 i+=1  
plt.annotate('alpha level=%.2f'%significance\_level,xy=(-0.7,significance\_level),xytext=(-0.75, 0.15),arrowprops=dict(facecolor='black', shrink=0.01),horizontalalignment='right',size=15)  
plt.xlabel("correlation",fontsize=20)  
plt.ylabel("p-value",fontsize=20)   
plt.tick\_params(axis='both', labelsize=15)  
plt.show()

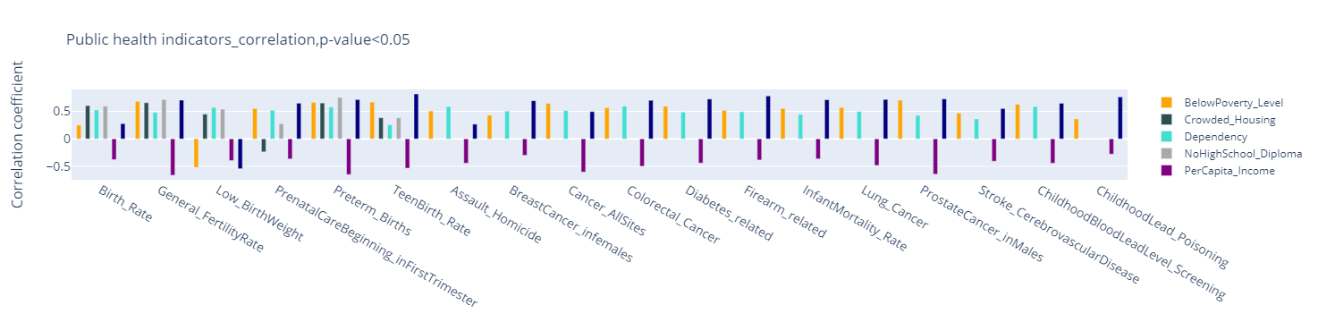


提取P值小于等于0.05的所有行，并以柱状图的形式打印相关系数结果。

pubicHealth\_pearsonr\_alpha=pubicHealth\_pearsonr\_df[pubicHealth\_pearsonr\_df.p\_value<=0.05]  
print\_html(pubicHealth\_pearsonr\_alpha)

|  |  | corr\_pV | correlation | **p\_value** |
| --- | --- | --- | --- | --- |
| BelowPoverty\_Level | Birth\_Rate | (0.24976406537114698, 0.028475743860025216) | 0.249764 | 2.847574e-02 |
| Low\_BirthWeight | (0.6810485083790098, 9.384671983490509e-12) | 0.681049 | 9.384672e-12 |
| PrenatalCareBeginning\_inFirstTrimester | (-0.5161448655124206, 1.5503295893976403e-06) | -0.516145 | 1.550330e-06 |
| Preterm\_Births | (0.5509056684724956, 2.086303329322895e-07) | 0.550906 | 2.086303e-07 |
| TeenBirth\_Rate | (0.6600382763346119, 6.598083934767818e-11) | 0.660038 | 6.598084e-11 |

import plotly.graph\_objects as go  
import random  
  
def generate\_colors():   
 '''  
 function - 生成颜色列表或者字典  
   
 Returns:  
 hex\_colors\_only - 16进制颜色值列表；list  
 hex\_colors\_dic - 颜色名称：16进制颜色值；dict  
 rgb\_colors\_dic - 颜色名称：(r,g,b)；dict  
 '''  
 import matplotlib  
   
 hex\_colors\_dic={}  
 rgb\_colors\_dic={}  
 hex\_colors\_only=[]  
 for name, hex in matplotlib.colors.cnames.items():  
 hex\_colors\_only.append(hex)  
 hex\_colors\_dic[name]=hex  
 rgb\_colors\_dic[name]=matplotlib.colors.to\_rgb(hex)  
 return hex\_colors\_only,hex\_colors\_dic,rgb\_colors\_dic  
  
generate\_colors,\_,\_=generate\_colors()  
  
economic\_factors=['BelowPoverty\_Level', 'Crowded\_Housing', 'Dependency', 'NoHighSchool\_Diploma', 'PerCapita\_Income', 'Unemployment']  
fig = go.Figure()  
  
for idx,data in pubicHealth\_pearsonr\_alpha.groupby(level=0):  
 fig.add\_trace(go.Bar(x=disease\_columns,  
 y=data.correlation,  
 name=idx,  
 marker\_color=random.choice(generate\_colors)  
 ))  
  
fig.update\_layout(  
 title='Public health indicators\_correlation,p-value<0.05',  
 xaxis\_tickfont\_size=14,  
 yaxis=dict(  
 title='correlation)',  
 titlefont\_size=16,  
 tickfont\_size=14,  
 ),  
 barmode='group',  
 bargap=0.1, # gap between bars of adjacent location coordinates.  
 bargroupgap=0.5 # gap between bars of the same location coordinate.  
)  
fig.show()



注释（Notes）：

① scipy.stats.t，a Student’s T continuous random variable.（<https://docs.scipy.org/doc/scipy-0.13.0/reference/generated/scipy.stats.t.html>）。

② Chicago Data Portal，CDP，为芝加哥城开放数据门户，可免费下载数据用于相关分析，其中许多数据集每天至少更新一次或数次（<https://data.cityofchicago.org/>）。

参考文献（References）:

[1] Timothy C.Urdan(蒂莫西•C•厄丹),彭志文译.白话统计学[M].中国人民大学出版社.2013,12.第3版. <Timothy C. Urdan.Statistics in plain English[M]. New York: Routledge, 2010>.

[2] (日)高桥 信著,陈刚译.株式会社TREND-PRO漫画制作.漫画统计学[M].科学出版社.北京,2019.8.

[3] Dawn Griffiths.Head First Statistics: A Brain-Friendly Guide[M]. Sebastopol:O’Reilly Media.September, 2008.