seaborn可视化(3) ——分布

类型	函数	说明
单变量	**distplot(data)**	灵活地绘制观测值的单变量分布
	kdeplot(x,y,data)	拟合并绘制单变量或双变量核密度估计值
	rugplot(x,y,data)	将数组中的数据点绘制为轴上的标尺
二元	**jointplot(x,y,data)**	用双变量图和单变量图绘制一个由两个变量组成的图
成对	**pairplot(data)**	将点估计和置信区间显示为矩形条

全文： >

distplot() :http://seaborn.pydata.org/generated/seaborn.distplot.html#seaborn.distplot>

kdeplot(): >

rugplot(): >

jointplot(): >

pairplot()： >

单变量分布

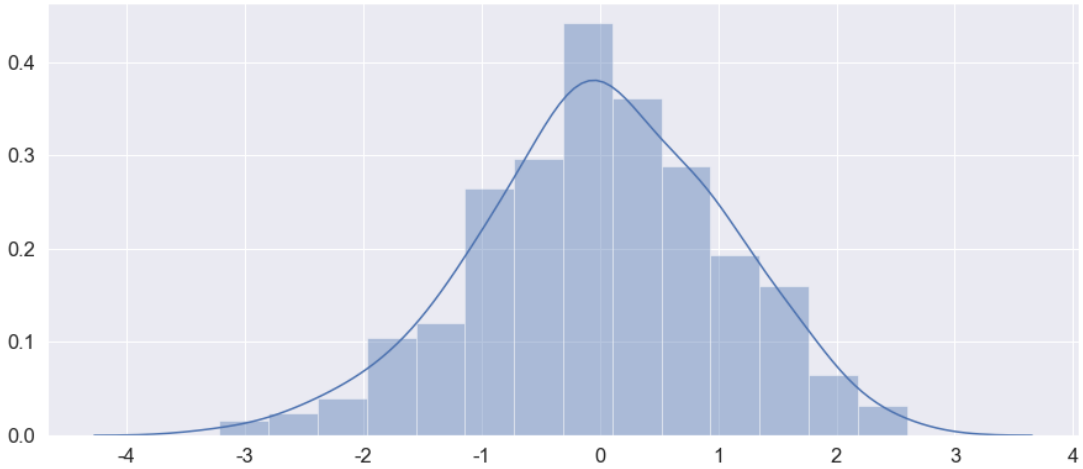
- **catplot() or warmplot() : 散点图可视化分类变量**

```
"""
要快速查看seaborn中的单变量分布，最方便的方法是distplot()函数。
默认情况下，这将绘制直方图并符合内核密度估计(KDE)
Signature:
sns.distplot(
    ['a', 'bins=None', 'hist=True', 'kde=True', 'rug=False', 'fit=None',
    'hist_kws=None', 'kde_kws=None', 'rug_kws=None', 'fit_kws=None',
    'color=None', 'vertical=False', 'norm_hist=False', 'axlabel=None',
    'label=None', 'ax=None'],)
"""
import numpy as np
import pandas as pd
from scipy import stats
import matplotlib.pyplot as plt
import seaborn as sns
sns.set(style="ticks", color_codes=True)
```

```
plt.rcParams['font.sans-serif'] = ['SimHei']
plt.rcParams['axes.unicode_minus'] = False
sns.set(font = 'SimHei')
sns.set(font_scale=1.5)
```

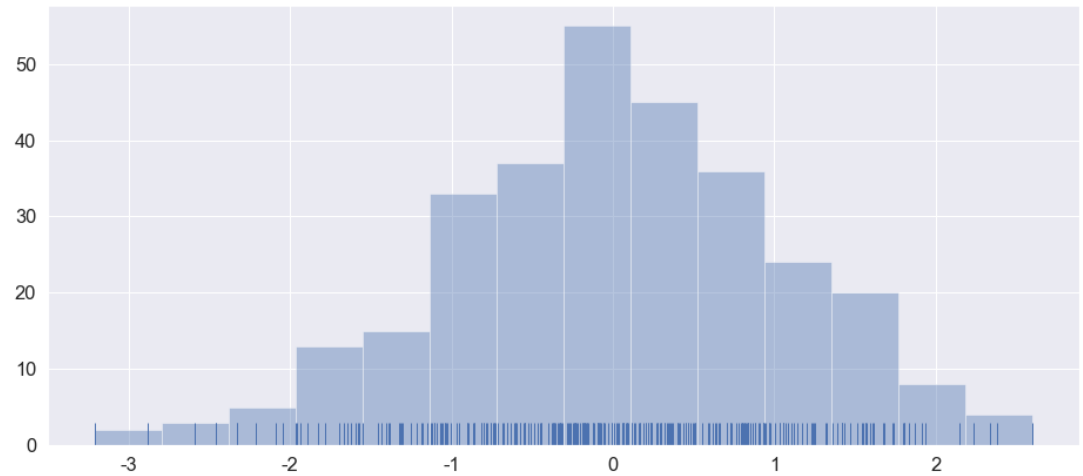
1:最基本的情况:

```
x = np.random.normal(size=300)
sns.distplot(x)
```



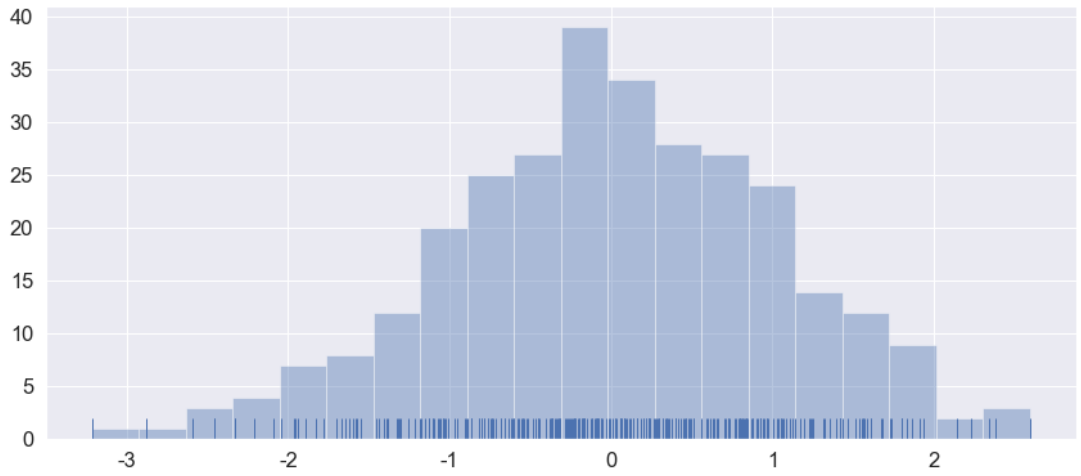
2. rugplot()

```
#rugplot():将数组中的数据点绘制为轴上的标尺
sns.distplot(x,kde = False,rug = True)
```



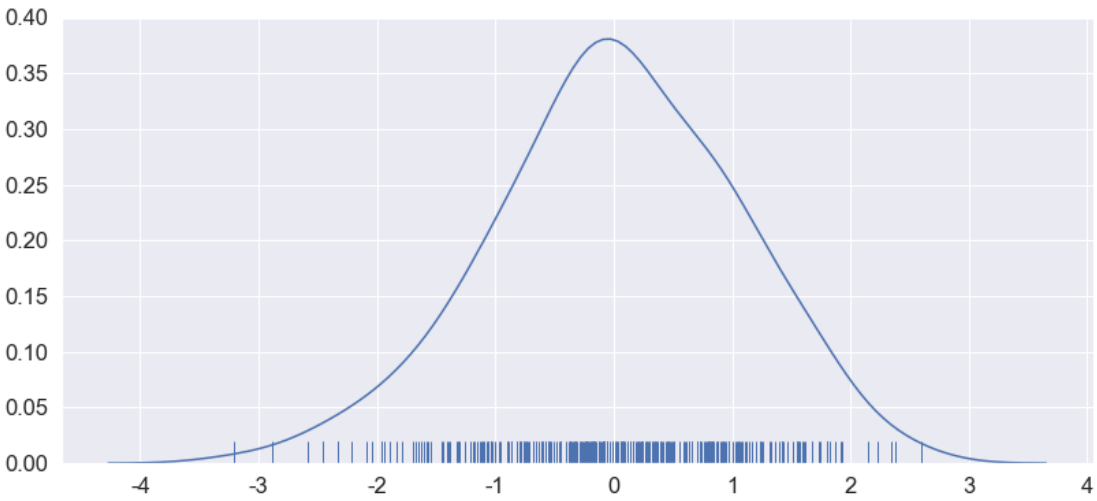
3. bins

```
#bins
sns.distplot(x, bins=20, kde=False, rug=True)
```



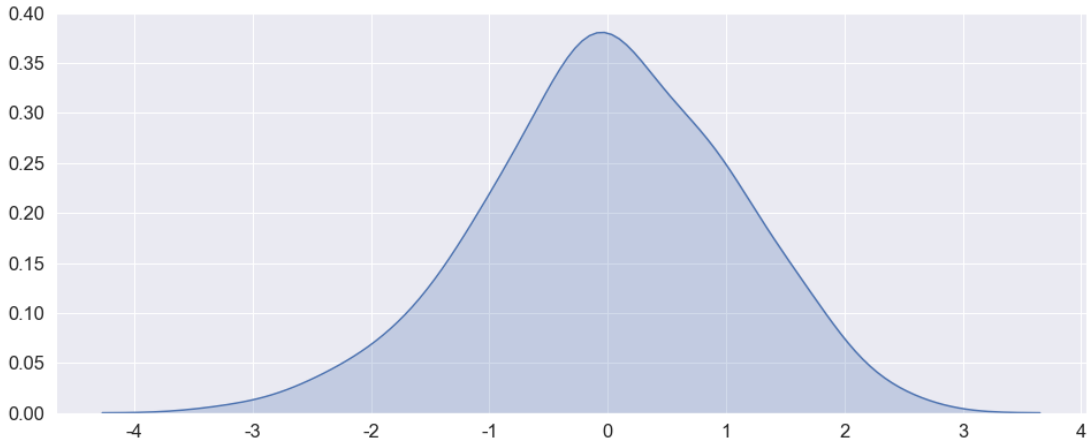
4. hist

```
#hist 是否绘制(赋范)直方图
sns.distplot(x, hist=False, rug=True)
```



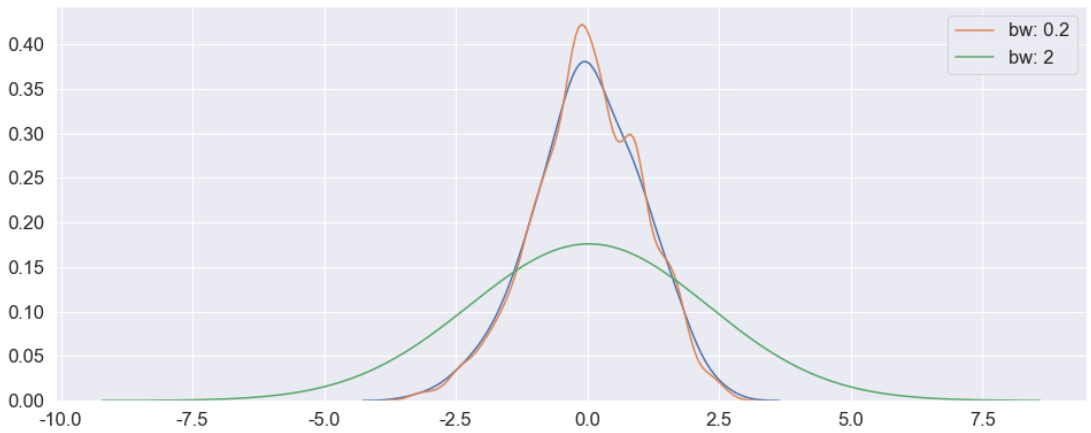
5.kdeplot()

```
"""
Signature:
sns.kdeplot(
    ['data', 'data2=None', 'shade=False', 'vertical=False',"kernel='gau'",
    "bw='scott'", 'gridsize=100', 'cut=3', 'clip=None', 'legend=True',
    'cumulative=False', 'shade_lowest=True', 'cbar=False',
    'cbar_ax=None', 'cbar_kws=None', 'ax=None', '**kwargs'],)
说明：拟合并绘制单变量或双变量核密度估计值
"""
#kdeplot() shade
sns.kdeplot(x, shade=True)
```



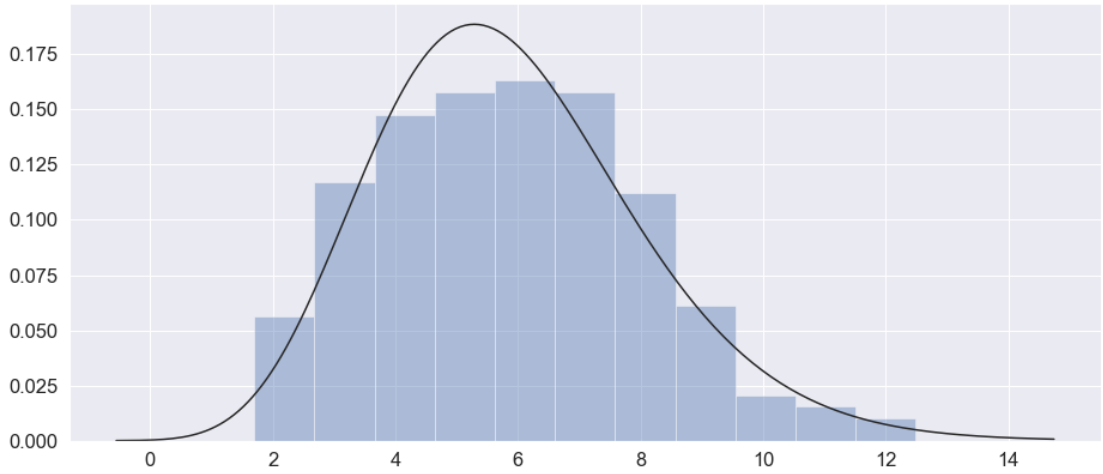
6.bandwidth:带宽

```
#bw
sns.kdeplot(x)
sns.kdeplot(x, bw=.2, label="bw: 0.2")
sns.kdeplot(x, bw=2, label="bw: 2")
plt.legend()
```



7.拟合参数分布

```
#拟合参数分布
x = np.random.gamma(6, size=200)
sns.distplot(x, kde=False, fit=stats.gamma)
```



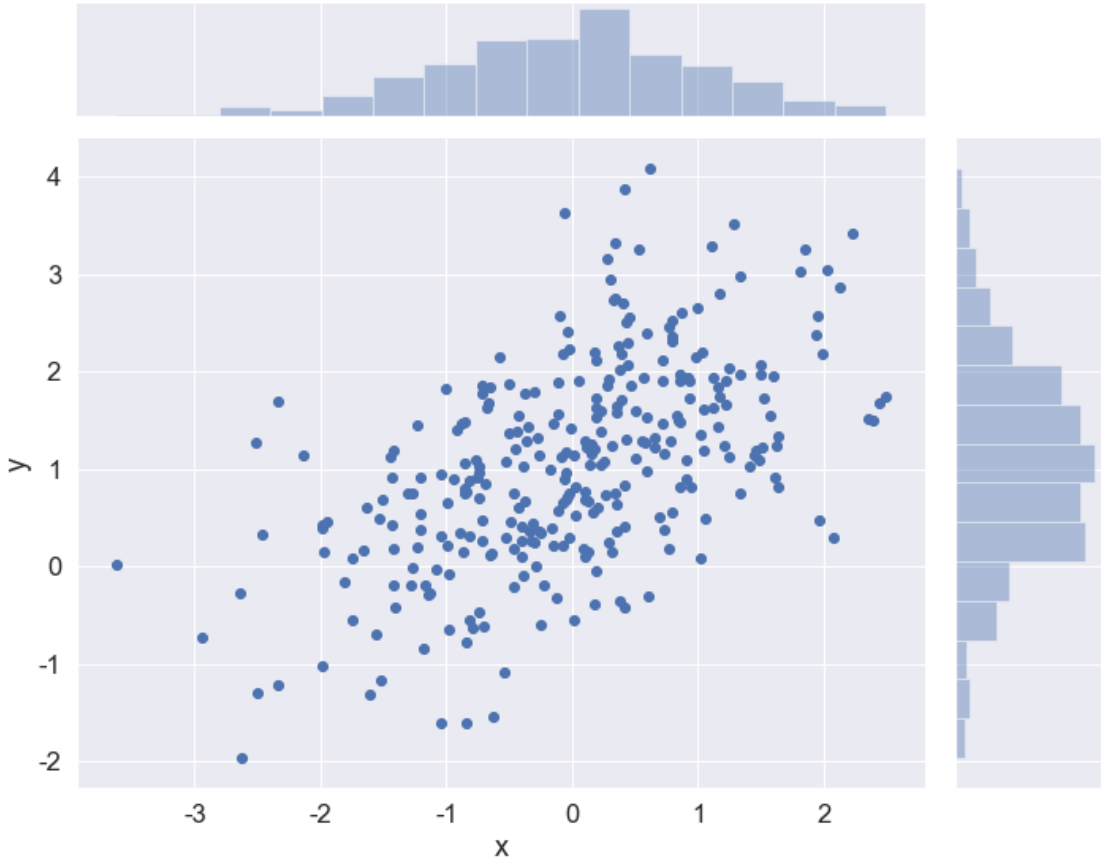
二元分布

- **jointplot() : 用双变量图和单变量图绘制一个由两个变量组成的图**

```
"""
用双变量图和单变量图绘制一个由两个变量组成的图
Signature:
sns.jointplot(
    ['x', 'y', 'data=None', "kind='scatter'", 'stat_func=None', 'color=None',
    'height=6', 'ratio=5', 'space=0.2', 'dropna=True', 'xlim=None',
    'ylim=None', 'joint_kws=None', 'marginal_kws=None', 'annot_kws=None',
    '**kwargs'],)
"""
```

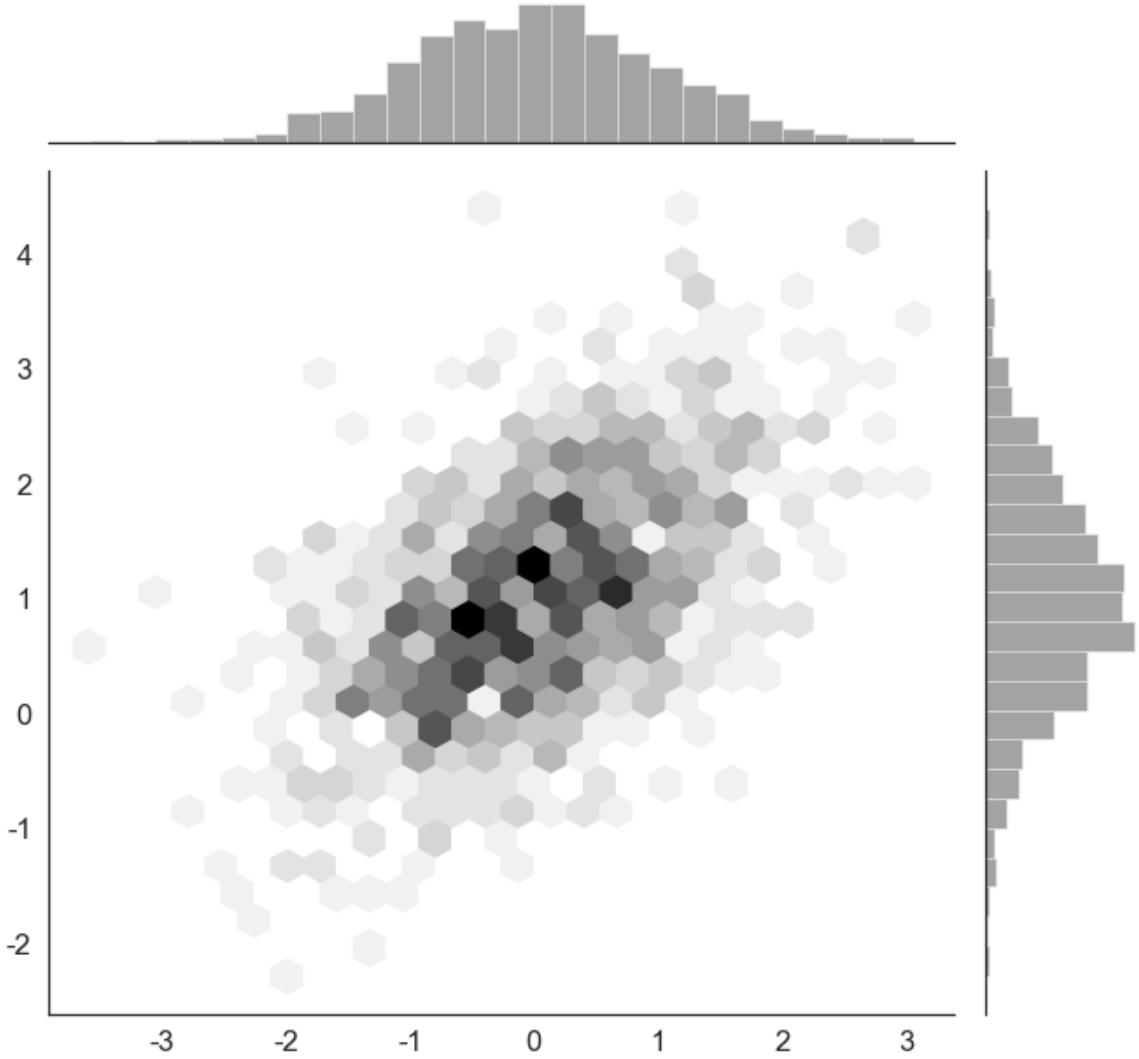
1.jointplot()

```
#散点图
mean, cov = [0, 1], [(1, .5), (.5, 1)]
data = np.random.multivariate_normal(mean, cov, 300)
df = pd.DataFrame(data, columns=["x", "y"])
sns.jointplot(x = "x",y = "y",data = df)
```



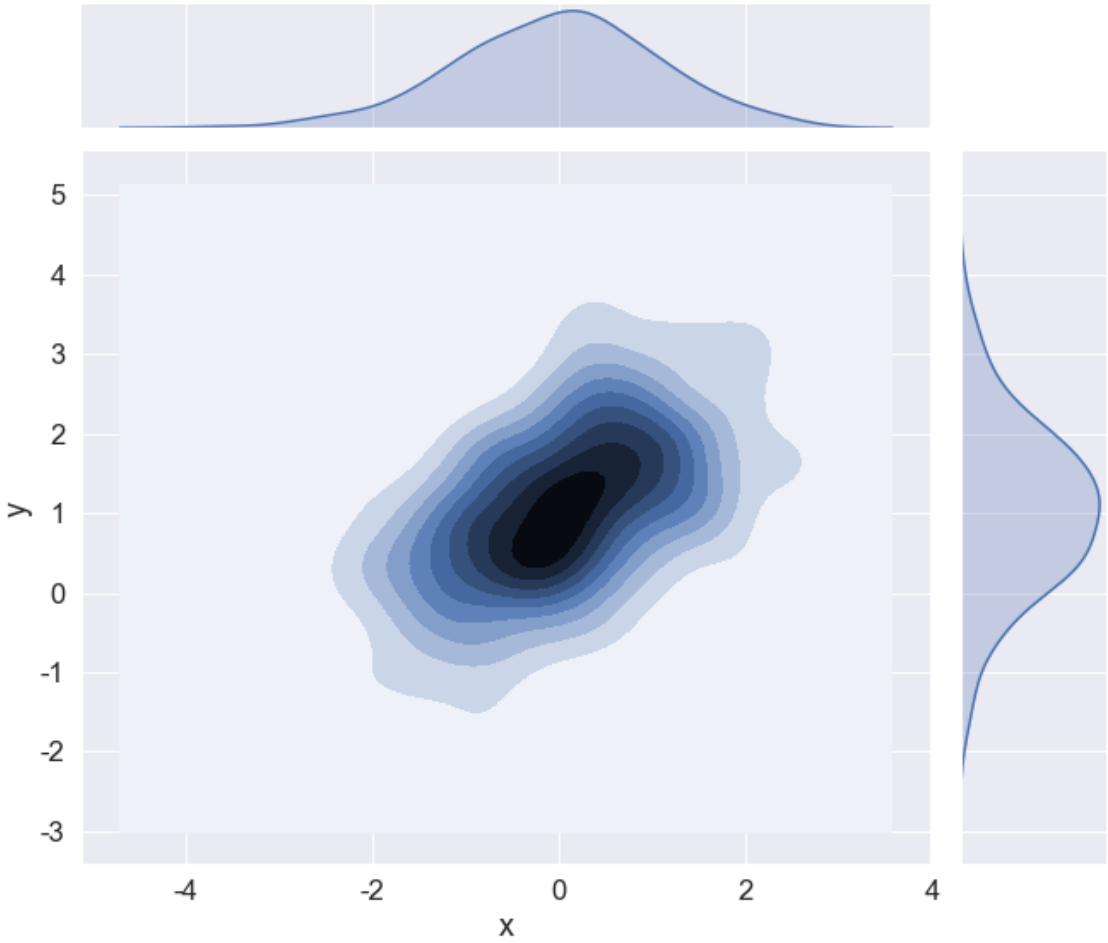
2. hexbin

```
#多边形图
x, y = np.random.multivariate_normal(mean, cov, 1000).T
with sns.axes_style("white"):
    sns.jointplot(x=x, y=y, kind="hex", color="k")
```



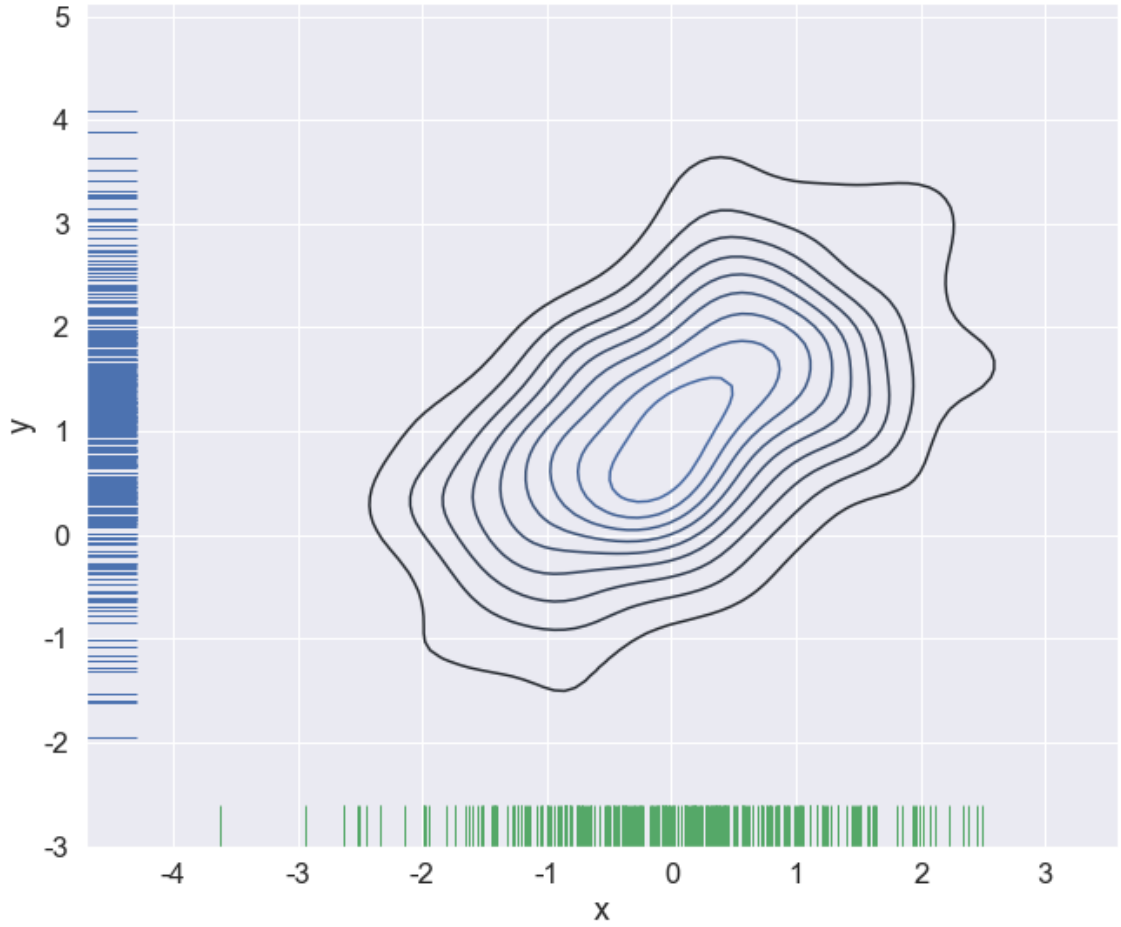
3.kde

```
#kde
sns.jointplot(x="x", y="y", data=df, kind="kde")
```



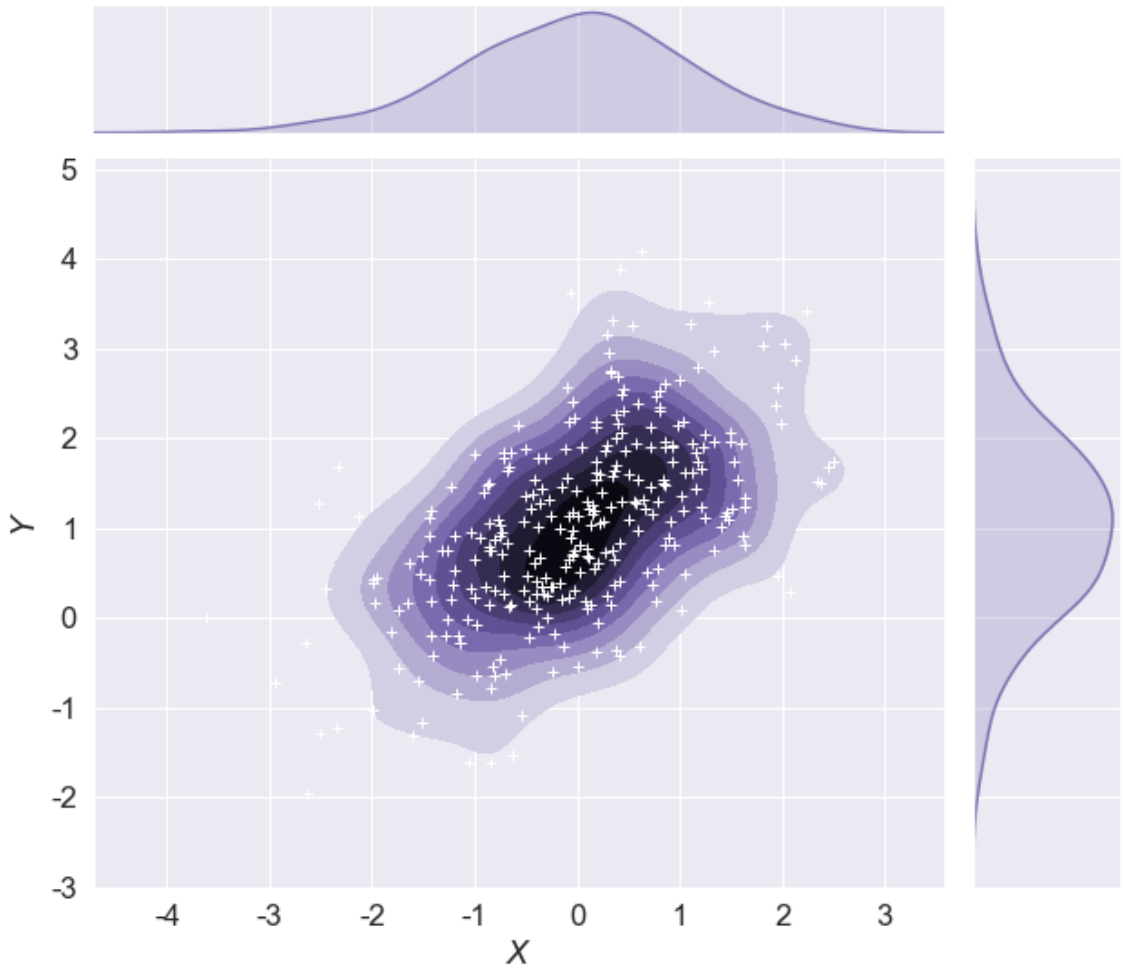
4.kde+rug

```
#kde+rug
f, ax = plt.subplots(figsize=(6, 6))
sns.kdeplot(df.x, df.y, ax=ax)
sns.rugplot(df.x, color="g", ax=ax)
sns.rugplot(df.y, vertical=True, ax=ax)"""
一个相关的函数boxenplot()绘制了一个类似于箱形图的图，但是经过了优化，
可以显示关于分布形状的更多信息。它最适合较大的数据集：
"""
diamonds = sns.load_dataset("diamonds")
sns.catplot(x="color", y="price", kind="boxen",
            data=diamonds.sort_values("color"))
```



5.多层

```
#多层次
g = sns.jointplot(x="x", y="y", data=df, kind="kde", color="m")
g.plot_joint(plt.scatter, c="w", s=30, linewidth=1, marker="+")
g.ax_joint.collections[0].set_alpha(0)
g.set_axis_labels("$X$", "$Y$")
```

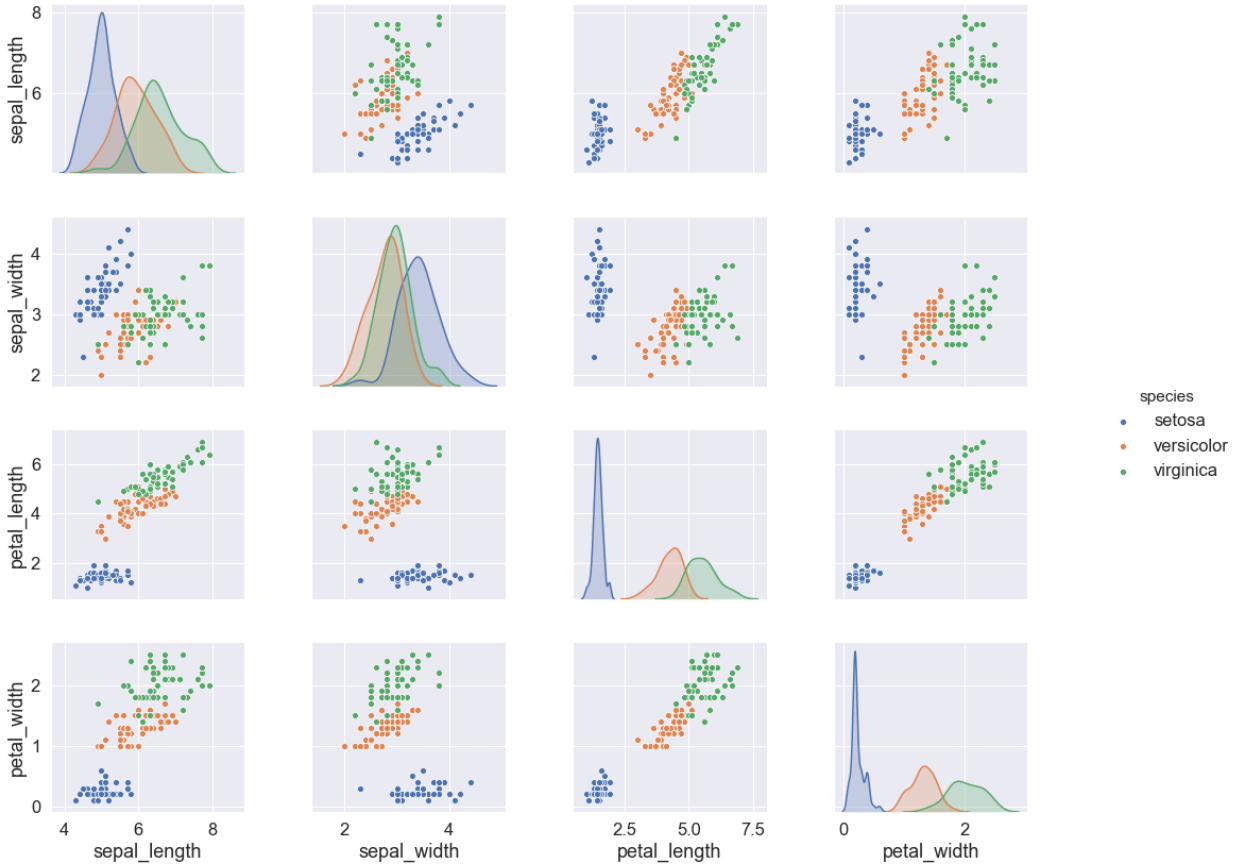


成对关系

1.pairplot()

```
"""
要在数据集中绘制多个成对的二元分布，可以使用pairplot()函数。
这将创建一个轴矩阵，并显示数据aframe中每对列的关系。
默认情况下，它还绘制了每个变量在对角线轴上的单变量分布:
pairplot() :在数据集中绘制成对关系
"""
Signature:
sns.pairplot(
    ['data', 'hue=None', 'hue_order=None', 'palette=None', 'vars=None',
    'x_vars=None', 'y_vars=None', "kind='scatter'", "diag_kind='auto'",
    'markers=None', 'height=2.5', 'aspect=1', 'dropna=True', 'plot_kws=None',
    'diag_kws=None', 'grid_kws=None', 'size=None'],)

iris = sns.load_dataset("iris")
sns.pairplot(iris,hue = 'species')
```


sepal_length
sepal_width
petal_length
petal_width
4
6
8
2
3
4
2
4
6
0
1
2
2.5
5.0
7.5
0
2
species
setosa
versicolor
virginica