Seaborn

March 21, 2019

Seaborn Tutorial

```
In [6]: import pandas as pd

In [7]: from matplotlib import pyplot as plt

In [15]: import seaborn as sb

In [17]: sb.get_dataset_names()

C:\Users\by3001pm\AppData\Local\Continuum\anaconda3\lib\site-packages\bs4\__init__.py:181: Use

The code that caused this warning is on line 193 of the file C:\Users\by3001pm\AppData\Local\C

 BeautifulSoup(YOUR_MARKUP})

to this:

 BeautifulSoup(YOUR_MARKUP, "lxml")

  markup_type=markup_type))


Out[17]: ['anscombe',
          'attention',
          'brain_networks',
          'car_crashes',
          'diamonds',
          'dots',
          'exercise',
          'flights',
          'fmri',
          'gammas',
          'iris',
          'mpg',
          'planets',
          'tips',
          'titanic']
```

In [11]:
```
df = sb.load_dataset('tips')
```

In [12]:
```
df.head()
```

Out[12]:

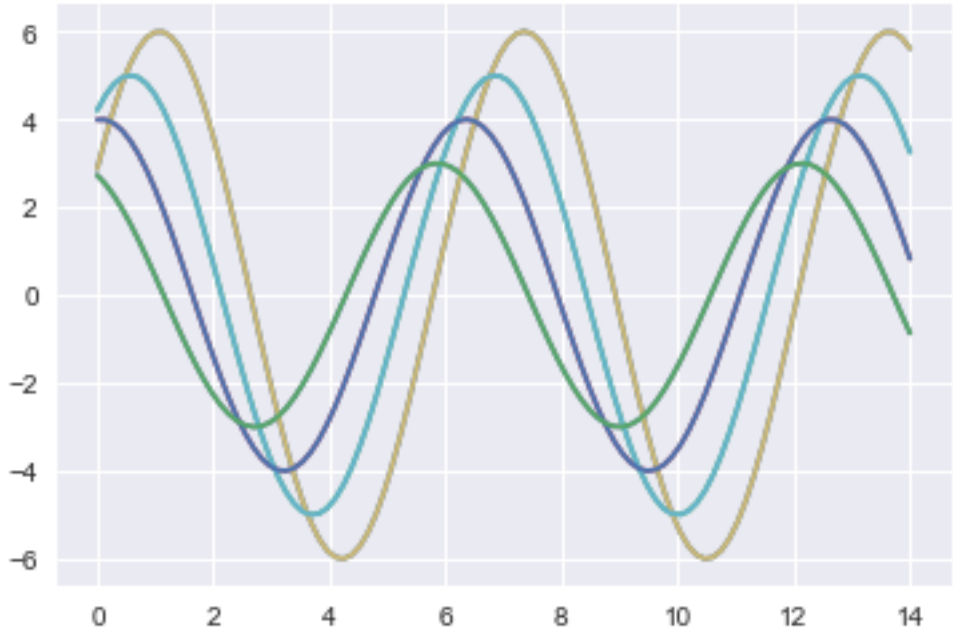
	total_bill	tip	sex	smoker	day	time	size
0	16.99	1.01	Female	No	Sun	Dinner	2
1	10.34	1.66	Male	No	Sun	Dinner	3
2	21.01	3.50	Male	No	Sun	Dinner	3
3	23.68	3.31	Male	No	Sun	Dinner	2
4	24.59	3.61	Female	No	Sun	Dinner	4

In [13]:
```
df.describe()
```

Out[13]:

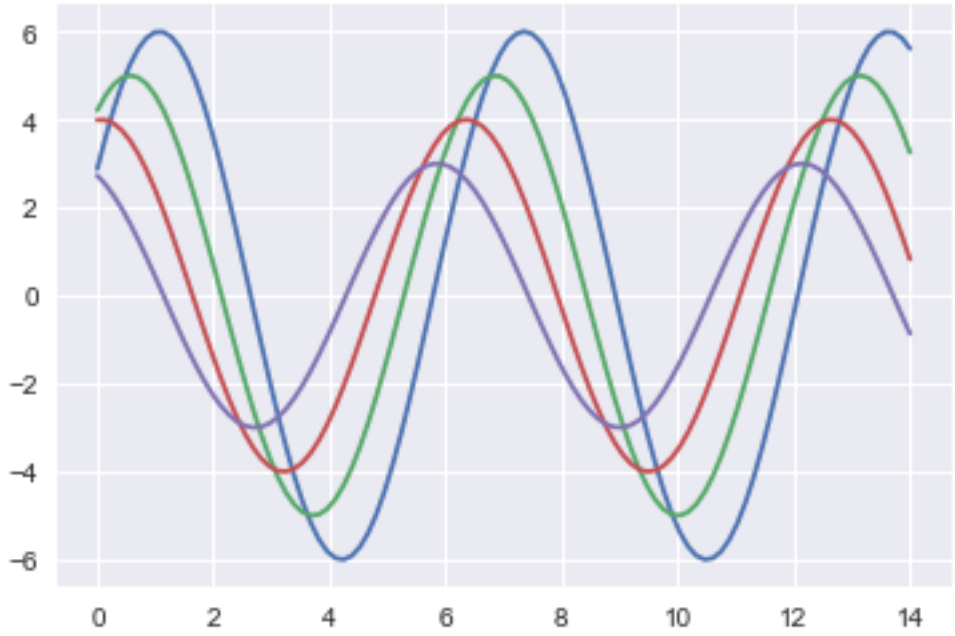
	total_bill	tip	size
count	244.000000	244.000000	244.000000
mean	19.785943	2.998279	2.569672
std	8.902412	1.383638	0.951100
min	3.070000	1.000000	1.000000
25%	13.347500	2.000000	2.000000
50%	17.795000	2.900000	2.000000
75%	24.127500	3.562500	3.000000
max	50.810000	10.000000	6.000000

In [20]:
```
import numpy as np
from matplotlib import pyplot as plt
def sinplot(flip = 1):
    x = np.linspace(0, 14, 100)
    for i in range(1, 5):
        plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)
sinplot()
plt.show()
```



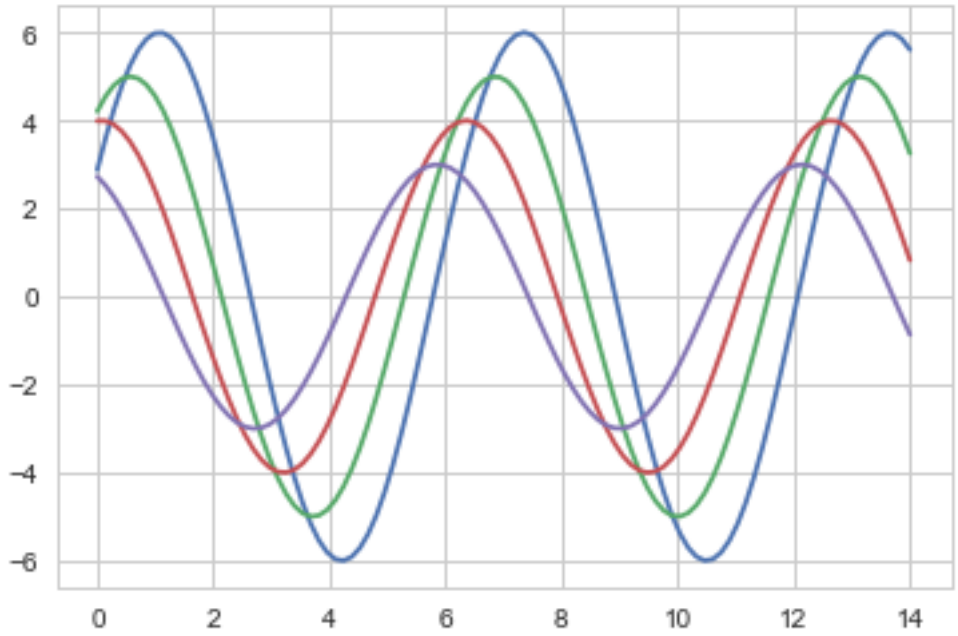
You can change the above plot to Seaborn with set() function

```
In [18]: import numpy as np
         from matplotlib import pyplot as plt
         def sinplot(flip = 1):
            x = np.linspace(0, 14, 100)
            for i in range(1, 5):
               plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)
         import seaborn as sb
         sb.set()
         sinplot()
         plt.show()
```



Using the set_style() function, you can set the theme of the plot. The default theme is darkgrid. The other themes are whitegrid, dark, white, and ticks. Let's change the theme to whitegraid.

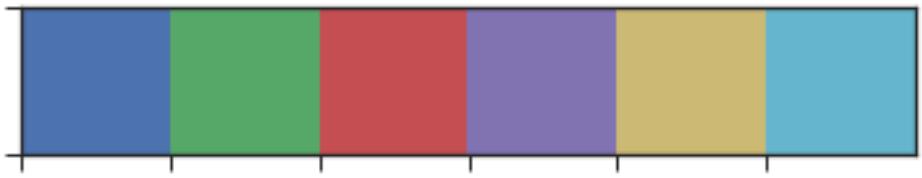
```
In [21]: import numpy as np
         from matplotlib import pyplot as plt
         def sinplot(flip=1):
            x = np.linspace(0, 14, 100)
            for i in range(1, 5):
               plt.plot(x, np.sin(x + i * .5) * (7 - i) * flip)
         import seaborn as sb
         sb.set_style("whitegrid")
         sinplot()
         plt.show()
```



In Seaborn, you can use color_palette(), to give colors and aestatics to a plot. For example: seaborn.color_palette(palette = None, n_colors = None, desat = None. n_colors: Number of colors in the palette. Default is 6 colors. desat: desatuartion gradient

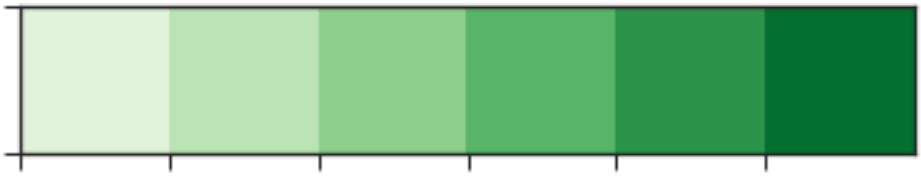
seaborn.palplot() can also be used for color palettes.

```
In [23]: from matplotlib import pyplot as plt
         import seaborn as sb
         current_palette = sb.color_palette()
         sb.palplot(current_palette)
         plt.show()
```



Shades of green.

```
In [24]: from matplotlib import pyplot as plt
         import seaborn as sb
         current_palette = sb.color_palette()
         sb.palplot(sb.color_palette("Greens"))
         plt.show()
```



The following is the histogram of the Iris built-in dataset using Seaborn.

```
In [44]: df = sb.load_dataset('iris')

In [46]: df.head()
```

Out[46]:

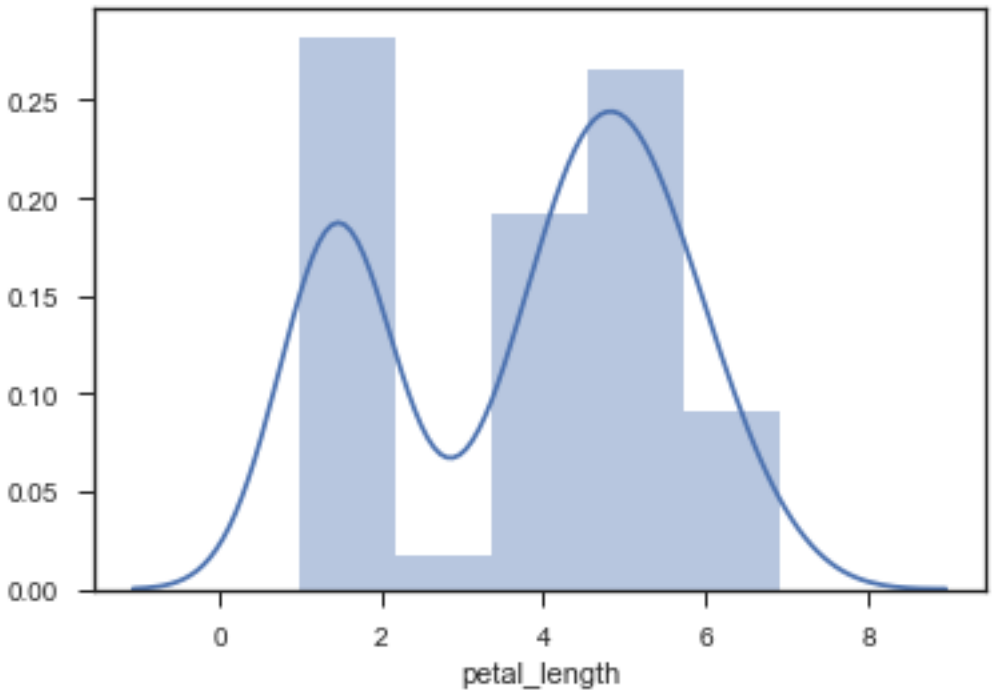
	sepal_length	sepal_width	petal_length	petal_width	species
0	5.1	3.5	1.4	0.2	setosa
1	4.9	3.0	1.4	0.2	setosa
2	4.7	3.2	1.3	0.2	setosa
3	4.6	3.1	1.5	0.2	setosa
4	5.0	3.6	1.4	0.2	setosa

```
In [45]: df.describe()
```

Out[45]:

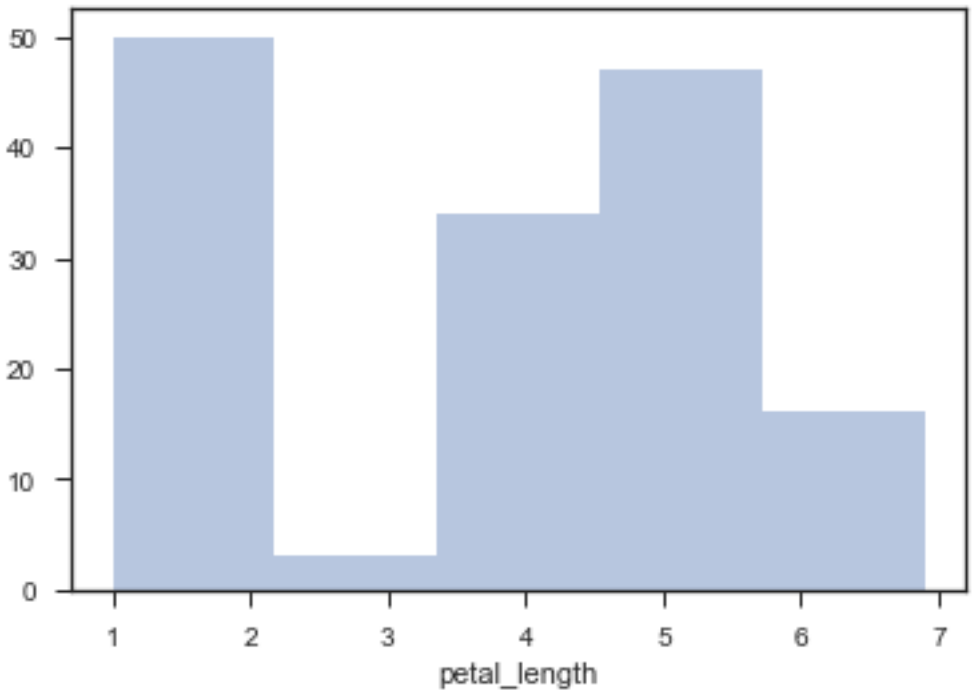
	sepal_length	sepal_width	petal_length	petal_width
count	150.000000	150.000000	150.000000	150.000000
mean	5.843333	3.057333	3.758000	1.199333
std	0.828066	0.435866	1.765298	0.762238
min	4.300000	2.000000	1.000000	0.100000
25%	5.100000	2.800000	1.600000	0.300000
50%	5.800000	3.000000	4.350000	1.300000
75%	6.400000	3.300000	5.100000	1.800000
max	7.900000	4.400000	6.900000	2.500000

```
In [26]: sb.distplot(df['petal_length'],kde = True)
         plt.show()
```



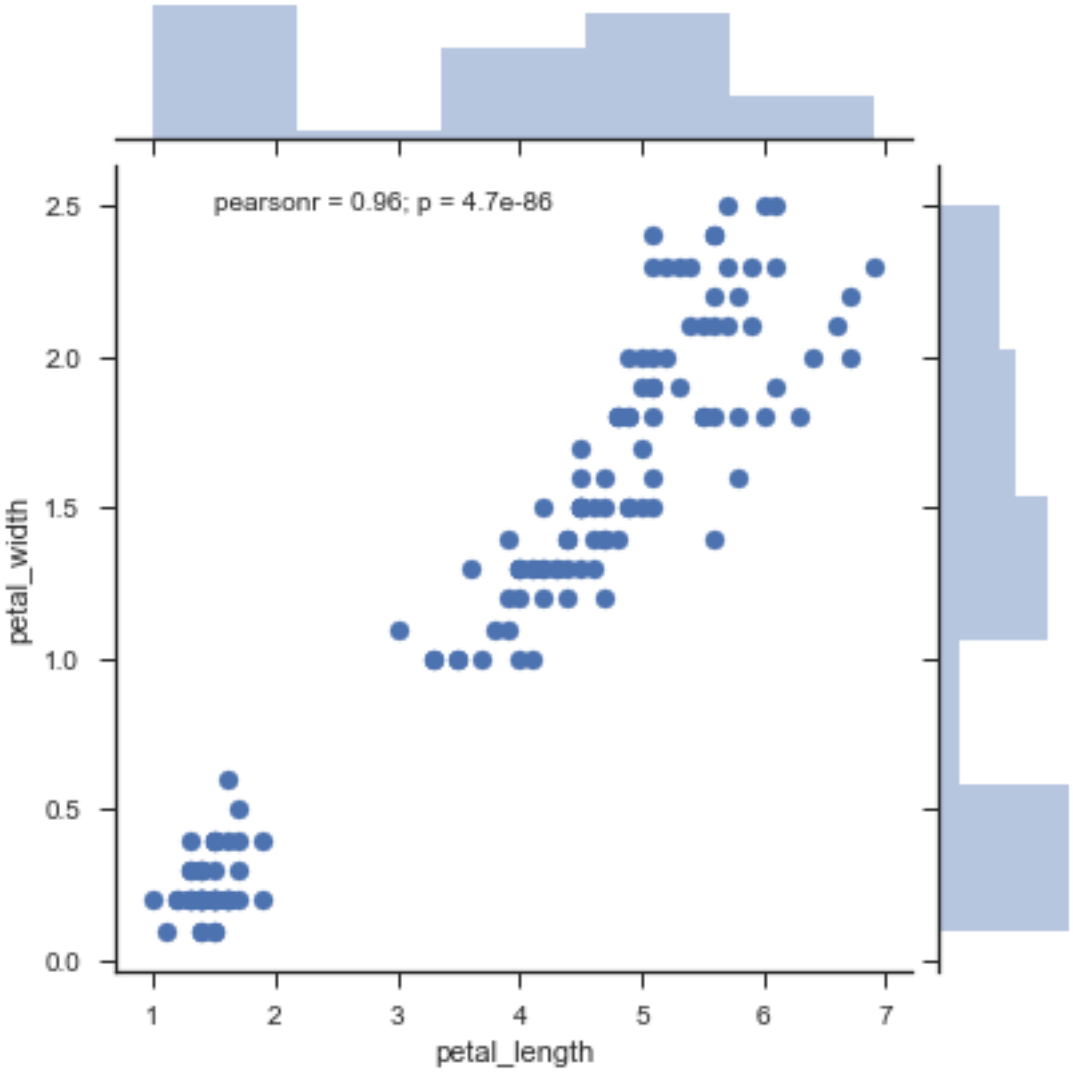
You genrally don't need to set kde to true. It will give you a gaussian kernel density estimate (used in signal processing.)

```
In [25]: df = sb.load_dataset('iris')
         sb.distplot(df['petal_length'],kde = False)
         plt.show()
```



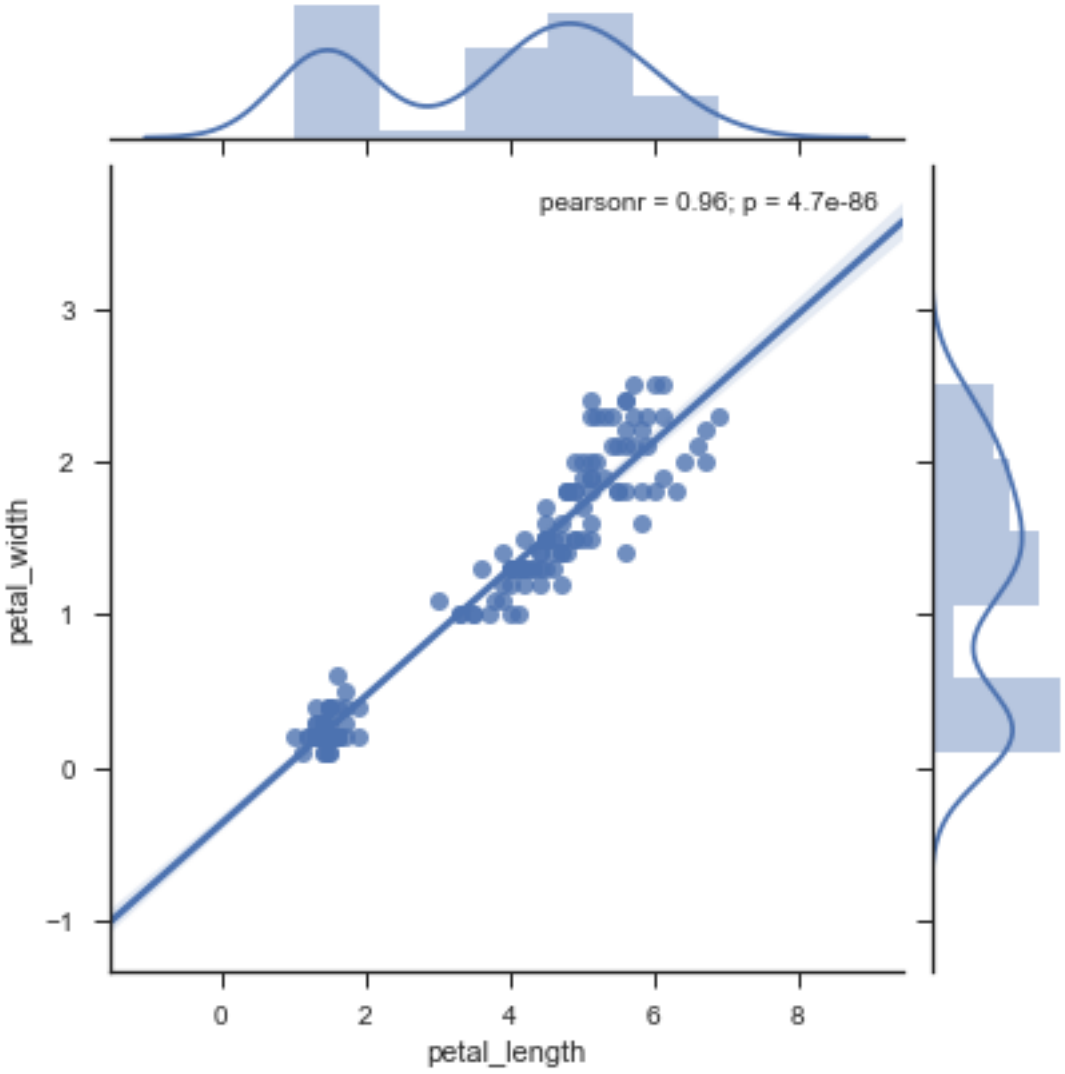
Scatterpolot in Seaborn using the Iris dataset. There are mutiple ways to do this. The folowing is a jointplot in Seaborn that plots two variables and shows their relationship. It also gives you Pearson r and marginal histogram.

```
In [27]: sb.jointplot(x = 'petal_length',y = 'petal_width',data = df)
         plt.show()
```



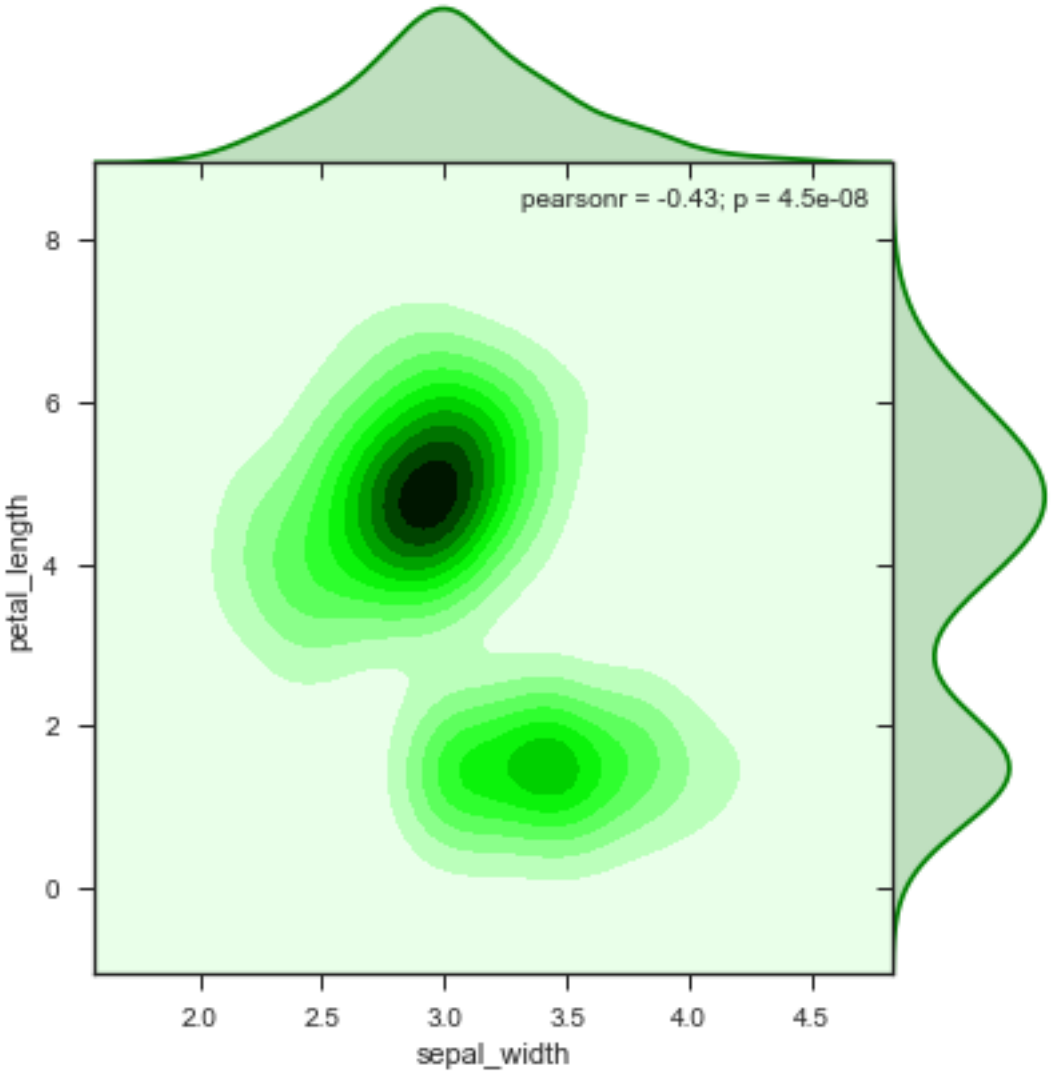
You can add regression and kernel density fit.

```
In [33]: sb.jointplot("petal_length", "petal_width", data=df, kind="reg")
         plt.show()
```



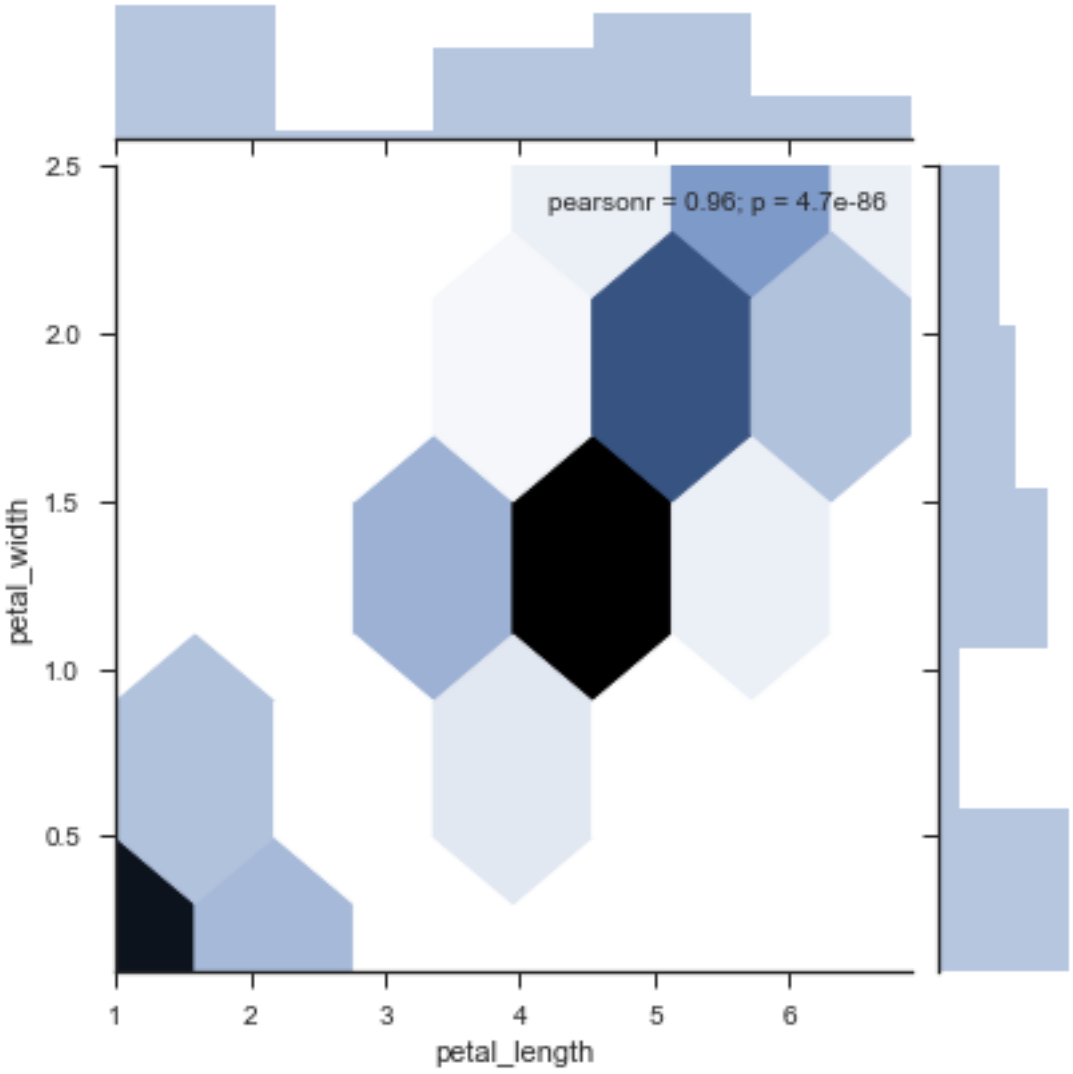
There appears be a strong linear relationship between petal_width and petal_length.
Density estimates.

```
In [37]: sb.jointplot("sepal_width", "petal_length", data=df, kind="kde", space=0, color="g")
         plt.show()
```



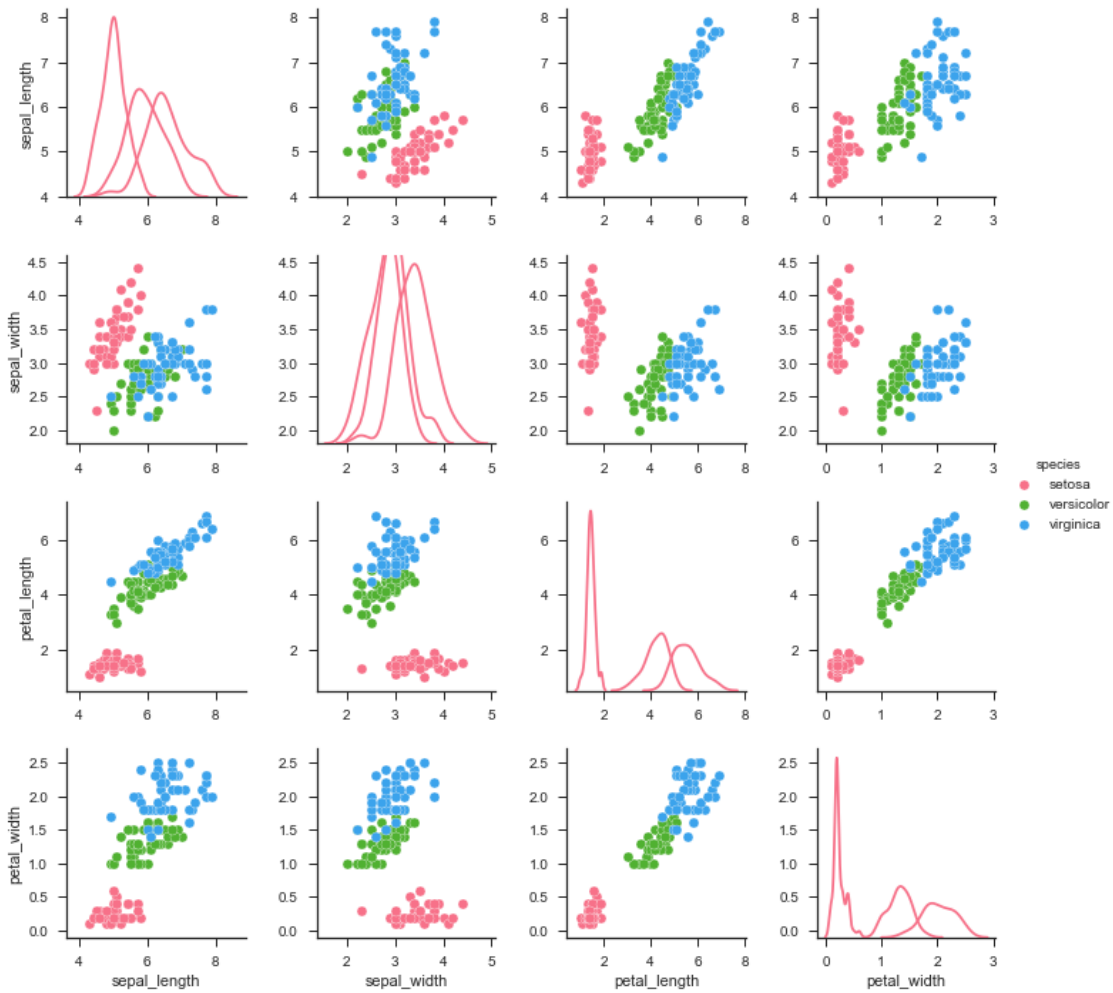
If your data is very scattered, you may use hexbin plot instead of scatterplot.

```
In [38]: sb.jointplot(x = 'petal_length',y = 'petal_width',data = df,kind = 'hex')
         plt.show()
```



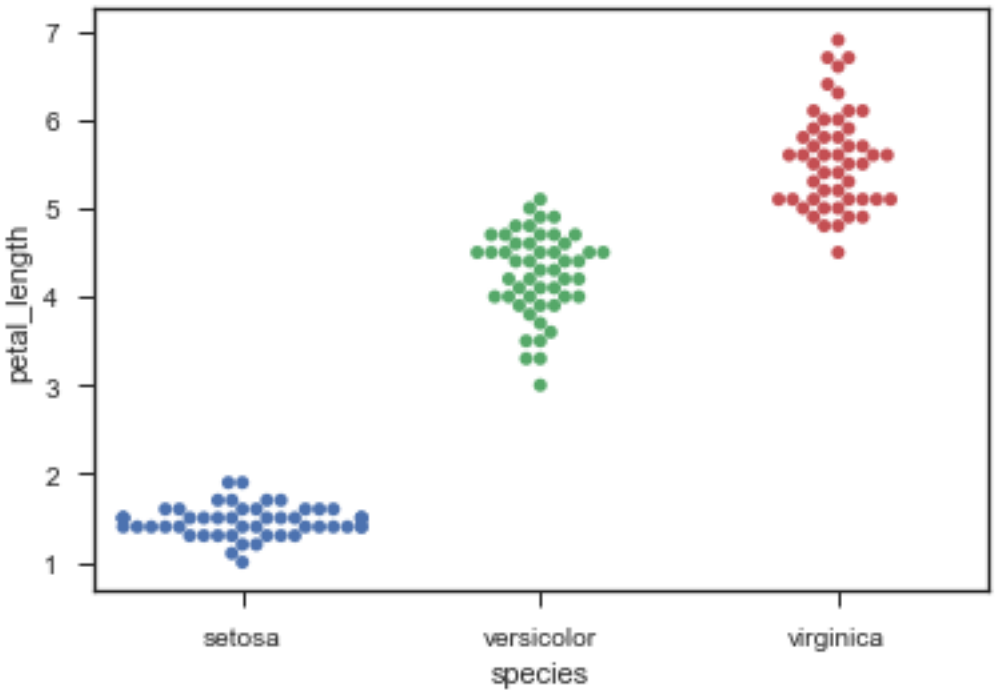
Use pairplot() to plot multiple pairwise bivariate distributions in a dataset.

```
In [39]: sb.set_style("ticks")
         sb.pairplot(df,hue = 'species',diag_kind = "kde",kind = "scatter",palette = "husl")
         plt.show()
```



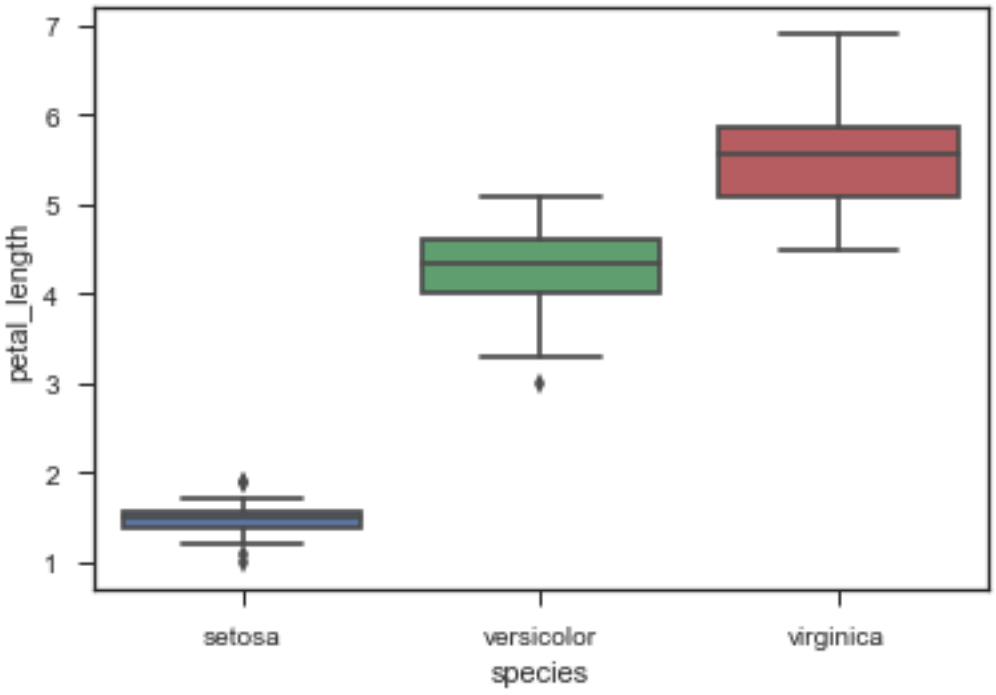
Swarmplot

```
In [40]: sb.swarmplot(x = "species", y = "petal_length", data = df)
         plt.show()
```



Boxplot

```
In [42]: sb.boxplot(x = "species", y = "petal_length", data = df)
         plt.show()
```



Violin plot

```
In [43]: sb.violinplot(x = "species", y = "petal_length", data = df)
         plt.show()
```