Visuvalization

- Matplotlib
- seaborn
- ggplot
- plotpy

Matplotlib

- Used for 2D visulaization

In [1]:
```
1 pip install matplotlib
```

```
Requirement already satisfied: matplotlib in c:\users\alekhya\anaconda3\lib\sit
e-packages (3.0.3)
Requirement already satisfied: numpy>=1.10.0 in c:\users\alekhya\anaconda3\lib
\site-packages (from matplotlib) (1.16.2)
Requirement already satisfied: cycler>=0.10 in c:\users\alekhya\anaconda3\lib\s
ite-packages (from matplotlib) (0.10.0)
Requirement already satisfied: kiwisolver>=1.0.1 in c:\users\alekhya\anaconda3
\lib\site-packages (from matplotlib) (1.0.1)
Requirement already satisfied: pyparsing!=2.0.4,!=2.1.2,!=2.1.6,>=2.0.1 in c:\u
sers\alekhya\anaconda3\lib\site-packages (from matplotlib) (2.3.1)
Requirement already satisfied: python-dateutil>=2.1 in c:\users\alekhya\anacond
a3\lib\site-packages (from matplotlib) (2.8.0)
Requirement already satisfied: six in c:\users\alekhya\anaconda3\lib\site-packa
ges (from cycler>=0.10->matplotlib) (1.12.0)
Requirement already satisfied: setuptools in c:\users\alekhya\anaconda3\lib\sit
e-packages (from kiwisolver>=1.0.1->matplotlib) (40.8.0)
Note: you may need to restart the kernel to use updated packages.

WARNING: You are using pip version 20.2.3; however, version 21.1.1 is availabl
e.
You should consider upgrading via the 'C:\Users\Alekhya\Anaconda3\python.exe -m
pip install --upgrade pip' command.
```

In [1]:
```
1 import matplotlib.pyplot as plt
```

In [2]:
```
1 print(dir(plt))
```

```
['Annotation', 'Arrow', 'Artist', 'AutoLocator', 'Axes', 'Button', 'Circle', 'F
igure', 'FigureCanvasBase', 'FixedFormatter', 'FixedLocator', 'FormatStrFormatt
er', 'Formatter', 'FuncFormatter', 'GridSpec', 'IndexLocator', 'Line2D', 'Linea
rLocator', 'Locator', 'LogFormatter', 'LogFormatterExponent', 'LogFormatterMath
text', 'LogLocator', 'MaxNLocator', 'MultipleLocator', 'Normalize', 'NullFormat
ter', 'NullLocator', 'Number', 'PolarAxes', 'Polygon', 'Rectangle', 'ScalarForm
atter', 'Slider', 'Subplot', 'SubplotTool', 'Text', 'TickHelper', 'Widget', '_I
NSTALL_FIG_OBSERVER', '_IP_REGISTERED', '__builtins__', '__cached__', '__doc_
_', '__file__', '__loader__', '__name__', '__package__', '__spec__', '_auto_dra
w_if_interactive', '_autogen_docstring', '_backend_mod', '_get_running_interact
ive_framework', '_interactive_bk', '_log', '_pylab_helpers', '_setp', '_setup_p
yplot_info_docstrings', '_show', '_string_to_bool', 'acorr', 'angle_spectrum',
'annotate', 'arrow', 'autoscale', 'autumn', 'axes', 'axhline', 'axhspan', 'axi
s', 'axvline', 'axvspan', 'bar', 'barbs', 'barh', 'bone', 'box', 'boxplot', 'br
oken_barh', 'cla', 'clabel', 'clf', 'clim', 'close', 'cm', 'cohere', 'colorba
r', 'colormaps', 'connect', 'contour', 'contourf', 'cool', 'copper', 'csd', 'cy
cler', 'dedent', 'delaxes', 'deprecated', 'disconnect', 'docstring', 'draw', 'd
raw_all', 'draw_if_interactive', 'errorbar', 'eventplot', 'figaspect', 'figimag
e', 'figlegend', 'fignum_exists', 'figtext', 'figure', 'fill', 'fill_between',
'fill_betweenx', 'findobj', 'flag', 'gca', 'gcf', 'gci', 'get', 'get_backend',
'get_cmap', 'get_current_fig_manager', 'get_figlabels', 'get_fignums', 'get_plo
t_commands', 'get_scale_docs', 'get_scale_names', 'getp', 'ginput', 'gray', 'gr
id', 'hexbin', 'hist', 'hist2d', 'hlines', 'hot', 'hsv', 'importlib', 'imread',
'imsave', 'imshow', 'inferno', 'inspect', 'install_repl_displayhook', 'interact
ive', 'ioff', 'ion', 'isinteractive', 'jet', 'legend', 'locator_params', 'loggi
ng', 'loglog', 'magma', 'magnitude_spectrum', 'margins', 'matplotlib', 'matsho
w', 'minorticks_off', 'minorticks_on', 'mlab', 'new_figure_manager', 'nipy_spec
tral', 'np', 'pause', 'pcolor', 'pcolormesh', 'phase_spectrum', 'pie', 'pink',
'plasma', 'plot', 'plot_date', 'plotfile', 'plotting', 'polar', 'prism', 'psd',
'pylab_setup', 'quiver', 'quiverkey', 'rc', 'rcParams', 'rcParamsDefault', 'rcP
aramsOrig', 'rc_context', 'rcdefaults', 'rcsetup', 're', 'register_cmap', 'rgri
ds', 'savefig', 'sca', 'scatter', 'sci', 'semilogx', 'semilogy', 'set_cmap', 's
etp', 'show', 'silent_list', 'specgram', 'spring', 'spy', 'stackplot', 'stem',
'step', 'streamplot', 'style', 'subplot', 'subplot2grid', 'subplot_tool', 'subp
lots', 'subplots_adjust', 'summer', 'suptitle', 'switch_backend', 'sys', 'tabl
e', 'text', 'thetagrids', 'tick_params', 'ticklabel_format', 'tight_layout', 't
ime', 'title', 'tricontour', 'tricontourf', 'tripcolor', 'triplot', 'twinx', 't
winy', 'uninstall_repl_displayhook', 'violinplot', 'viridis', 'vlines', 'waitfo
rbuttonpress', 'warn_deprecated', 'warnings', 'winter', 'xcorr', 'xkcd', 'xlabe
l', 'xlim', 'xscale', 'xticks', 'ylabel', 'ylim', 'yscale', 'yticks']
```

In [3]:
```
1 help(plt)
```

```
C:\Users\Alekhya\Anaconda3\lib\site-packages\matplotlib\__init__.py:886: Matp
lotlibDeprecationWarning:
examples.directory is deprecated; in the future, examples will be found relat
ive to the 'datapath' directory.
  "found relative to the 'datapath' directory.".format(key))
C:\Users\Alekhya\Anaconda3\lib\site-packages\matplotlib\__init__.py:886: Matp
lotlibDeprecationWarning:
examples.directory is deprecated; in the future, examples will be found relat
ive to the 'datapath' directory.
  "found relative to the 'datapath' directory.".format(key))
```

Types of plotting

- Line plot
- Bar chart
- pie chart
- Box plot
- how to read an image and how to save an image and how to display an image

- Line plot

In [4]:
```
1 import numpy as np
```

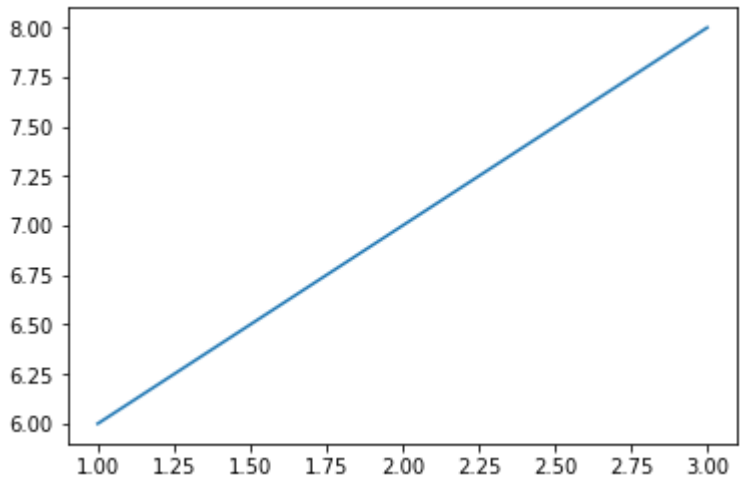
In [5]:
```
1 x = np.array([10,20,30])
2 y = [30,12,45]
3 plt.plot(x,y)
4 plt.show()
5
```

In [2]:
```
import matplotlib.pyplot as plt
help(plt.plot)
```

```
Help on function plot in module matplotlib.pyplot:

plot(*args, scalex=True, scaley=True, data=None, **kwargs)
    Plot y versus x as lines and/or markers.

    Call signatures::

        plot([x], y, [fmt], data=None, **kwargs)
        plot([x], y, [fmt], [x2], y2, [fmt2], ..., **kwargs)

    The coordinates of the points or line nodes are given by *x*, *y*.

    The optional parameter *fmt* is a convenient way for defining basic
    formatting like color, marker and linestyle. It's a shortcut string
    notation described in the *Notes* section below.

    >>> plot(x, y)        # plot x and y using default line style and color
    >>> plot(x, y, 'bo')  # plot x and y using blue circle markers
    >>> plot(y)           # plot y using x as index array 0..N-1
```

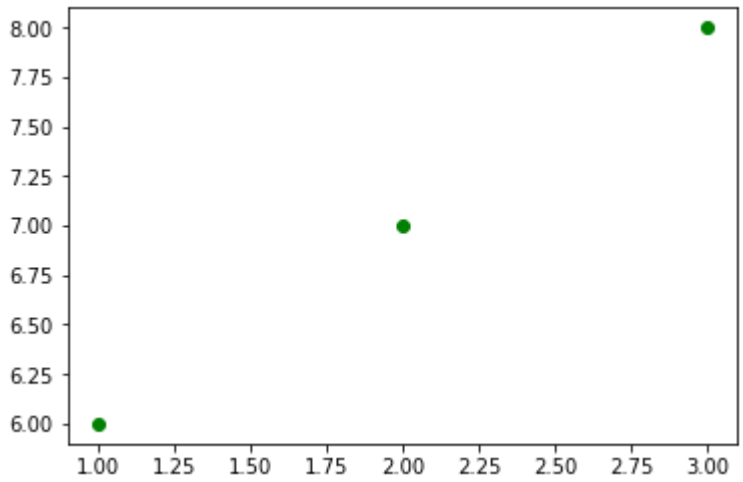
In [3]:
```
x = [1,2,3]
y = [6,7,8]
plt.plot(x,y)
```

Out[3]:
```
[<matplotlib.lines.Line2D at 0x2359a519b38>]
```



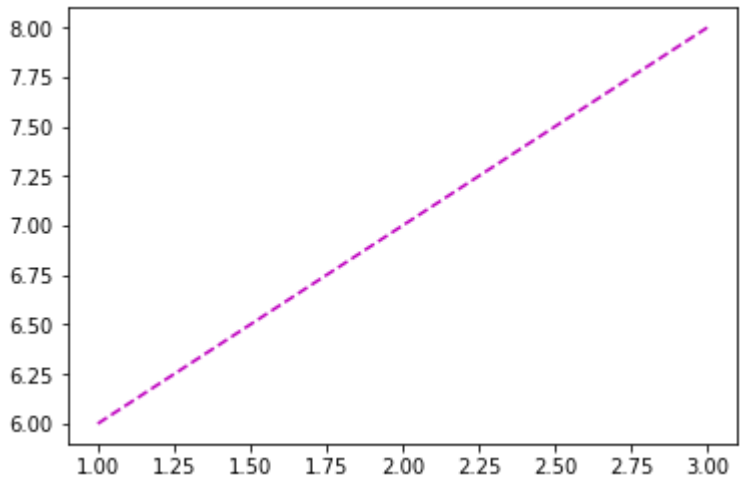
In [4]:
```
1 plt.plot(x,y,"go")
```

Out[4]: [<matplotlib.lines.Line2D at 0x2359a5cdef0>]



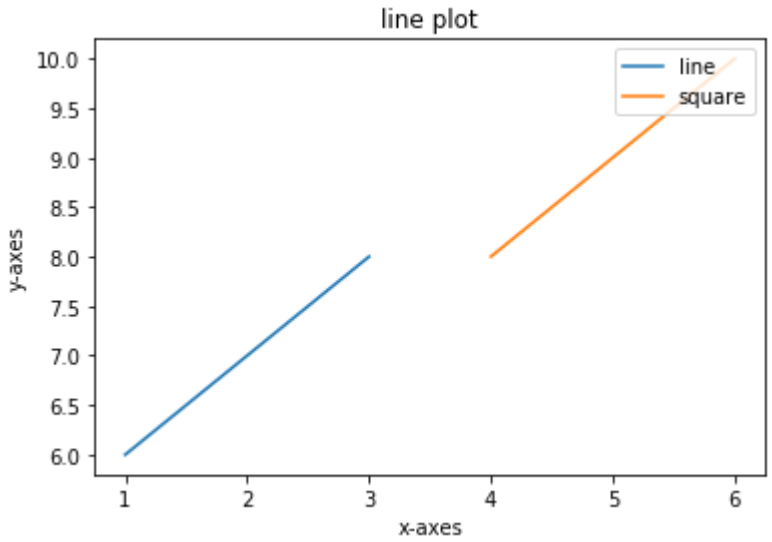
In [5]:
```
1 plt.plot(x,y,"m--")
```

Out[5]: [<matplotlib.lines.Line2D at 0x2359a641e10>]



In [11]:
```
x1 = [4,5,6]
x2 = [8,9,10]
plt.plot(x,y,label = "line")
plt.plot(x1,x2,label = "square")
plt.legend(loc=1)
plt.xlabel("x-axes")
plt.ylabel("y-axes")
plt.title("line plot")
```

Out[11]: Text(0.5, 1.0, 'line plot')

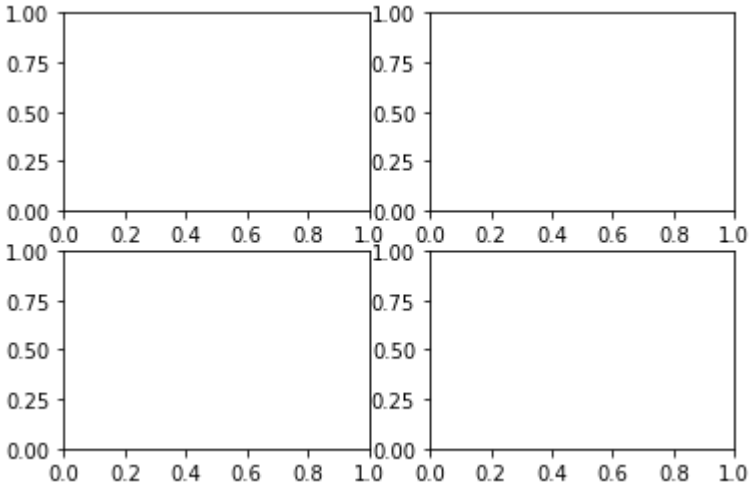


In [9]:
```
z = [1,2,3]
import numpy as np
x =np.array([1,2,3])
x**2

```

Out[9]: array([1, 4, 9], dtype=int32)

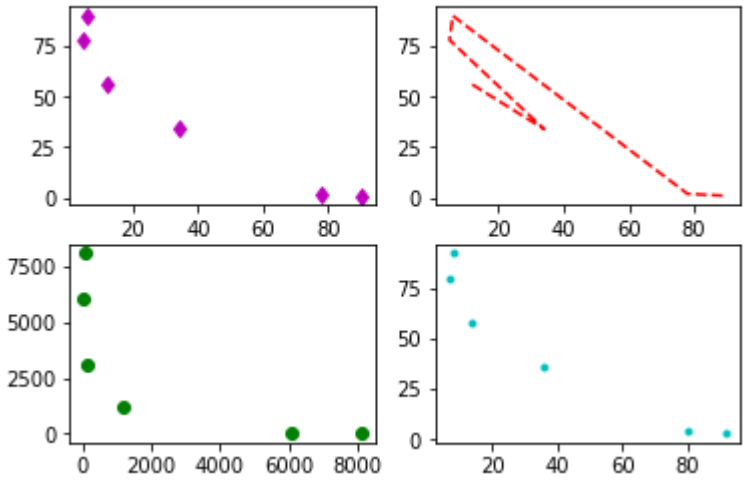
subplots

In [13]:
```
1 plt.subplots(2,2)
2 plt.show()
```



In [16]:
```
x = np.array([12,34,5,6,78,90])
y = np.array([56,34,78,90,2,1])
plt.subplot(2,2,1)#row,column,position
plt.plot(x,y,"md")
plt.subplot(2,2,2)
plt.plot(x,y,"r--")
plt.subplot(2,2,3)
plt.plot(x**2,y**2,"go")
plt.subplot(2,2,4)
plt.plot(x+2,y+2,"c.")
```

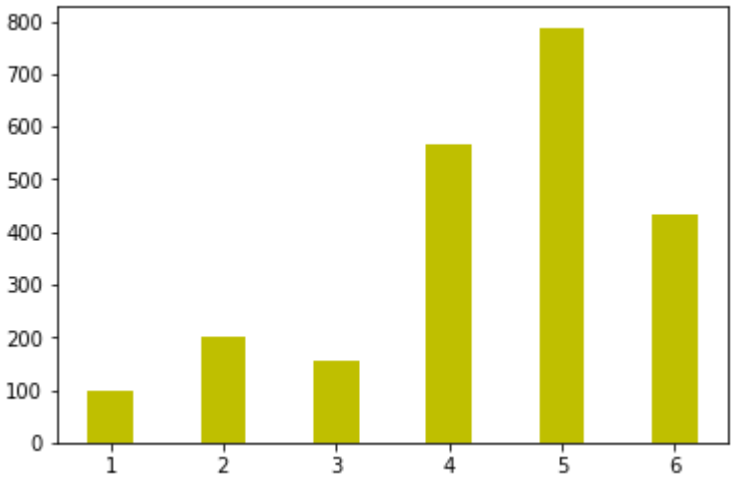
Out[16]:
```
[<matplotlib.lines.Line2D at 0x2359a9c5a90>]
```



barcharts

In [19]:
```
plt.bar([1,2,3,4,5,6],[100,200,155,566,788,433],color = "y",width = 0.4)
```

Out[19]:
```
<BarContainer object of 6 artists>
```



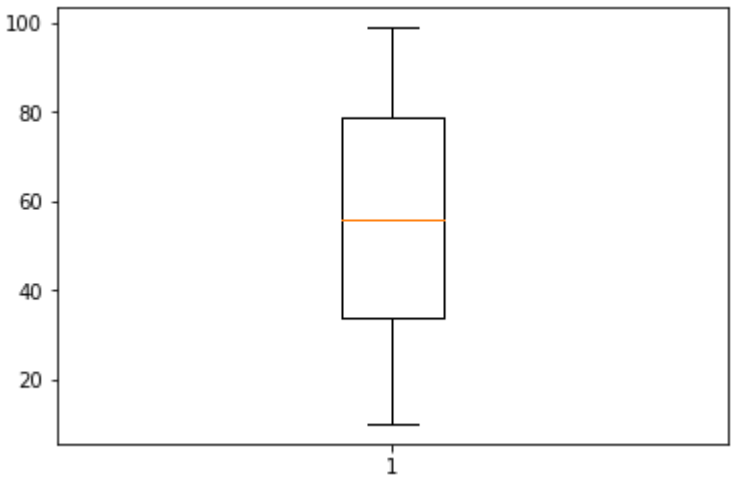
In [20]:
```
1 help(plt.bar)
```

```
Help on function bar in module matplotlib.pyplot:

bar(x, height, width=0.8, bottom=None, *, align='center', data=None, **kwarg
s)
    Make a bar plot.

    The bars are positioned at *x* with the given *align*\ment. Their
    dimensions are given by *width* and *height*. The vertical baseline
    is *bottom* (default 0).

    Each of *x*, *height*, *width*, and *bottom* may either be a scalar
    applying to all bars, or it may be a sequence of length N providing a
    separate value for each bar.

    Parameters
    ----------
    x : sequence of scalars
        The x coordinates of the bars. See also *align* for the
        alignment of the bars to the coordinates.
```

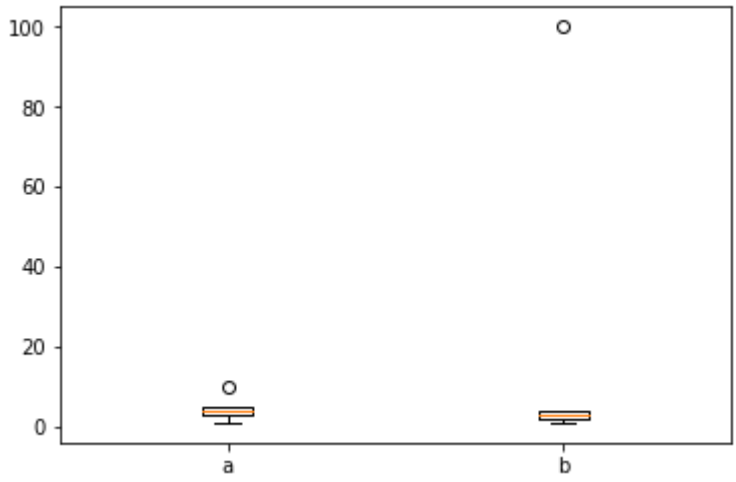
Box plot

- this is also known as whisker plot
- to find the outliers

In [21]:
```
1 x = np.random.randint(10,100,101)
2 plt.boxplot(x)
3 plt.show()
```

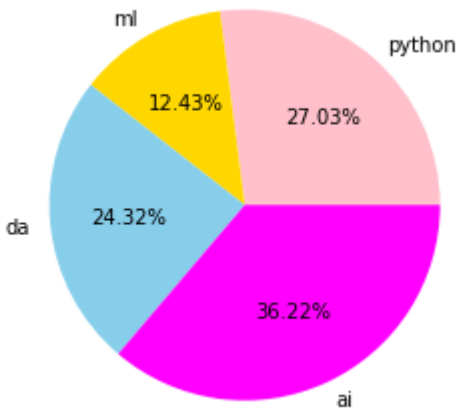


In [23]:
```
a = [1,10,3,4,5]
b = [1,2,3,4,100]
plt.boxplot([a,b],labels=["a","b"])
plt.show()
```

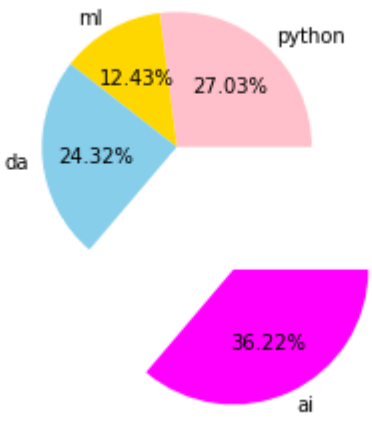


Pie chart

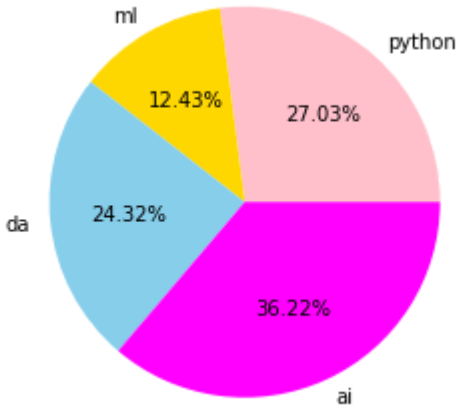
In [37]:
```
labels = ["python","ml","da","ai"]
val = [50,23,45,67]
col = ["pink","gold","skyblue","magenta"]
plt.pie(val,labels=labels,autopct = "%1.2f%%",colors=col)
plt.axis("equal")
plt.show()
```



In [39]:
```
labels = ["python","ml","da","ai"]
val = [50,23,45,67]
col = ["pink","gold","skyblue","magenta"]
e = (0,0,0,1)
plt.pie(val,labels=labels,autopct = "%1.2f%%",colors=col,explode=e)
plt.axis("equal")
plt.show()
```



In [42]:
```
labels = ["python","ml","da","ai"]
val = [50,23,45,67]
col = ["pink","gold","skyblue","magenta"]
plt.pie(val,labels=labels,autopct = "%1.2f%%",colors=col)
plt.axis("equal")
plt.savefig("pieplot.jpg") # d://foldername//filename.png
plt.show()
```

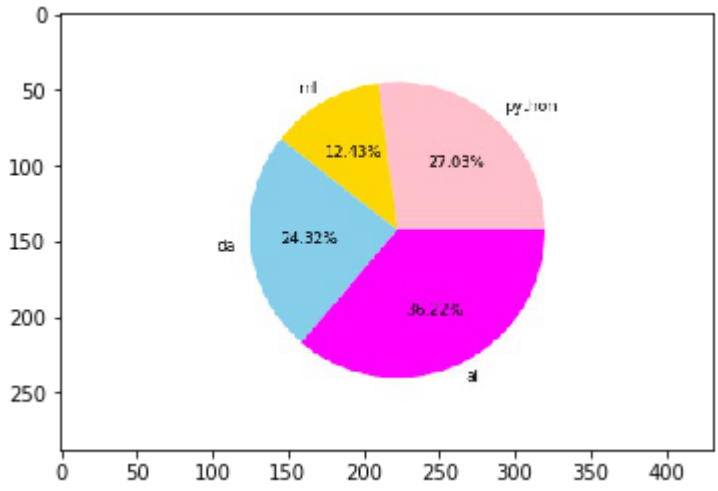


In [43]:
```
# reading an image
```

In [44]:
```
a = plt.imread("pieplot.jpg")
a
```

Out[44]:
```
array([[[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       ...,

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]],

       [[255, 255, 255],
        [255, 255, 255],
        [255, 255, 255],
        ...,
        [255, 255, 255],
        [255, 255, 255],
        [255, 255, 255]]], dtype=uint8)
```

In [48]:
```
plt.imshow(a)
plt.figure(figsize = (5,5))
plt.show()
```




```
<Figure size 360x360 with 0 Axes>
```

Seaborn

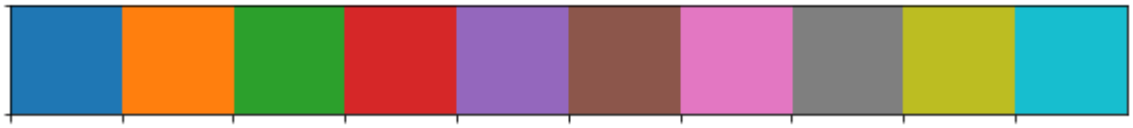
- seaborn library is based on matplotlib.it provides a high level interface for drawing attractive and informative statical graphs

In [49]:
```
import seaborn as sns
```

In [50]:
```
print(dir(sns))
```

```
['FacetGrid', 'JointGrid', 'PairGrid', '__builtins__', '__cached__', '__doc__',
'__file__', '__loader__', '__name__', '__package__', '__path__', '__spec__', '_
_version__', '_orig_rc_params', 'algorithms', 'axes_style', 'axisgrid', 'barplo
t', 'blend_palette', 'boxenplot', 'boxplot', 'categorical', 'catplot', 'choose_
colorbrewer_palette', 'choose_cubehelix_palette', 'choose_dark_palette', 'choos
e_diverging_palette', 'choose_light_palette', 'clustermap', 'cm', 'color_palett
e', 'colors', 'countplot', 'crayon_palette', 'crayons', 'cubehelix_palette', 'd
ark_palette', 'desaturate', 'despine', 'distplot', 'distributions', 'diverging_
palette', 'dogplot', 'external', 'factorplot', 'get_dataset_names', 'heatmap',
'hls_palette', 'husl_palette', 'jointplot', 'kdeplot', 'light_palette', 'linepl
ot', 'lmplot', 'load_dataset', 'lvplot', 'matrix', 'miscplot', 'mpl', 'mpl_pale
tte', 'pairplot', 'palettes', 'palplot', 'plotting_context', 'pointplot', 'rcmo
d', 'regplot', 'regression', 'relational', 'relplot', 'reset_defaults', 'reset_
orig', 'residplot', 'rugplot', 'saturate', 'scatterplot', 'set', 'set_color_cod
es', 'set_context', 'set_hls_values', 'set_palette', 'set_style', 'stripplot',
'swarmplot', 'timeseries', 'tsplot', 'utils', 'violinplot', 'widgets', 'xkcd_pa
lette', 'xkcd_rgb']
```

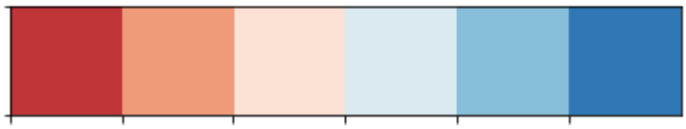
In [51]:
```
1 sns.palplot(sns.color_palette())
```



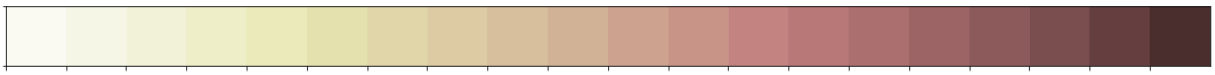
In [52]:
```
1 sns.color_palette()
```

Out[52]:
```
[(0.12156862745098039, 0.4666666666666667, 0.7058823529411765),
 (1.0, 0.4980392156862745, 0.054901960784313725),
 (0.17254901960784313, 0.6274509803921569, 0.17254901960784313),
 (0.8392156862745098, 0.15294117647058825, 0.1568627450980392),
 (0.5803921568627451, 0.403921568627451, 0.7411764705882353),
 (0.5490196078431373, 0.33725490196078434, 0.29411764705882354),
 (0.8901960784313725, 0.4666666666666667, 0.7607843137254902),
 (0.4980392156862745, 0.4980392156862745, 0.4980392156862745),
 (0.7372549019607844, 0.7411764705882353, 0.13333333333333333),
 (0.09019607843137255, 0.7450980392156863, 0.8117647058823529)]
```

In [54]:
```
1 sns.palplot(sns.color_palette("RdBu"))
```



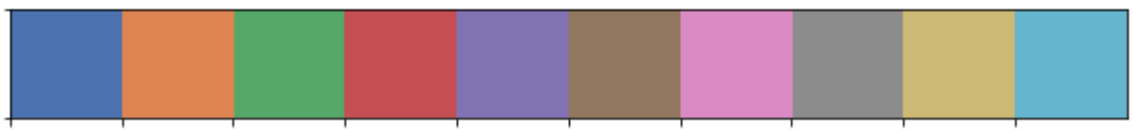
In [60]:
```
1 sns.palplot(sns.color_palette("pink_r",n_colors=20))
```



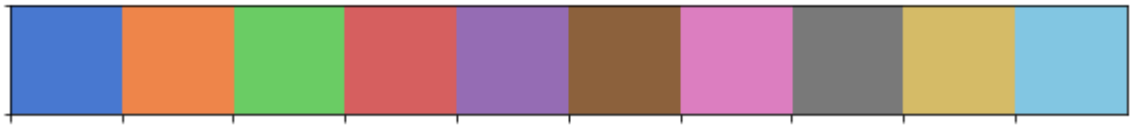
seaborn has six variations of its default color palette

- deep
- muted
- dark
- bright
- pastel
- colorblind

In [61]:
```
1 sns.palplot(sns.color_palette("deep"))
```



In [62]:
```
1 sns.palplot(sns.color_palette("muted"))
```



In [63]:
```
1 sns.palplot(sns.color_palette("pastel"))
```

In [64]:
```
1 sns.get_dataset_names()
```

```
C:\Users\Alekhya\Anaconda3\lib\site-packages\seaborn\utils.py:376: UserWarning:
No parser was explicitly specified, so I'm using the best available HTML parser
for this system ("lxml"). This usually isn't a problem, but if you run this cod
e on another system, or in a different virtual environment, it may use a differ
ent parser and behave differently.

The code that caused this warning is on line 376 of the file C:\Users\Alekhya\A
naconda3\lib\site-packages\seaborn\utils.py. To get rid of this warning, pass t
he additional argument 'features="lxml"' to the BeautifulSoup constructor.

  gh_list = BeautifulSoup(http)
```

Out[64]:
```
['anagrams',
 'anscombe',
 'attention',
 'brain_networks',
 'car_crashes',
 'diamonds',
 'dots',
 'exercise',
 'flights',
 'fmri',
 'gammas',
 'geyser',
 'iris',
 'mpg',
 'penguins',
 'planets',
 'tips',
 'titanic']
```

In [67]:
```
1 s = sns.load_dataset("iris")
2 s.head()
```

Out[67]:

	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 [72]:
```
1 s["species"].value_counts()
```

Out[72]:
```
setosa        50
virginica     50
versicolor    50
Name: species, dtype: int64
```

In [73]:
```
1 s.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 150 entries, 0 to 149
Data columns (total 5 columns):
sepal_length    150 non-null float64
sepal_width     150 non-null float64
petal_length    150 non-null float64
petal_width     150 non-null float64
species         150 non-null object
dtypes: float64(4), object(1)
memory usage: 5.9+ KB
```

In [74]:
```
1 s.describe()
```

Out[74]:

	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 [75]:
```
1 s1 = sns.load_dataset("titanic")
2 s1.head()
```

Out[75]:

	survived	pclass	sex	age	sibsp	parch	fare	embarked	class	who	adult_male	de
0	0	3	male	22.0	1	0	7.2500	S	Third	man	True	Na
1	1	1	female	38.0	1	0	71.2833	C	First	woman	False	
2	1	3	female	26.0	0	0	7.9250	S	Third	woman	False	Na
3	1	1	female	35.0	1	0	53.1000	S	First	woman	False	
4	0	3	male	35.0	0	0	8.0500	S	Third	man	True	Na

In [76]:
```
1 s
```

Out[76]:

	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
5	5.4	3.9	1.7	0.4	setosa
6	4.6	3.4	1.4	0.3	setosa
7	5.0	3.4	1.5	0.2	setosa
8	4.4	2.9	1.4	0.2	setosa
9	4.9	3.1	1.5	0.1	setosa
10	5.4	3.7	1.5	0.2	setosa
11	4.8	3.4	1.6	0.2	setosa
12	4.8	3.0	1.4	0.1	setosa
13	4.3	3.0	1.1	0.1	setosa
14	5.8	4.0	1.2	0.2	setosa
15	5.7	4.4	1.5	0.4	setosa
16	5.4	3.9	1.3	0.4	setosa
17	5.1	3.5	1.4	0.3	setosa
18	5.7	3.8	1.7	0.3	setosa
19	5.1	3.8	1.5	0.3	setosa
20	5.4	3.4	1.7	0.2	setosa
21	5.1	3.7	1.5	0.4	setosa
22	4.6	3.6	1.0	0.2	setosa
23	5.1	3.3	1.7	0.5	setosa
24	4.8	3.4	1.9	0.2	setosa
25	5.0	3.0	1.6	0.2	setosa
26	5.0	3.4	1.6	0.4	setosa
27	5.2	3.5	1.5	0.2	setosa
28	5.2	3.4	1.4	0.2	setosa
29	4.7	3.2	1.6	0.2	setosa
...	...	...	...	...	...
120	6.9	3.2	5.7	2.3	virginica
121	5.6	2.8	4.9	2.0	virginica
122	7.7	2.8	6.7	2.0	virginica

	sepal_length	sepal_width	petal_length	petal_width	species
123	6.3	2.7	4.9	1.8	virginica
124	6.7	3.3	5.7	2.1	virginica
125	7.2	3.2	6.0	1.8	virginica
126	6.2	2.8	4.8	1.8	virginica
127	6.1	3.0	4.9	1.8	virginica
128	6.4	2.8	5.6	2.1	virginica
129	7.2	3.0	5.8	1.6	virginica
130	7.4	2.8	6.1	1.9	virginica
131	7.9	3.8	6.4	2.0	virginica
132	6.4	2.8	5.6	2.2	virginica
133	6.3	2.8	5.1	1.5	virginica
134	6.1	2.6	5.6	1.4	virginica
135	7.7	3.0	6.1	2.3	virginica
136	6.3	3.4	5.6	2.4	virginica
137	6.4	3.1	5.5	1.8	virginica
138	6.0	3.0	4.8	1.8	virginica
139	6.9	3.1	5.4	2.1	virginica
140	6.7	3.1	5.6	2.4	virginica
141	6.9	3.1	5.1	2.3	virginica
142	5.8	2.7	5.1	1.9	virginica
143	6.8	3.2	5.9	2.3	virginica
144	6.7	3.3	5.7	2.5	virginica
145	6.7	3.0	5.2	2.3	virginica
146	6.3	2.5	5.0	1.9	virginica
147	6.5	3.0	5.2	2.0	virginica
148	6.2	3.4	5.4	2.3	virginica
149	5.9	3.0	5.1	1.8	virginica

150 rows × 5 columns

In [78]:
```
1 print(dir(sns))
```

```
['FacetGrid', 'JointGrid', 'PairGrid', '__builtins__', '__cached__', '__doc__',
'__file__', '__loader__', '__name__', '__package__', '__path__', '__spec__', '_
_version__', '_orig_rc_params', 'algorithms', 'axes_style', 'axisgrid', 'barplo
t', 'blend_palette', 'boxenplot', 'boxplot', 'categorical', 'catplot', 'choose_
colorbrewer_palette', 'choose_cubehelix_palette', 'choose_dark_palette', 'choos
e_diverging_palette', 'choose_light_palette', 'clustermap', 'cm', 'color_palett
e', 'colors', 'countplot', 'crayon_palette', 'crayons', 'cubehelix_palette', 'd
ark_palette', 'desaturate', 'despine', 'distplot', 'distributions', 'diverging_
palette', 'dogplot', 'external', 'factorplot', 'get_dataset_names', 'heatmap',
'hls_palette', 'husl_palette', 'jointplot', 'kdeplot', 'light_palette', 'linepl
ot', 'lmplot', 'load_dataset', 'lvplot', 'matrix', 'miscplot', 'mpl', 'mpl_pale
tte', 'pairplot', 'palettes', 'palplot', 'plotting_context', 'pointplot', 'rcmo
d', 'regplot', 'regression', 'relational', 'relplot', 'reset_defaults', 'reset_
orig', 'residplot', 'rugplot', 'saturate', 'scatterplot', 'set', 'set_color_cod
es', 'set_context', 'set_hls_values', 'set_palette', 'set_style', 'stripplot',
'swarmplot', 'timeseries', 'tsplot', 'utils', 'violinplot', 'widgets', 'xkcd_pa
lette', 'xkcd_rgb']
```

categorical methods

- strip plot
- swarm plot

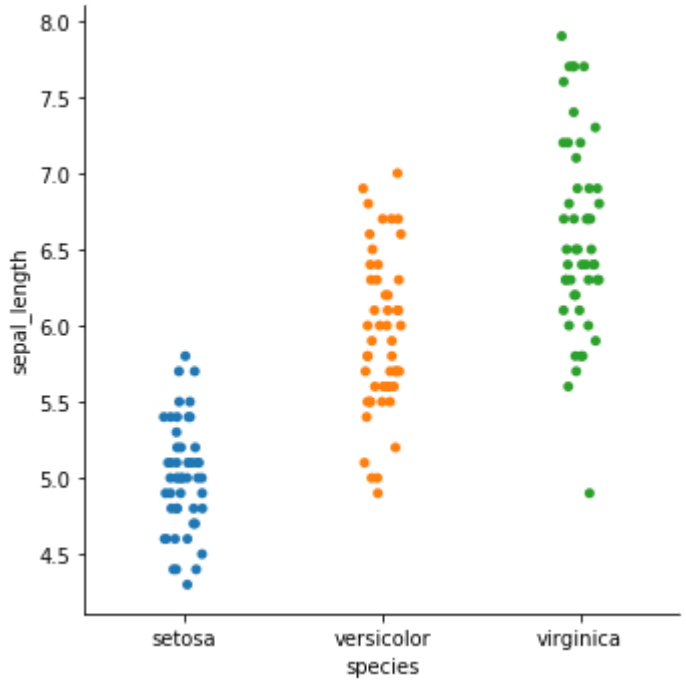
In [80]:
```
1 s.columns
```

Out[80]:
```
Index(['sepal_length', 'sepal_width', 'petal_length', 'petal_width',
       'species'],
      dtype='object')
```

In [ ]:
```
1
```

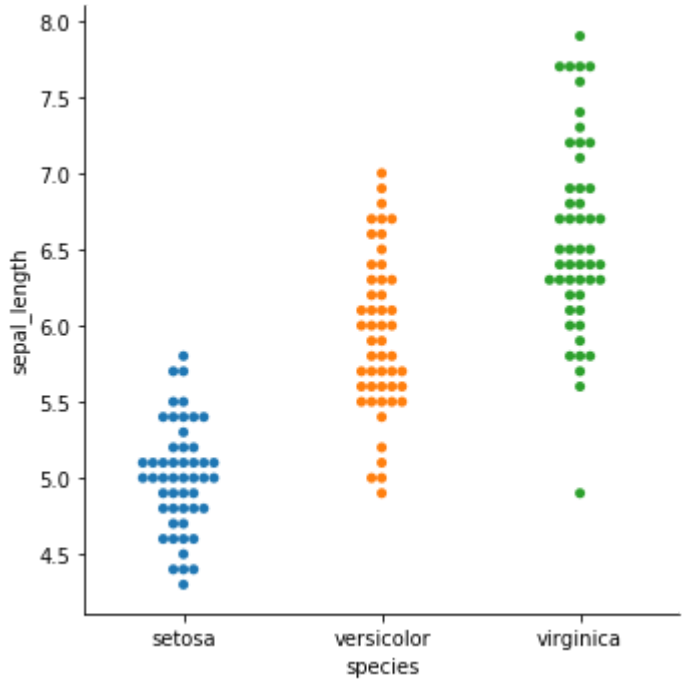
In [85]:
```
1 sns.catplot(x="species", y="sepal_length", data=s)
```

Out[85]: <seaborn.axisgrid.FacetGrid at 0x2359d7adef0>



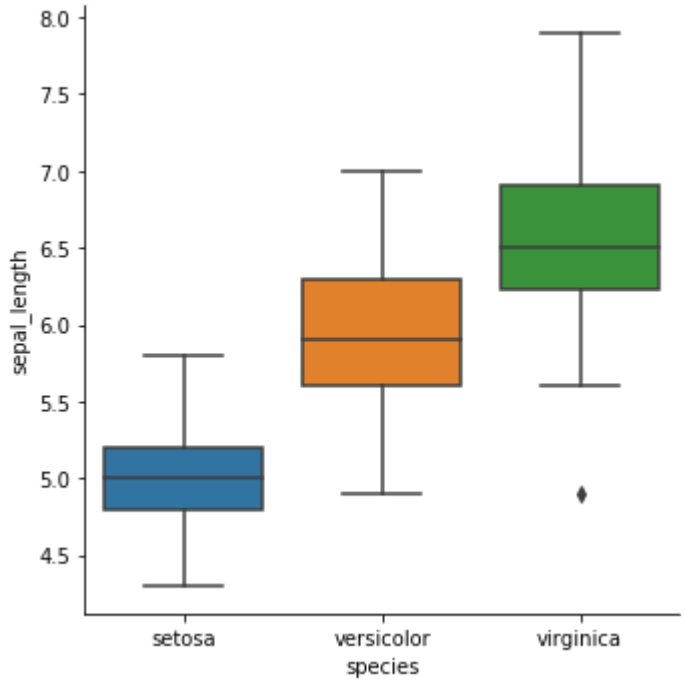
In [87]:
```
#swarm plot- adjust the points along categorical data using
#some algorithms that prevents the overlapping of data
sns.catplot(x="species",y="sepal_length",data=s,kind="swarm")
```

Out[87]: <seaborn.axisgrid.FacetGrid at 0x2359c028908>



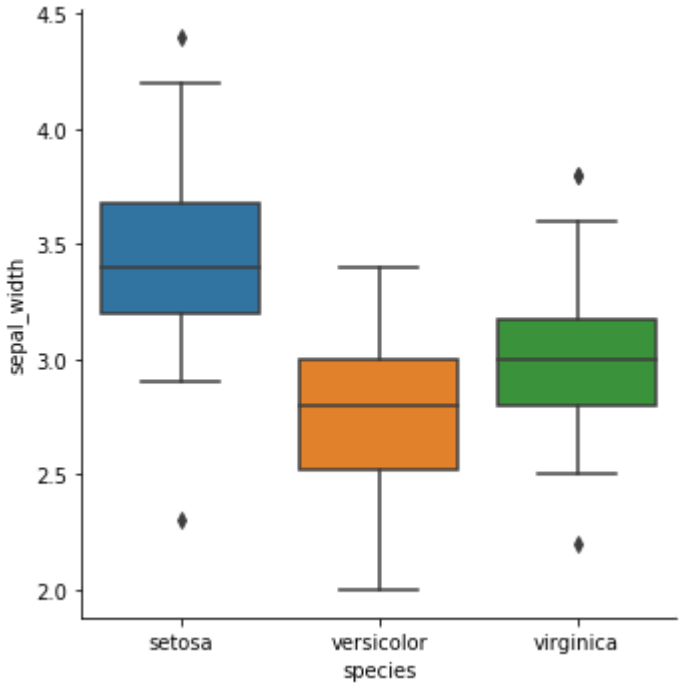
In [88]:
```
1 sns.catplot(x = "species",y = "sepal_length",data=s,kind="box")
```

Out[88]: <seaborn.axisgrid.FacetGrid at 0x2359f248198>



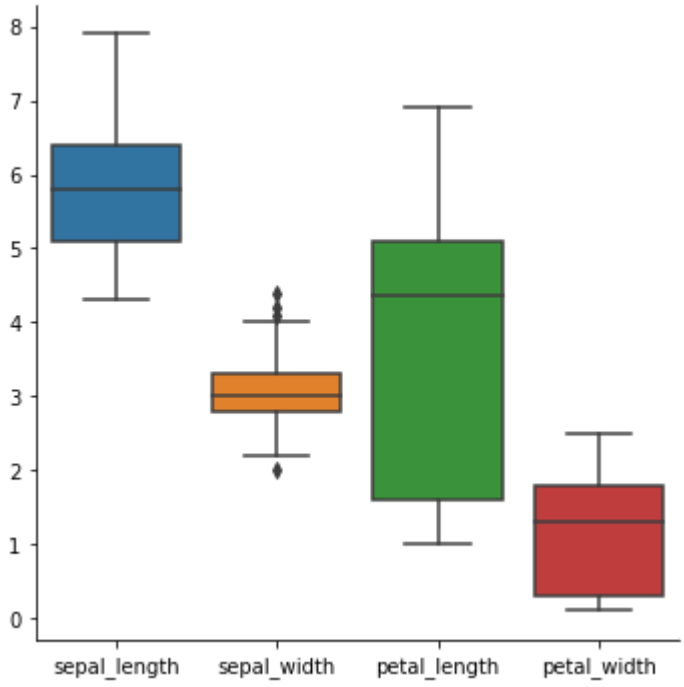
In [89]:
```
1 sns.catplot(x = "species",y = "sepal_width",data=s,kind="box")
```

Out[89]: <seaborn.axisgrid.FacetGrid at 0x2359f259160>



In [90]:
```
1 sns.catplot(data=s,kind="box")
```

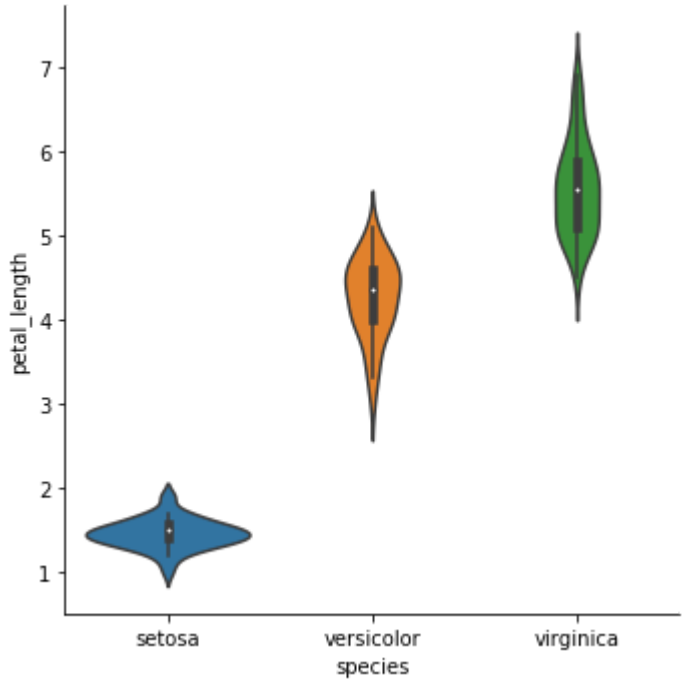
Out[90]: <seaborn.axisgrid.FacetGrid at 0x2359d842438>



In [91]:
```
1 ### Violin plot
```

In [92]:
```
1 sns.catplot(x="species",y="petal_length",data=s,kind="violin")
```

Out[92]: <seaborn.axisgrid.FacetGrid at 0x2359d6ce7b8>

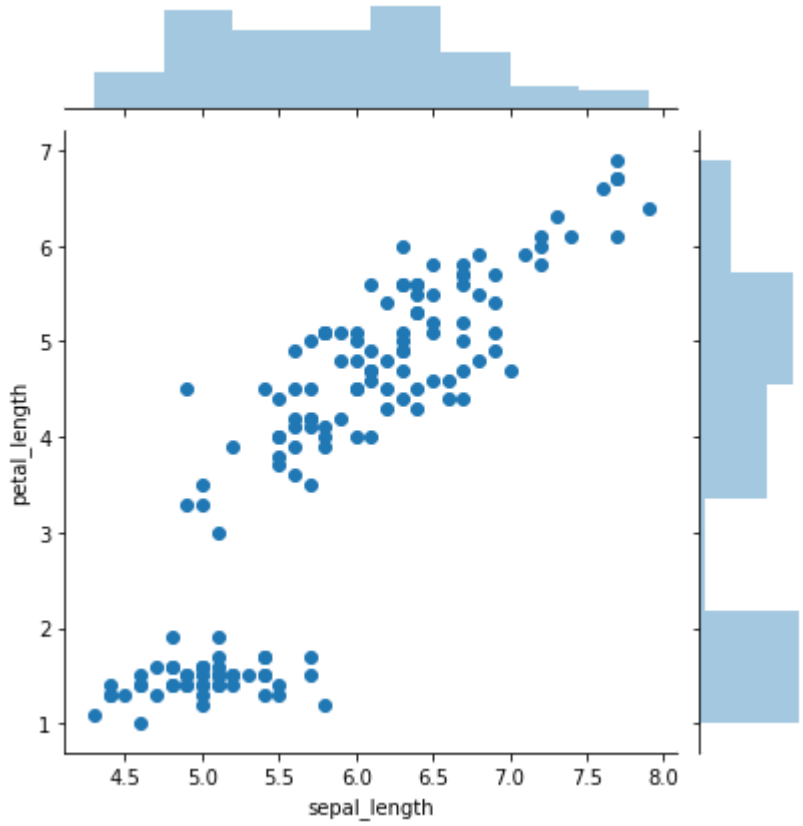


joint plot

- it combines multiple kinds of plots for getting the insites from the data

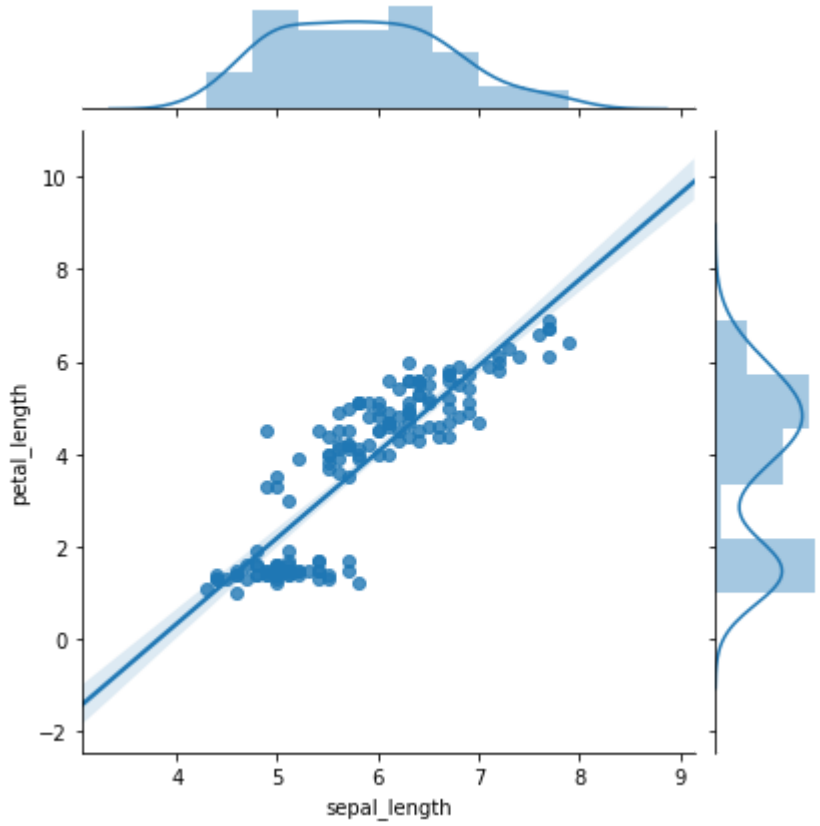
In [93]:
```
1 sns.jointplot(x="sepal_length",y="petal_length",data=s)
```

Out[93]: <seaborn.axisgrid.JointGrid at 0x2359d8306a0>



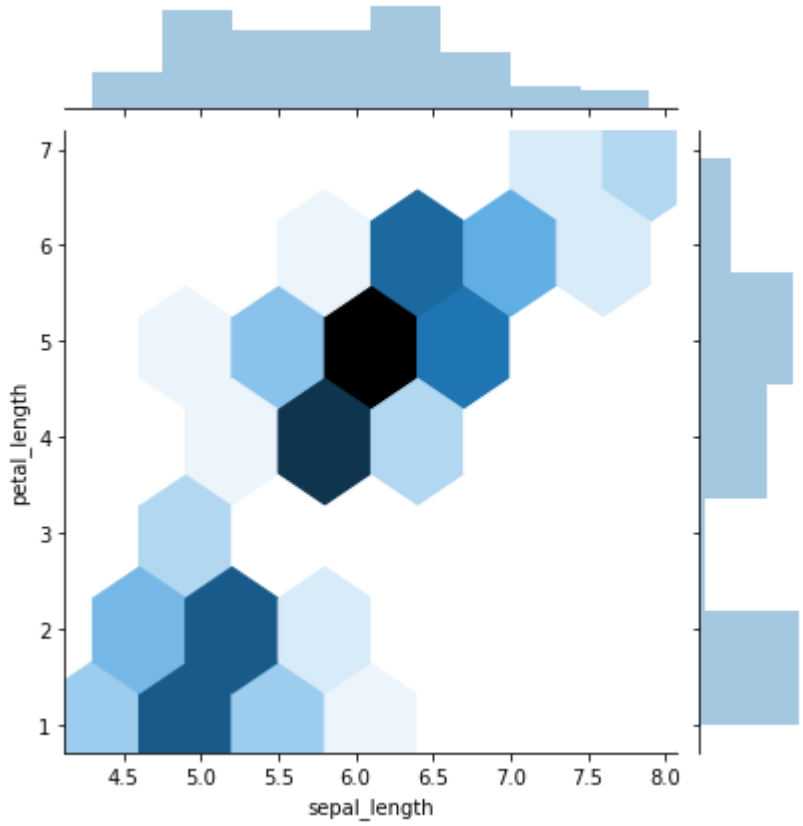
In [94]:
```
1 sns.jointplot(x="sepal_length",y="petal_length",kind ="reg",data=s)
```

Out[94]: <seaborn.axisgrid.JointGrid at 0x2359eca1438>



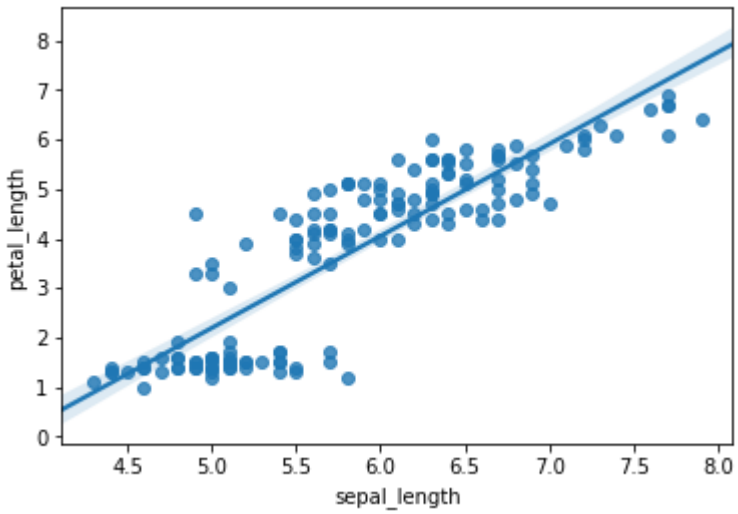
In [95]:
```
1 sns.jointplot(x="sepal_length",y="petal_length",kind ="hex",data=s)
```

Out[95]: <seaborn.axisgrid.JointGrid at 0x2359f30d828>



In [96]:
```
1 sns.regplot(x = "sepal_length",y="petal_length",data=s)
```

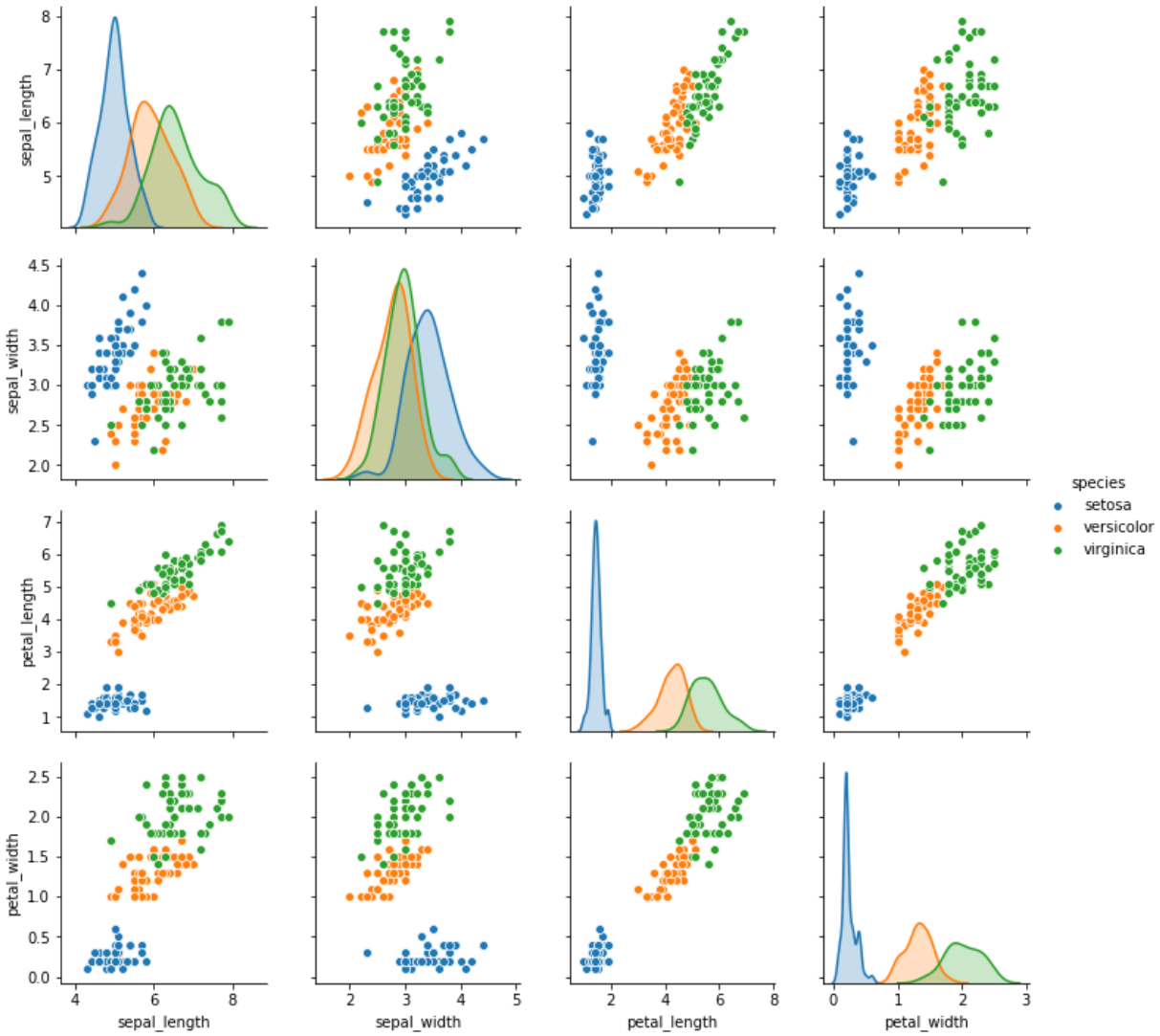
Out[96]: <matplotlib.axes._subplots.AxesSubplot at 0x2359f40d0b8>



pair plot

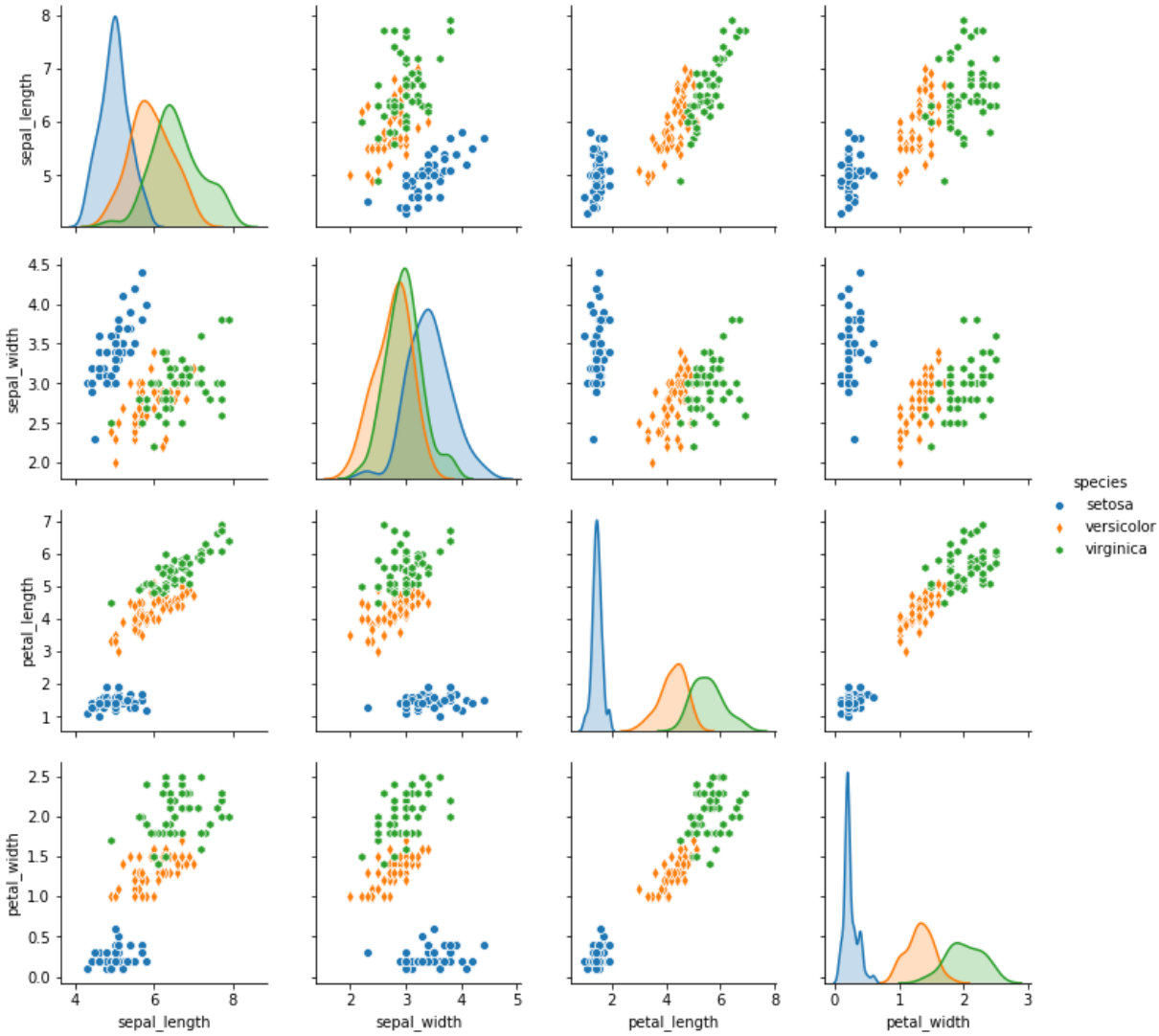
In [97]:
```
1 sns.pairplot(data=s,hue="species")
```

Out[97]: <seaborn.axisgrid.PairGrid at 0x2359f498c88>



In [98]:
```
1 sns.pairplot(data=s,hue="species",markers=["o","d","h"])
```

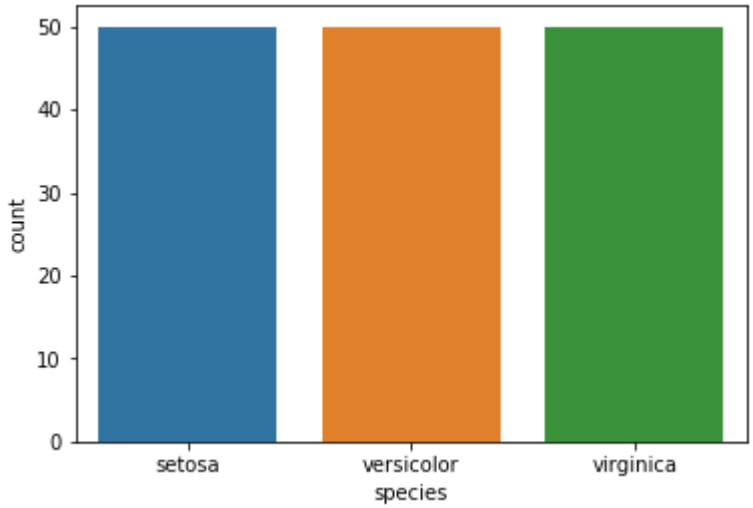
Out[98]: <seaborn.axisgrid.PairGrid at 0x235a0f14ac8>



Countplot

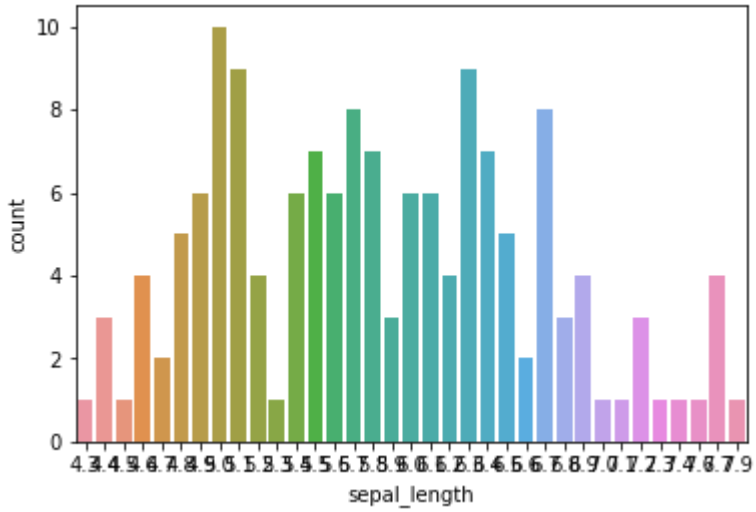
In [100]:
```
1 sns.countplot(s["species"])
```

Out[100]: <matplotlib.axes._subplots.AxesSubplot at 0x235a186edd8>



In [101]:
```
1 sns.countplot(s["sepal_length"])
```

Out[101]: <matplotlib.axes._subplots.AxesSubplot at 0x235a1ee6828>



In [102]:
```
1 s["sepal_length"].value_counts()
```

Out[102]:
```
5.0    10
6.3     9
5.1     9
6.7     8
5.7     8
5.5     7
5.8     7
6.4     7
6.0     6
4.9     6
6.1     6
5.4     6
5.6     6
6.5     5
4.8     5
7.7     4
6.9     4
5.2     4
6.2     4
4.6     4
7.2     3
6.8     3
4.4     3
5.9     3
6.6     2
4.7     2
7.6     1
7.4     1
4.3     1
7.9     1
7.3     1
7.0     1
4.5     1
5.3     1
7.1     1
Name: sepal_length, dtype: int64
```

Heatmap

In [103]:
```
1 corr = s.corr()
```

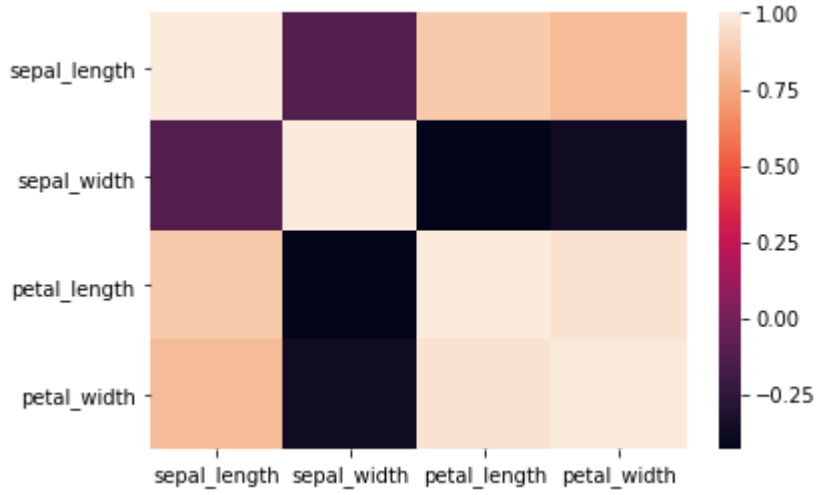
In [104]:
```
1 corr
```

Out[104]:

	sepal_length	sepal_width	petal_length	petal_width
sepal_length	1.000000	-0.117570	0.871754	0.817941
sepal_width	-0.117570	1.000000	-0.428440	-0.366126
petal_length	0.871754	-0.428440	1.000000	0.962865
petal_width	0.817941	-0.366126	0.962865	1.000000

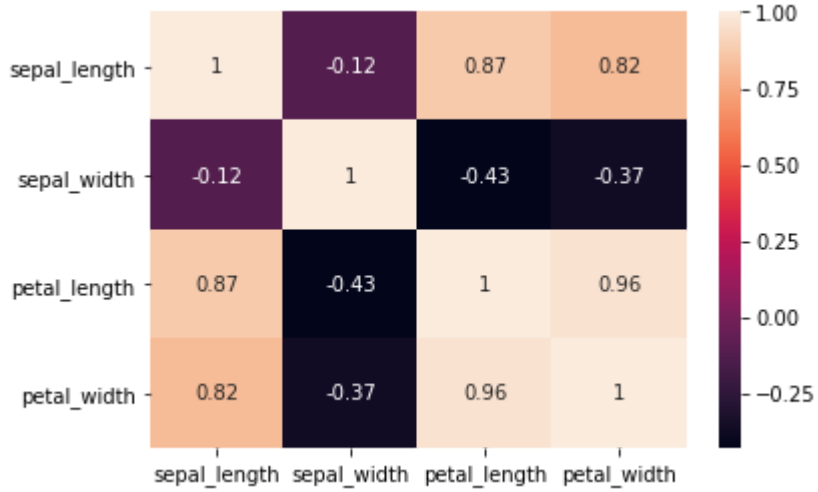
In [105]:
```
1 sns.heatmap(corr)
```

Out[105]: `<matplotlib.axes._subplots.AxesSubplot at 0x235a201f1d0>`



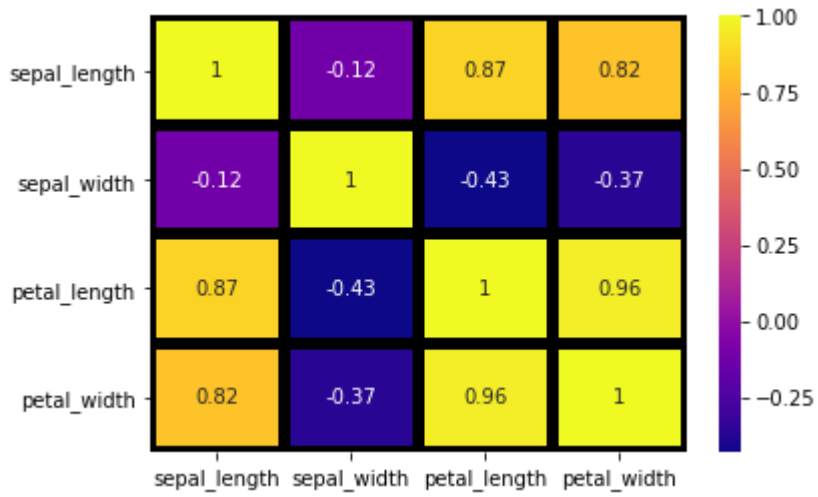
In [107]:
```
1 sns.heatmap(corr,annot=True)
```

Out[107]: `<matplotlib.axes._subplots.AxesSubplot at 0x235a20beda0>`



In [110]:
```
1 sns.heatmap(corr,annot=True,cmap="plasma",linecolor="black",linewidth=5)
```

Out[110]: <matplotlib.axes._subplots.AxesSubplot at 0x235a2246ac8>



In [111]:
```
1 help(sns.heatmap)
```

```
Help on function heatmap in module seaborn.matrix:

heatmap(data, vmin=None, vmax=None, cmap=None, center=None, robust=False, ann
ot=None, fmt='.2g', annot_kws=None, linewidths=0, linecolor='white', cbar=Tru
e, cbar_kws=None, cbar_ax=None, square=False, xticklabels='auto', yticklabels
='auto', mask=None, ax=None, **kwargs)
    Plot rectangular data as a color-encoded matrix.

    This is an Axes-level function and will draw the heatmap into the
    currently-active Axes if none is provided to the ``ax`` argument.  Part o
f
    this Axes space will be taken and used to plot a colormap, unless ``cbar`
`
    is False or a separate Axes is provided to ``cbar_ax``.

    Parameters
    ----------
    data : rectangular dataset
        2D dataset that can be coerced into an ndarray. If a Pandas DataFrame
```

In [ ]:
```
1
```