

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
```

Creating Arrays	Notes
<code>np.array([[1, 2, 3], [4, 5, 5]])</code>	From nested list; optionally specify <code>dtype='float'</code>
<code>np.full((3, 5), 2)</code>	3x5 array full of 2s
<code>np.arange(0, 10, 3)</code>	Equivalent to <code>np.array(range(0, 10, 3))</code>
<code>np.linspace(0, 2, 10)</code>	10 numbers evenly spaced from 0 through 2
<code>np.random.random((5, 2))</code>	5x2 array of random numbers in [0, 1). See also <code>np.random.randint</code> , <code>np.random.normal</code>

NumPy Expression	Notes
<code>a.ndim</code>	Number of dimensions of a
<code>a.shape</code>	Shape of a (a tuple)
<code>a.size</code>	Total number of elements in a
<code>a.dtype</code>	Data type of a's elements
<code>a[2, 3]</code>	Element in row 2, column 3 of a. Each index can be a slice
<code>a[2:4, 5:8]</code>	Rows 2-3 and 5-7 of a. A view without <code>.copy()</code>
<code>a.reshape(3, 16)</code>	Elements of a rearranged into a 3x16 array. A view
<code>np.concatenate((a, b))</code>	a and b stacked. To concatenate side-by-side: <code>axis=1</code>
<code>a + np.sin(b)</code>	Applied element-wise. If a is 4x1 and b is 1x6, result is 4x6, broadcast to make dimensions match
<code>np.sum(a, axis=0)</code>	Sums along row dimension, giving one sum per column. See also <code>prod</code> , <code>mean</code> , <code>std</code> , <code>min</code> , <code>max</code>
<code>a[(a > 3) & (a < 10)]</code>	Elements of a greater than and less than 10. <code>a > 3</code> itself is an array of bools. Use <code> </code> for or, <code>~</code> for not

```
df = pd.read_csv('path/to/your_file.csv')
```

Pandas Expression	Notes
<code>pd.DataFrame({'a':[1, 2, 3], 'b':[4, 5, 6]})</code>	A dataframe with the specified values in columns 'a' and 'b'.
<code>df.columns</code>	Column names of df
<code>df.head()</code>	First five rows of df
<code>df.describe()</code>	Basic stats on numeric columns of df
<code>df['area']</code>	The 'area' column of df
<code>df.loc['Oregon']</code>	The row of df indexed by 'Oregon'. Can specify column in addition.

Pandas Expression	Notes
<code>df.iloc[3]</code>	Row 3 of df
<code>df.dropna()</code>	A copy of df without any rows containing NaN
<code>pd.concat([a, b])</code>	Concatenation of DataFrames a and b
<code>pd.merge(a, b, how='inner')</code>	Joins a and b; how can be 'inner', 'outer', 'left', or 'right'
<code>df.sort_values(by='name')</code>	A copy of df, sorted by the 'name' column
<code>df.pivot_table('survived', index='sex', columns='class', aggfunc='mean')</code>	Pivot table showing the mean survival rate for each combination of sex and class

Matplotlib Expression or Statement	Notes
<code>plt.plot(x, y)</code>	Line plot of x and y (Series, DataFrame columns, or lists). Optional third argument like '--c' for dashed cyan line
<code>plt.scatter(x, y)</code>	Scatter plot of x and y
<code>plt.hist(x)</code>	Histogram of x
<code>plt.xlim(0, 10)</code>	Set x limits of plot
<code>plt.xlabel('Year')</code>	Set x label of plot
<code>plt.title('Duck Prices')</code>	Set title of plot
<code>plt.legend()</code>	Add legend
<code>plt.text(x, y, 'look here')</code>	Add annotation
<code>fig, ax = plt.subplots(2, 3)</code>	Creates a figure with a 2x3 grid of axes

Named Argument	Matplotlib Functions	Notes
c	plot, scatter	Color (plot) or sequence of colors (scatter)
s	scatter	Sequence of sizes (numbers)
label	plot, scatter	Label to use in legend
marker	plot, scatter	Marker to use for each point

Marker	Linestyle	Color
'.' point	'-' solid	'blue'
'o' circle	':' dotted	'g' green, from among rgbcmk
'v', '^', '<' triangles	'--' dashed	'0.75' grayscale
's' square	'-.' dashdot	'#FFDD44' hex code
'+' plus		(1.0, 0.2, 0.3) RGB tuple
'x' x		
'D' diamond		Color examples from VanderPlas, <i>Python Data Science Handbook</i>