Python for Data Science

NumPy Arrays

NumPy Arrays

In [26]:
```
my_list = [1,2,3]
```

In [28]:
```
import numpy as np
```

In [30]:
```
arr = np.array(my_list)
```

In [31]:
```
arr
```

Out[31]:
```
array([1, 2, 3])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

```
In [28]: import numpy as np
```

```
In [30]: arr = np.array(my_list)
```

```
In [31]: arr
```

```
Out[31]: array([1, 2, 3])
```

```
In [32]: my_mat = [[1,2,3],[4,5,6],[7,8,9]]
```

```
In [34]: np.array(my_mat)
```

```
Out[34]: array([[1, 2, 3],
                [4, 5, 6],
                [7, 8, 9]])
```

```
In [ ]:
```

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

In [31]:
```
arr
```

Out[31]:
```
array([1, 2, 3])
```

In [32]:
```
my_mat = [[1,2,3],[4,5,6],[7,8,9]]
```

In [34]:
```
np.array(my_mat)
```

Out[34]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [ ]:
```
np.arange()
```

Docstring:
arange([start,] stop[, step,], dtype=None)

Return evenly spaced values within a given interval.

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

Out[31]:
```
array([1, 2, 3])
```

In [32]:
```
my_mat = [[1,2,3],[4,5,6],[7,8,9]]
```

In [34]:
```
np.array(my_mat)
```

Out[34]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [36]:
```
np.arange(0,11)
```

Out[36]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

Out[31]:
```
array([1, 2, 3])
```

In [32]:
```
my_mat = [[1,2,3],[4,5,6],[7,8,9]]
```

In [34]:
```
np.array(my_mat)
```

Out[34]:
```
array([[1, 2, 3],
       [4, 5, 6],
       [7, 8, 9]])
```

In [37]:
```
np.arange(0,11,2)
```

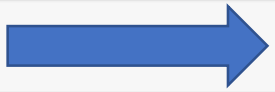
Out[37]:
```
array([ 0,  2,  4,  6,  8, 10])
```

In [ ]:

```
       [7, 8, 9]])
```

In [37]:
```
np.arange(0,11,2)
```

Out[37]:
```
array([ 0,  2,  4,  6,  8, 10])
```



2 in (0,11,2) is the step size in -- .arrange()

In [38]:
```
np.zeros(3)
```

Out[38]:
```
array([ 0.,  0.,  0.])
```

In [39]:
```
np.zeros((5,5))
```

Out[39]:
```
array([[ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.],
       [ 0.,  0.,  0.,  0.,  0.]])
```

In [ ]:

```
[7, 8, 9]])
```

In [37]:
```
np.arange(0,11,2)
```

Out[37]:
```
array([ 0,  2,  4,  6,  8, 10])
```

In [38]:
```
np.zeros(3)
```

Out[38]:
```
array([ 0.,  0.,  0.])
```

In [40]:
```
np.zeros((2,3))
```

Out[40]:
```
array([[ 0.,  0.,  0.],
       [ 0.,  0.,  0.]])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

In [40]:
```
np.zeros((2,3))
```

Out[40]:
```
array([[ 0.,  0.,  0.],
       [ 0.,  0.,  0.]])
```

In [41]:
```
np.ones(4)
```

Out[41]:
```
array([ 1.,  1.,  1.,  1.])
```

In [42]:
```
np.ones((3,4))
```

Out[42]:
```
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

```
       [ 0.,  0.,  0.]])
```

In [41]:
```
np.ones(4)
```

Out[41]:
```
array([ 1.,  1.,  1.,  1.])
```

In [42]:
```
np.ones((3,4))
```

Out[42]:
```
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
```

In [ ]:
```
np.linspace()
```

```
Signature: np.linspace(start, stop, num=50, endpoint=True, retstep=False, dtype=None)
Docstring:
Return evenly spaced numbers over a specified interval.
```

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

```
In [41]: np.ones(4)
Out[41]: array([ 1.,  1.,  1.,  1.])

In [42]: np.ones((3,4))
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])

In [43]: np.linspace(0,5,10)
Out[43]: array([ 0.        ,  0.55555556,  1.11111111,  1.66666667,  2.22222222,
                 2.77777778,  3.33333333,  3.88888889,  4.44444444,  5.        ])

In [ ]:
```

No. of brackets shows 1D, 2D, or 3D arrays.

10 in (0,5,10) is 10 evenly spaced points from 0 to 5 --- .linspace()

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

In [42]:
```
np.ones((3,4))
```

Out[42]:
```
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
```

In [45]:
```
np.eye(4)
```

Out[45]:
```
array([[ 1.,  0.,  0.,  0.],
       [ 0.,  1.,  0.,  0.],
       [ 0.,  0.,  1.,  0.],
       [ 0.,  0.,  0.,  1.]])
```

Identity matrix = equal rows and columns (Square Matrix) with diagonal value 1

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

```
Out[41]: array([ 1.,  1.,  1.,  1.])

In [42]: np.ones((3,4))

Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])

In [ ]:

In [46]: np.random.rand(5)

Out[46]: array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])

In [ ]:
```

Just gives the random number arrays from Uniform distribution

```
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])
```

In [ ]:

In [46]:
```
np.random.rand(5)
```

Out[46]:
```
array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

In [47]:
```
np.random.rand(5,5)
```

Out[47]:
```
array([[ 0.7352132 ,  0.26785947,  0.05728073,  0.28446422,  0.98632407],
       [ 0.35977821,  0.15971553,  0.2508447 ,  0.97831464,  0.71733947],
       [ 0.10020856,  0.72423996,  0.40744763,  0.45633017,  0.05571964],
       [ 0.92728243,  0.874622  ,  0.3109018 ,  0.07692912,  0.02602928],
       [ 0.92694128,  0.18668115,  0.03540084,  0.41861232,  0.58422942]])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

```
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])
```

In [ ]:

Just gives the random number arrays from Normal distribution

```
In [46]: np.random.rand(5)
Out[46]: array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

```
In [49]: np.random.randn(2)
Out[49]: array([ 0.1748905 , -0.91796204])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

```
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])
```

```
In [ ]:
```

Not a tuple

```
In [46]: np.random.rand(5)
```

```
Out[46]: array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

```
In [50]: np.random.randn(4,4)
```

```
Out[50]: array([[-0.13715711, -0.62171097, -1.12971724, -2.46821243],
                [-0.28742644, -0.16806968, -0.81600717,  0.51147028],
                [-0.93184389, -1.62815983,  0.283949  ,  0.51164452],
                [-1.92648925, -0.33546775,  0.33918453,  0.98255606]])
```

```
In [ ]:
```

```
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])
```

```
In [ ]:
```

```
In [46]: np.random.rand(5)
```

```
Out[46]: array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

```
In [50]: np.random.randint()
```

```
Out[50]:
```

```
Docstring:
randint(low, high=None, size=None, dtype='l')

Return random integers from `low` (inclusive) to `high` (exclusive).
```

```
In [ ]:
```

Out[42]:
```
array([[ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.],
       [ 1.,  1.,  1.,  1.]])
```

In [ ]:

In [46]:
```
np.random.rand(5)
```

Out[46]:
```
array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

In [51]:
```
np.random.randint(1,100)
```

Out[51]:
```
60
```

In [ ]:

```
Out[42]: array([[ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.],
                [ 1.,  1.,  1.,  1.]])
```

```
In [ ]:
```

```
In [46]: np.random.rand(5)
```

```
Out[46]: array([ 0.92251232,  0.07198415,  0.12872838,  0.99571974,  0.94977206])
```

```
In [52]: np.random.randint(1,100,10)
```

```
Out[52]: array([80, 15, 64, 51, 32, 84, 34, 19, 13, 88])
```

```
In [ ]:
```

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help

Python [conda env:py35]

In [53]:
```
arr = np.arange(25)
```

In [54]:
```
arr
```

Out[54]:
```
array([ 0,  1,  2,  3,  4,  5,  6,  7,  8,  9, 10, 11, 12, 13, 14, 15, 16,
       17, 18, 19, 20, 21, 22, 23, 24])
```

In [55]:
```
ranarr = np.random.randint(0,50,10)
```

In [56]:
```
ranarr
```

Out[56]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

```
17, 18, 19, 20, 21, 22, 23, 24])
```

In [55]:
```
ranarr = np.random.randint(0,50,10)
```

In [56]:
```
ranarr
```

Out[56]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [57]:
```
arr.reshape(5,5)
```

Out[57]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [ ]:

Not a tuple => 5*5 = 25, so the numbers are too

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help | Python [conda env:py35]

In [56]:
```
ranarr
```

Out[56]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [58]:
```
arr.reshape(5,10)
```

```
---------------------------------------------------------------------------
ValueError                                Traceback (most recent call last)
<ipython-input-58-5f9d63542514> in <module>()
----> 1 arr.reshape(5,10)

ValueError: total size of new array must be unchanged
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help

Python [conda env:py35]

Out[56]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [59]:
```
arr.reshape(5,5)
```

Out[59]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [60]:
```
ranarr
```

Out[60]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [ ]:

Out[56]:
```
array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])
```

In [59]:
```
arr.reshape(5,5)
```

Out[59]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [61]:
```
ranarr.max()
```

Out[61]:
```
49
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

Out[59]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [61]:
```
ranarr.max()
```

Out[61]:
```
49
```

In [62]:
```
ranarr.min()
```

Out[62]:
```
3
```

In [ ]:

```
In [61]: ranarr.max()
Out[61]: 49

In [62]: ranarr.min()
Out[62]: 3

In [63]: ranarr
Out[63]: array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])

In [64]: ranarr.argmax()
Out[64]: 3

In [ ]:
```

Shows the index value of the maximum value of the array

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help

Python [conda env:py35]

```
In [63]: ranarr
Out[63]: array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])

In [64]: ranarr.argmax()
Out[64]: 3

In [65]: ranarr.argmin()
Out[65]: 6

In [ ]:
```

Shows the index value of the minimum value of the array

```
Out[63]: array([35, 18, 26, 49, 34, 48,  3, 30,  6,  6])

In [64]: ranarr.argmax()
Out[64]: 3

In [65]: ranarr.argmin()
Out[65]: 6

In [66]: arr.shape
Out[66]: (25,)

In [ ]:
```

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help Python [conda env:py35]

In [64]:
```
ranarr.argmax()
```

Out[64]:
```
3
```

In [65]:
```
ranarr.argmin()
```

Out[65]:
```
6
```

In [67]:
```
arr = arr.reshape(5,5)
```

In [68]:
```
arr
```

Out[68]:
```
array([[ 0,  1,  2,  3,  4],
       [ 5,  6,  7,  8,  9],
       [10, 11, 12, 13, 14],
       [15, 16, 17, 18, 19],
       [20, 21, 22, 23, 24]])
```

In [ ]:

NumPy Arrays

localhost:8888/notebooks/NumPy%20Arrays.ipynb

File Edit View Insert Cell Kernel Widgets Help

Python [conda env:py35]

In [65]:
```
ranarr.argmin()
```

Out[65]: 6

In [67]:
```
arr = arr.reshape(5,5)
```

In [69]:
```
arr.shape
```

Out[69]: (5, 5)

In [70]:
```
arr.dtype
```

Out[70]: dtype('int32')

In [ ]:

In [69]:
```
arr.shape
```

Out[69]: (5, 5)

In [70]:
```
arr.dtype
```

Out[70]: dtype('int32')

In [73]:
```
from numpy.random import randint
```

In [74]:
```
randint(2,10)
```

Out[74]: 3

In [ ]: