NUMPY , PANDAS AND MATPLOTLIB

WHAT IS NUMPY

- NumPy, which stands for Numerical Python, is a library consisting of multidimensional array objects and a collection of routines for processing those arrays.
- Using NumPy, mathematical and logical operations on arrays can be performed.
- NumPy is often used along with packages like **SciPy** (Scientific Python) and **Matplotlib** (plotting library).

CONTINUED

- Using NumPy, a developer can perform the following operations:

1. Mathematical and logical operations on arrays.
2. Fourier transforms and routines for shape manipulation.
3. Operations related to linear algebra. NumPy has in-built functions for linear algebra and random number generation.
4. It is open source, which is an added advantage of NumPy.

Working with NumPy

- If you want to use numpy in your code you just use "import numpy" and usually it is abbreviated as np.
- The most important object defined in NumPy is an N-dimensional array type called **ndarray**.

EXAMPLES

```
IMPORT NUMPY AS NP
A = NP.ARRAY([1,2,3])
PRINT(A)
-------------------------------------
MORE THAN ONE DIMENSION
IMPORT NUMPY AS NP
A = NP.ARRAY([[1, 2], [3, 4]])
PRINT (A)
```

EXAMPLES

```
IMPORT NUMPY AS NP
A = NP.ARRAY([1, 2, 3,4,5], NDMIN = 2) #MIN DIMENSION
PRINT A
---------------------------------------
IMPORT NUMPY AS NP
A = NP.ARRAY([1, 2, 3], DTYPE = COMPLEX)
PRINT A
```

DATA TYPE SUPPORTS

- NumPy supports a much greater variety of numerical types than Python does.

1. **bool_** Boolean (True or False) stored as a byte
2. **int_** Default integer type (same as C long; normally either int64 or int32)
3. **Uint8** Unsigned integer (0 to 255)
4. **Float** shorthand for float64
5. **complex_** shorthand for complex128

EXAMPLES

```
IMPORT NUMPY AS NP
DT = NP.DTYPE(NP.INT32)
PRINT DT
----------------------------------------
#INT8, INT16, INT32, INT64 CAN BE REPLACED BY EQUIVALENT STRING 'I1', 'I2','I4'
IMPORT NUMPY AS NP
DT = NP.DTYPE('I4')
PRINT DT
```

EXAMPLES

```
DT = NP.DTYPE([('AGE',NP.INT8)])
A = NP.ARRAY([(10,),(20,),(30,)], DTYPE = DT)
PRINT A['AGE']
-----------------------------------------
STUDENT = NP.DTYPE([('NAME','S20'), ('AGE', 'I1'), ('MARKS', 'F4')])
A = NP.ARRAY([('ABC', 21, 50),('XYZ', 18, 75)], DTYPE = STUDENT)
PRINT A
```

NUMPY METHODS

```
IMPORT NUMPY AS NP
A = NP.ARRAY([[1,2,3],[4,5,6]])
PRINT (A.SHAPE)
-------------------------------------
A = NP.ARRAY([[1,2,3],[4,5,6]])
A.SHAPE = (3,2)
PRINT (A) #USED TO RESIZE NOT JUST FIND THE SHAPE
-------------------------------------
A = NP.ARRAY([[1,2,3],[4,5,6]])
B = A.RESHAPE(3,2)
PRINT B
```

METHODS CONTINUED

- NUMPY.EMPTY(SHAPE, DTYPE = FLOAT, ORDER = 'C')
- C IS FOR C AND F IS FOR FORTRAN. (NOT IMPORTANT)
- NUMPY.ZEROS(SHAPE, DTYPE = FLOAT, ORDER = 'C')
- NUMPY.ONES(SHAPE, DTYPE = NONE, ORDER = 'C')

EXAMPLES

```
IMPORT NUMPY AS NP
X = NP.EMPTY([3,2], DTYPE = INT)
PRINT X
----------------------------------------
X = NP.ZEROS(5)
PRINT X
----------------------------------------
X = NP.ONES([2,2], DTYPE = INT)
PRINT X
```

METHODS CONTINUED

- NUMPY.ASARRAY :THIS FUNCTION IS SIMILAR TO NUMPY.ARRAY EXCEPT FOR THE FACT THAT IT HAS FEWER PARAMETERS. THIS ROUTINE IS USEFUL FOR CONVERTING PYTHON SEQUENCE INTO NDARRAY.

```
X = [(1,2,3),(4,5)]

A = NP.ASARRAY(X)

PRINT A
```

- `A = NP.LOGSPACE(1.0, 2.0, NUM = 10)`
- `PRINT A`

INDEXING AND SLICING

```
A = NP.ARANGE(10)
S = SLICE(2,7,2) #START STOP STEP
PRINT A[S]
X = A[2]
X = A[2:7:2]
X = A[2:]
X = A[-1]
```

ADVANCED INDEXING AND SLICING

```
X = NP.ARRAY([[1, 2], [3, 4], [5, 6]])
Y = X[[0,1,2], [0,1,0]]
PRINT Y
```

BROADCASTING

- Review matrix operations
- The term **broadcasting** refers to the ability of NumPy to treat arrays of different shapes during arithmetic operations. Arithmetic operations on arrays are usually done on corresponding elements. If two arrays are of exactly the same shape, then these operations are smoothly performed.

EXAMPLE

```
IMPORT NUMPY AS NP
A = NP.ARRAY([[0.0,0.0,0.0],[10.0,10.0,10.0],[20.0,20.0,20.0],[30.0,30.0,30.0]])
B = NP.ARRAY([1.0,2.0,3.0])

PRINT 'FIRST ARRAY:'
PRINT A
PRINT '\N'

PRINT 'SECOND ARRAY:'
PRINT B
PRINT '\N'

PRINT 'FIRST ARRAY + SECOND ARRAY'
PRINT A + B
```

TRANSPOSE OF A MATRIX

- Rows become columns and columns become rows in multi dimensional arrays.

```
a = np.arange(0,60,5)
a = a.reshape(3,4)

print 'Original array is:'
print a
print '\n'

print 'Transpose of the original array is:'
b = a.T
print b
print '\n'
```

OPERATIONS ON ARRAY

- Resize
- Append
- Insert
- Delete
- unique

Operations continued

- Bitwise and
- invert
- Bitwise or
- Shift right
- Shift left