Numpy for Machine Learning

January 13, 2022

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

[2]:
```
arr = np.array([[1,2,3],[5,9,4]])
arr.shape
```

[2]:
```
(2, 3)
```

0.1 To get specific row and cloumn values ,also change

Row : Syntax arr[1,:]
Column : Syntax arr[:,0]

[3]:
```
arr[1,:]
```

[3]:
```
array([5, 9, 4])
```

[4]:
```
arr[:,0]
arr
```

[4]:
```
array([[1, 2, 3],
       [5, 9, 4]])
```

[5]:
```
arr[:,2] = 1
arr
```

[5]:
```
array([[1, 2, 1],
       [5, 9, 1]])
```

0.2 Initialize different types of array in Numpy

Resources

(https://numpy.org/doc/stable/reference/routines.array-creation.html)

Creating matrix of particular number of different shape and size.

[6]:
```
# A maatrix of zeros 1D
np.zeros(4)
```

[6]:
```
array([0., 0., 0., 0.])
```

[7]:
```
# A maatrix of zeros 2D
np.zeros((2,5))
```

[7]:
```
array([[0., 0., 0., 0., 0.],
       [0., 0., 0., 0., 0.]])
```

[8]:
```
# Same for one also
np.ones((2,3))
```

[8]:
```
array([[1., 1., 1.],
       [1., 1., 1.]])
```

[9]:
```
# Any other number
np.full((2,5) , 100) # (shape , value)
```

[9]:
```
array([[100, 100, 100, 100, 100],
       [100, 100, 100, 100, 100]])
```

[10]:
```
#Repeating an Array
arra = np.array([[1,2,3]])
print(np.repeat(arra , 3 , axis = 0))
print(np.repeat(arra , 3 , axis = 1))
```

```
[[1 2 3]
 [1 2 3]
 [1 2 3]]
[[1 1 1 2 2 2 3 3 3]]
```

Random in Numpy

[11]:
```
# Random decimal and integer numbers
print(np.random.rand(2,5))
print(np.random.randint(7,10, size = (2,5)))
```

```
[[0.27590366 0.92227258 0.50946775 0.5077942  0.16836195]
 [0.83112557 0.39463817 0.99534666 0.55551744 0.5411929 ]]
[[9 8 7 8 9]
 [7 8 7 7 8]]
```

Matrices properties

[12]:
```
#identity Matrix
np.identity(3)
```

[12]:
```
array([[1., 0., 0.],
       [0., 1., 0.],
       [0., 0., 1.]])
```

0.3 Mathematics in Numpy

[13]:
```
ary = np.array([1,2,3,4,5] , dtype = 'int64')
#Operations
print(ary+2 , ary - 2 , ary *2 , ary /2 , arr**2, sep=" ")
```

```
[3 4 5 6 7] [-1  0  1  2  3] [ 2  4  6  8 10] [0.5 1.  1.5 2.  2.5] [[ 1  4  1]
 [25 81  1]]
```

Trignometry

(https://numpy.org/doc/stable/reference/routines.math.html)

[14]:
```
values = np.array([0 , 30 , 45 , 60 , 90])
print(np.sin(values) , np.cos(values), np.tan(values), sep = " | ")
```

```
[ 0.         -0.98803162  0.85090352 -0.30481062  0.89399666] | [ 1.
0.15425145  0.52532199 -0.95241298 -0.44807362] | [ 0.         -6.4053312
1.61977519  0.32004039 -1.99520041]
```

Linear Algebra

(https://numpy.org/doc/stable/reference/routines.linalg.html)

Matrix Multiplication

[15]:
```
m1 = np.ones((2,3))
m2 = np.full((3,2) , 3)
print(m1,m2,sep="\n")
print(np.matmul(m1,m2))
```

```
[[1. 1. 1.]
 [1. 1. 1.]]
[[3 3]
 [3 3]
 [3 3]]
[[9. 9.]
 [9. 9.]]
```

In above example what happening :

```
    [[1,1,1]    [[3,3]                [[3+3+3  3+3+3]
    [1,1,1]]    [3,3]]       ----->    [3+3+3   3+3+3]]
```

Find the Determinant

[16]:
```
m3 = np.identity(3)
np.linalg.det(m3)
```

[16]: 1.0

1 Statistics

[17]:
```
arr
```

[17]:
```
array([[1, 2, 1],
       [5, 9, 1]])
```

[18]:
```
# axis = 1 give max and min of each array in 2-D array by default give max and␣
 ↪min of whole array
print(np.min(arr , axis = 1) , np.max(arr , axis = 1) , np.mean(arr , axis = 1␣
 ↪, dtype='int64') , sep = ",")
```

```
[1 1],[2 9],[1 5]
```

[19]:
```
#Default give sum whole 2-D array
# Specifing axis = 1 give sum of each array in 2-D array
# Specifing axis = 0 give sum of each column in 2-D array
p = np.sum(arr)
q = np.sum(arr , axis = 0)
r = np.sum(arr , axis = 1)
print(p,q,r,sep=',')
```

```
19,[ 6 11  2],[ 4 15]
```

1.1 Load Data from File

[21]:
```
filedata = np.genfromtxt('data.txt', delimiter=',')
filedata = filedata.astype('int32')
print(filedata)
```

```
[[  1  13  21  11 196  75   4   3  34   6   7   8   0   1   2   3   4   5]
 [  3  42  12  33 766  75   4  55   6   4   3   4   5   6   7   0  11  12]
 [  1  22  33  11 999  11   2   1  78   0   1   2   9   8   7   1  76  88]]
```

1.2 Boolean Masking and Advanced Indexing

[22]:
```
(~((filedata > 50) & (filedata < 100)))
```

[22]:
```
array([[ True,  True,  True,  True,  True, False,  True,  True,  True,
         True,  True,  True,  True,  True,  True,  True,  True,  True],
       [ True,  True,  True,  True,  True, False,  True, False,  True,
         True,  True,  True,  True,  True,  True,  True,  True,  True],
       [ True,  True,  True,  True,  True,  True,  True,  True, False,
         True,  True,  True,  True,  True,  True,  True, False, False]])
```

[ ]:
```

```