AIML_Tutorial_1_numpy

December 30, 2023

0.1 # Tutorial 1 - Numpy

Name: Thanki Naman Rajeshbhai

Roll Number / Enrolment No: 40

Division: BTech - 6CEE

Subject: Artificial Intelligence and Machine Learning

1. Import NumPy library.

[2]:
```
import numpy as np
```

2. Create 1D array with 5 elements

[10]:
```
a = np.array([ 1, 2, 3, 4, 5 ])
print(a)
```

```
[1 2 3 4 5]
```

3. Create 1D array with 5 elements random value. (by default values between 0 to 1)

[11]:
```
b = np.random.rand(5)
print(b)
```

```
[0.00178403 0.49082323 0.92076552 0.50856792 0.96604445]
```

4. Generate random value from range 5 to 15 with 1D array size 10.

[12]:
```
c = np.random.randint(5, 15, 10)
print(c)
```

```
[ 5  7  6 11  6  7  9  5  5 11]
```

5. Create 2D array - 4 by 3 size.

[13]:
```
d = np.array([[1, 2, 3,], [4, 5, 6], [7, 8, 9], [10, 11, 12]])
print(d)
```

```
[[ 1  2  3]
 [ 4  5  6]
 [ 7  8  9]
 [10 11 12]]
```

6. Create 2D array with size 3 by 3. Elements values are random number(by default values between 0 to 1)

[14]:

```
e = np.random.rand(3, 3)
print(e)
```

```
[[0.62529786 0.47441765 0.65395844]
 [0.38513849 0.13745355 0.78001766]
 [0.17058914 0.91254207 0.34455203]]
```

7. Generate random value from range 5 to 15 with 2D array size 3 by 3.

[8]:

```
f = np.random.randint(5, 16, size=(3, 3))
print(f)
```

```
[[14  8 15]
 [13 13 14]
 [13  8 14]]
```

8. Create 3D array with size 2 * 3 * 2.

[15]:

```
e = np.array([[[1, 2], [3, 4], [5, 6]], [[7, 8], [9, 10], [11, 12]]])

print(e)
```

```
[[[ 1  2]
  [ 3  4]
  [ 5  6]]

 [[ 7  8]
  [ 9 10]
  [11 12]]]
```

9. Initialize 5 by 5 array with all values are zeros.

[16]:

```
f = np.zeros((5, 5), dtype=int)
print(f)
```

```
[[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]]
```

10. Initialize 5 by 5 array with all values are ones.

[17]:

```
ones = np.ones((5, 5), dtype=int)
print(ones)
```

```
[[1 1 1 1 1]
 [1 1 1 1 1]
 [1 1 1 1 1]
 [1 1 1 1 1]
 [1 1 1 1 1]]
```

11. Initialize 5 by 5 array with all values are particular values (consider 4).

[18]:

```
fours = np.full((5, 5), 4, dtype=int)
print(fours)
```

```
[[4 4 4 4 4]
 [4 4 4 4 4]
 [4 4 4 4 4]
 [4 4 4 4 4]
 [4 4 4 4 4]]
```

12. Create identity matrix with all diagonal values are 1 and rest all values are zeros.

[19]:

```
identity_mat = np.eye(5, 5, dtype=int)
print(identity_mat)
```

```
[[1 0 0 0 0]
 [0 1 0 0 0]
 [0 0 1 0 0]
 [0 0 0 1 0]
 [0 0 0 0 1]]
```

13. Find out shape of array.

[20]:

```
shape = identity_mat.shape
print(shape)
```

```
(5, 5)
```

14. Find dimension of array.

[21]:

```
dimension = identity_mat.ndim
print(dimension)
```

```
2
```

15. Find no. of elements from above any array

[22]:

```
elements_count = identity_mat.size
print(elements_count)
```

```
25
```

16. Find type of array.

[23]:

```
typearr = identity_mat.dtype
print(typearr)
```

```
int64
```

17. Find maximum value from array.

[24]:

```
max = identity_mat.max()
print(max)
```

```
1
```

18. Find minimum value from array.

[25]:

```
min = identity_mat.min()
print(min)
```

```
0
```

19. Find average (mean) values from array.

[26]:

```
avg = identity_mat.mean()
print(avg)
```

```
0.2
```

20. Find location of maximum value from arr

[27]:

```
max_location = identity_mat.argmax()
print(max_location)
```

```
0
```

21. Create two 2D array of size 3 by 3 and perform four basic mathematical operations

[32]:

```
mat_a = np.array([[1, 2, 3], [3, 4, 5], [5, 6, 7]])
mat_b = np.array([[7, 6, 8], [8, 9, 10], [10, 11, 12]])
# print(mat_a)
# print(mat_b)

sum = np.add(mat_a, mat_b)
print("Sum")
print(sum)

subtract = np.subtract(mat_a, mat_b)
print("Subtract")
print(subtract)

multiply = np.multiply(mat_a, mat_b)
print("Multiply")
print(multiply)

divide = np.divide(mat_a, mat_b)
print("Divide")
print(divide)
```

```
Sum
[[ 8  8 11]
 [11 13 15]
 [15 17 19]]
Subtract
[[-6 -4 -5]
 [-5 -5 -5]
 [-5 -5 -5]]
Multiply
[[ 7 12 24]
 [24 36 50]
 [50 66 84]]
Divide
[[0.14285714 0.33333333 0.375     ]
 [0.375      0.44444444 0.5       ]
 [0.5        0.54545455 0.58333333]]
```

22. Print index 2 value (indexing)

[29]:
```
x = np.array([1, 2, 3, 4, 5, 6, 7, 8, 9, 10])
print(x[2])
```

```
3
```

23. Print value of index 2 to 5 (slicing)

[33]:
```
print(x[2:6])
```

```
[3 4 5 6]
```

24. Print from index 0 to 5 values

[34]:
```
print(x[:6])
```

```
[1 2 3 4 5 6]
```

25. Print from index 5 to all

[35]:
```
print(x[5:])
```

```
[ 6  7  8  9 10]
```

26. Reassign value for particular index.

[36]:
```
x[9] = 0
print(x)
```

```
[1 2 3 4 5 6 7 8 9 0]
```

27. Extract elements from index 1 to 7 with a step of 2.

[38]:

```
print(x[1:8:2])
```

```
[2 4 6 8]
```

28. Print Reverse the array.

[39]:

```
print(x[::-1])
```

```
[0 9 8 7 6 5 4 3 2 1]
```