Copy of NumPy

August 12, 2023

[34]:
```
import numpy as np
print(np.__version__)
import matplotlib.pylab as plt
```

```
1.22.4
```

1 Numpy Basics:

Exercise #1: Creating a numpy array with particular value

Your Task Create a numpy array with size 2*3 with all elements as 5.

[35]:
```
# YOUR CODE STARTS HERE
arr=np.ones((2,3))*5
arr=np.full((2,3), 5)
# YOUR CODE ENDS HERE
print(arr)
```

```
[[5 5 5]
 [5 5 5]]
```

Exercise #2: Creating a numpy array with *random* value

Your Task Create a numpy array with size 2*3 with all random values

[36]:
```
# YOUR CODE STARTS HERE
rand_arr=np.random.randn(2,3)
# YOUR CODE ENDS HERE
print(rand_arr)
```

```
[[ 1.86668311 -1.34104166 -0.99525942]
 [ 1.21324819 -0.96427704 -1.14707812]]
```

Exercise #3: Basic NumPy operatrions

Your Task For the random array `arr`defined above, find the `sum` of all the elements, `mean`, `maximum` and `minimum` value

[37]:
```
#YOUR CODE STARTS HERE
arr_sum, arr_mean, arr_max, arr_min= np.sum(rand_arr), np.mean(rand_arr), np.
  ↪max(rand_arr), np.min(rand_arr)
```

```
#YOUR CODE ENDS HERE

print(f'sum: {arr_sum}\nmean: {arr_mean}\nmax: {arr_max}\nmin:{arr_min}')
```

```
sum: -1.3677249428948768
mean: -0.22795415714914613
max: 1.866683113603748
min:-1.3410416574045112
```

Exercise #4: `argmin` and `argmax`

Your Task For the random array `arr`defined below:

- Find the position of maximum and minimum value of above array
- Find the indices of maximum and minimum value along each of its columns.
- Find the indices of maximum and minimum value along each of the its rows.

[38]:
```
# Define an array
arr = np.array([[5,12,51,25] ,[25,29,2,27]])
print(f'Array:\n{arr}\n')
# YOUR CODE STARTS HERE
#Find the position of maximum and minimum value of above array
max_pos, min_pos= np.argmax(arr), np.argmin(arr)
print(f"max_pos: {max_pos} min_pos: {min_pos}\n")
#Find the indices of maximum and minimum value along each of its columns.
for c in range(arr.shape[1]):
  temp_arr=arr[:,c]
  max_id, min_id=np.argmax(temp_arr), np.argmin(temp_arr)
  print(f"for column: {c} max_idx: {(max_id, c)} min_idx: {min_id, c}")
print()
#Find the indices of maximum and minimum value along each of the its rows.
for r in range(arr.shape[0]):
  temp_arr=arr[r,:]
  max_id, min_id=np.argmax(temp_arr), np.argmin(temp_arr)
  print(f"for row: {r} max_idx: {(r, max_id)} min_idx: {r, min_id}")

#YOUR CODE ENDS HERE
```

```
Array:
[[ 5 12 51 25]
 [25 29  2 27]]

max_pos: 2 min_pos: 6

for column: 0 max_idx: (1, 0) min_idx: (0, 0)
for column: 1 max_idx: (1, 1) min_idx: (0, 1)
for column: 2 max_idx: (0, 2) min_idx: (1, 2)
for column: 3 max_idx: (1, 3) min_idx: (0, 3)
for row: 0 max_idx: (0, 2) min_idx: (0, 0)
for row: 1 max_idx: (1, 1) min_idx: (1, 2)
```

Numpy Arrays vs. Python Lists?

```
Why the need for numpy arrays? Can't we just use Python lists?
Iterating over numpy arrays is slow. Slicing is faster
```

Python lists may contain items of different types. This flexibility comes at a price: Python lists store pointers to memory locations. On the other hand, numpy arrays are typed, where the default type is floating point. Because of this, the system knows how much memory to allocate, and if you ask for an array of size 100, it will allocate one hundred contiguous spots in memory, where the size of each spot is based on the type. This makes access extremely fast.

If you want to know more, we will suggest that you read this from Jake Vanderplas's Data Science Handbook. You will find that book an incredible resource for learning numpy, pandas and plotting. All the lessons are available as colab notebooks so that u can try out the commands on your own.