NumPy

Sunday, November 15, 2020 7:56 PM

Possibly the most important library in python! It's ~50x faster than using lists or loops

NumPy is ….
We use NumPy by first importing it
- import numpy as np

np.array([…])

linearly spaced array
np.linspaced(0,10,5) will create an array between 0 and 10 with 5 equally spaced elements
- [0, 2.5, 5, 7.5, 10]
- note that the start and stop are inclusive

regularly spaced array
np.arange(0,10,2) will create an array between 0 and 10 with elements spaced 2 apart
- [0,2,4,6,8]
- note that start is inclusive, stop is exclusive

create a random array
np.random.randint(0, 100, size=(2,3)) will create a 2 row X 3 column array filled with random numbers between 0 and 99 (stop not inclusive)

concatenate arrays
np.concatenate(array1, array2, axis=0) will concatenate array1 and array2 along rows while axis 1 would do so along columns

NumPy allows us to broadcast an operation through all members of an array
- we could do this with for loops and lists in python, but it's much easier if we use NumPy

Lots of descriptive statistics
np.sum() will sum entire array
np.sum(axis=1) will sum along rows
we have min(), max(), std(), variance(), mean(), cumsum() and more!
- cumsum() adds each row with previous value

Don't know the size of the array? Use shape() It will return the dimensions

Cool math stuff
np.sqrt()
np.exp()
np.log()
np.sin()
np.tan()
np.tanh()
np.dot(array1, array1.T) is the dot product of array1 and the transpose of array1
- you can also just do array1 @ array2

np.corrcoef() will return correlation coefficient
and more!

you can also use LOGIC with these arrays
logic = array >= 80 will return an array with same dimensions as original array but filled with True or False values where cells are greater than or equal to 80
you could then do an operation on just those true cells, for example make them equal to 99
array[logic] = 99

We can visualize slicing and indexing by looking at an image which is itself an array
from skimage import io
photo = io.imread('scout.png')

Heads up with copy() vs view()
- copy will create a new copy but view will point to original so changes made to new version in view will affect original

reshape() allows us to reshape dimensions of an array. Make sure the rows*columns equals the number of elements in your initial array. You can also do this in conjunction with transpose