

Machine Learning Laboratory

(410302)

BE Sem I Honors in AI/ML

Academic Year: 2021-22

Lab Assignment No.2

Name: Aboli Marathe

Roll Number: 41301

Branch: Department of Computer Engineering

▼ NumPy Exercises

Now that we've learned about NumPy let's test your knowledge. We'll start off with a few simple tasks and then you'll be asked some more complicated questions.

IMPORTANT NOTE! Make sure you don't run the cells directly above the example output shown, otherwise you will end up writing over the example output!

▼ 1. Import NumPy as np

```
import numpy as np
```

▼ 2. Create an array of 10 zeros

```
# CODE HERE
array=np.zeros(10)
array

array([0., 0., 0., 0., 0., 0., 0., 0., 0., 0.])

# DON'T WRITE HERE

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

▼ 3. Create an array of 10 ones

```
array=np.ones(10)
array

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

```
# DON'T WRITE HERE
```

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

▼ 4. Create an array of 10 fives

```
array=np.ones(10)*5
array

array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])
```

```
# DON'T WRITE HERE
```

```
array([5., 5., 5., 5., 5., 5., 5., 5., 5., 5.])
```

▼ 5. Create an array of the integers from 10 to 50

```
array=np.arange(10,51)
array

array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
       27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
       44, 45, 46, 47, 48, 49, 50])
```

```
# DON'T WRITE HERE
```

```
array([10, 11, 12, 13, 14, 15, 16, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26,
       27, 28, 29, 30, 31, 32, 33, 34, 35, 36, 37, 38, 39, 40, 41, 42, 43,
       44, 45, 46, 47, 48, 49, 50])
```

▼ 6. Create an array of all the even integers from 10 to 50

```
array=np.arange(10,51,2)
array

array([10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42,
       44, 46, 48, 50])
```

```
# DON'T WRITE HERE
```

```
array([10, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34, 36, 38, 40, 42,
       44, 46, 48, 50])
```

▼ 7. Create a 3x3 matrix with values ranging from 0 to 8

```
array = np.arange(0,9).reshape(3,3)
array
```

```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

```
# DON'T WRITE HERE
```

```
array([[0, 1, 2],
       [3, 4, 5],
       [6, 7, 8]])
```

▼ 8. Create a 3x3 identity matrix

```
array = np.identity(3)
array
```

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

```
# DON'T WRITE HERE
```

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

9. Use NumPy to generate a random number between 0 and 1

▼

NOTE: Your result's value should be different from the one shown below.

```
num = np.random.normal(0,1,1)
num
```

```
array([0.70895154])
```

```
# DON'T WRITE HERE
```

```
array([0.65248055])
```

10. Use NumPy to generate an array of 25 random numbers sampled from a standard normal distribution

NOTE: Your result's values should be different from the ones shown below.

```
array = np.random.normal(0,1,25)
array

array([ 0.71604242, -1.29996805,  1.06013943,  1.09209567,  0.29555709,
        -2.20524194,  0.84833508, -0.46113199,  1.14391578,  1.02391418,
        -2.55266574,  0.74500945,  1.32430336, -0.73850128,  2.80196881,
         0.13774943,  0.64650434,  0.60355817,  1.03935338, -1.10196514,
        -0.09899275,  0.21652005,  1.02909358, -0.92819524, -1.29948475])
```

DON'T WRITE HERE

```
array([ 1.80076712, -1.12375847, -0.98524305,  0.11673573,  1.96346762,
        1.81378592, -0.33790771,  0.85012656,  0.0100703 , -0.91005957,
         0.29064366,  0.69906357,  0.1774377 , -0.61958694, -0.45498611,
        -2.0804685 , -0.06778549,  1.06403819,  0.4311884 , -1.09853837,
         1.11980469, -0.48751963,  1.32517611, -0.61775122, -0.00622865])
```

▼ 11. Create the following matrix:

```
array = np.arange(0.01,1.01,0.01).reshape(10,10)
array

array([[0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1 ],
       [0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2 ],
       [0.21, 0.22, 0.23, 0.24, 0.25, 0.26, 0.27, 0.28, 0.29, 0.3 ],
       [0.31, 0.32, 0.33, 0.34, 0.35, 0.36, 0.37, 0.38, 0.39, 0.4 ],
       [0.41, 0.42, 0.43, 0.44, 0.45, 0.46, 0.47, 0.48, 0.49, 0.5 ],
       [0.51, 0.52, 0.53, 0.54, 0.55, 0.56, 0.57, 0.58, 0.59, 0.6 ],
       [0.61, 0.62, 0.63, 0.64, 0.65, 0.66, 0.67, 0.68, 0.69, 0.7 ],
       [0.71, 0.72, 0.73, 0.74, 0.75, 0.76, 0.77, 0.78, 0.79, 0.8 ],
       [0.81, 0.82, 0.83, 0.84, 0.85, 0.86, 0.87, 0.88, 0.89, 0.9 ],
       [0.91, 0.92, 0.93, 0.94, 0.95, 0.96, 0.97, 0.98, 0.99, 1.  ]])
```

DON'T WRITE HERE

```
array([[0.01, 0.02, 0.03, 0.04, 0.05, 0.06, 0.07, 0.08, 0.09, 0.1 ],
       [0.11, 0.12, 0.13, 0.14, 0.15, 0.16, 0.17, 0.18, 0.19, 0.2 ],
       [0.21, 0.22, 0.23, 0.24, 0.25, 0.26, 0.27, 0.28, 0.29, 0.3 ],
       [0.31, 0.32, 0.33, 0.34, 0.35, 0.36, 0.37, 0.38, 0.39, 0.4 ],
       [0.41, 0.42, 0.43, 0.44, 0.45, 0.46, 0.47, 0.48, 0.49, 0.5 ],
       [0.51, 0.52, 0.53, 0.54, 0.55, 0.56, 0.57, 0.58, 0.59, 0.6 ],
       [0.61, 0.62, 0.63, 0.64, 0.65, 0.66, 0.67, 0.68, 0.69, 0.7 ],
       [0.71, 0.72, 0.73, 0.74, 0.75, 0.76, 0.77, 0.78, 0.79, 0.8 ],
       [0.81, 0.82, 0.83, 0.84, 0.85, 0.86, 0.87, 0.88, 0.89, 0.9 ],
       [0.91, 0.92, 0.93, 0.94, 0.95, 0.96, 0.97, 0.98, 0.99, 1.  ]])
```

▼ 12. Create an array of 20 linearly spaced points between 0 and 1:

```
array = np.linspace(0, 1, 20)
array

array([0.          , 0.05263158, 0.10526316, 0.15789474, 0.21052632,
       0.26315789, 0.31578947, 0.36842105, 0.42105263, 0.47368421,
       0.52631579, 0.57894737, 0.63157895, 0.68421053, 0.73684211,
       0.78947368, 0.84210526, 0.89473684, 0.94736842, 1.          ])
```

```
# DON'T WRITE HERE
```

```
array([0.          , 0.05263158, 0.10526316, 0.15789474, 0.21052632,
       0.26315789, 0.31578947, 0.36842105, 0.42105263, 0.47368421,
       0.52631579, 0.57894737, 0.63157895, 0.68421053, 0.73684211,
       0.78947368, 0.84210526, 0.89473684, 0.94736842, 1.          ])
```

▼ Numpy Indexing and Selection

Now you will be given a starting matrix (be sure to run the cell below!), and be asked to replicate the resulting matrix outputs:

```
# RUN THIS CELL - THIS IS OUR STARTING MATRIX
mat = np.arange(1,26).reshape(5,5)
mat
```

```
array([[ 1,  2,  3,  4,  5],
       [ 6,  7,  8,  9, 10],
       [11, 12, 13, 14, 15],
       [16, 17, 18, 19, 20],
       [21, 22, 23, 24, 25]])
```

13. Write code that reproduces the output shown below.

▼ Be careful not to run the cell immediately above the output, otherwise you won't be able to see the output any more.

```
# CODE HERE
mat[2:,1:]

array([[12, 13, 14, 15],
       [17, 18, 19, 20],
       [22, 23, 24, 25]])
```

```
# DON'T WRITE HERE
```

```
array([[12, 13, 14, 15],
       [17, 18, 19, 20],
       [22, 23, 24, 25]])
```

▼ 14. Write code that reproduces the output shown below.

```
mat[3,4]
```

```
20
```

```
# DON'T WRITE HERE
```

```
20
```

▼ 15. Write code that reproduces the output shown below.

```
array = mat[0:3,1].reshape(3,1,)
```

```
array
```

```
array([[ 2],  
       [ 7],  
       [12]])
```

```
# DON'T WRITE HERE
```

```
array([[ 2],  
       [ 7],  
       [12]])
```

▼ 16. Write code that reproduces the output shown below.

```
mat[4,:]
```

```
array([21, 22, 23, 24, 25])
```

```
# DON'T WRITE HERE
```

```
array([21, 22, 23, 24, 25])
```

▼ 17. Write code that reproduces the output shown below.

```
mat[3:,:]
```

```
array([[16, 17, 18, 19, 20],  
       [21, 22, 23, 24, 25]])
```

```
# DON'T WRITE HERE
```

```
array([[16, 17, 18, 19, 20],  
       [21, 22, 23, 24, 25]])
```

▼ NumPy Operations

▼ 18. Get the sum of all the values in mat

```
np.sum(mat)
```

```
325
```

```
# DON'T WRITE HERE
```

```
325
```

▼ 19. Get the standard deviation of the values in mat

```
np.std(mat)
```

```
7.211102550927978
```

```
# DON'T WRITE HERE
```

```
7.211102550927978
```

▼ 20. Get the sum of all the columns in mat

```
np.sum(mat,axis=0)
```

```
array([55, 60, 65, 70, 75])
```

```
# DON'T WRITE HERE
```

```
array([55, 60, 65, 70, 75])
```

Great Job!

✓ 0s completed at 10:28 PM

● ✕