Nama : Ghina Khoerunnisa
Program : Data Scientist Intern

Numpy

- **Introduction**
  Numpy merupakan standar universal untuk bekerja dengan data numerik menggunakan python. Array numpy lebih cepat dan compact daripada list python. Array adalah grid of values dan berisi informasi tentang data mentah, cara menemukan elemen, dan cara menginterpretasikan elemen. Dalam numpy, dimensi disebut dengan axes. Jika axis = 0, ini berarti baris. Jika axis = 1, ini berarti kolom.
- **Create Array**
  Untuk membuat array numpy dapat menggunakan fungsi : np.array(), membuat array yang sudah terinisialisasi nilainya dapat menggunakan : np.ones() -> berisi nilai 1, np.zeros() -> berisi nilai 0. np.empty() -> empty array, np.arange() -> berisi nilai dengan rentang tertentu, np.random.random() -> berisi nilai float [0.0,1.0] secara random.
- **Add, Remove, Sort**
  Untuk menambah elemen ke array dapat menggunakan: np.append(array, elemen), untuk menghapus elemen di posisi tertentu dapat menggunakan np.delete(array, posisi), untuk mengurutkan elemen-elemen di array dapat menggunakan np.sort().
- **Ndim, Shape, Size**
  Untuk melihat jumlah axes / dimensi array dapat menggunakan ndarray.ndim, untuk melihat jumlah total elemen array dapat menggunakan ndarray.size, untuk melihat jumlah elemen yang disimpan di sepanjang setiap dimensi array dapat menggunakan ndarray.shape.
- **Reshape**
  np.reshape() digunakan untuk mengubah bentuk array tanpa mengubah datanya dengan syarat jumlah elemennya sama dengan array asli.
- **Convert 1D to 2D**
  Terdapat dua cara untuk meningkatkan dimensi array yang sudah ada yaitu, np.newaxis dan np.expand_dims. Dengan menggunakan np.newaxis akan meningkatkan dimensi array sebesar satu dimensi. Dengan menggunakan np.expand_dims dapat memperluas array dengan memasukkan axis baru pada posisi yang ditentukan.
- **Indexing & Slicing**
  Indexing dan slicing pada array sama seperti pada python list.

```
num = np.arange(1,11)
sample = np.array([1,2,3,4]).reshape((2,2))
print("num:",num)
print(num[8:])
print(num[-2:])
print(num[:2])
print(num[num>2])
print(num[num%2==0])
print("sample",sample)
print(sample[:,0])
print(sample[0,1])
```

```
num: [ 1  2  3  4  5  6  7  8  9 10]
[ 9 10]
[ 9 10]
[1 2]
[ 3  4  5  6  7  8  9 10]
[ 2  4  6  8 10]
sample [[1 2]
 [3 4]]
[1 3]
2
```

- **Create Array from Existing Data**

Membuat array dari data yang telah ada dapat dilakukan dengan Indexing & Slicing, np.vstack() menumpuk dua array secara vertikal, np.hstack() menumpuk array secara horizontal, np.hsplit() membagi array menjadi beberapa array yang lebih kecil. ndarray.view() untuk membuat objek array baru yang sama dengan array asli. ndarray.copy() sama seperti view tapi objek baru tidak akan mempengaruhi array asli.

- **Array Operation**

```
data = np.random.randint(100,size=(3,3))
ones = np.ones((3,3))
print("data: \n",data)
print("ones: \n",ones)
print("Add: \n", data+ones)
print("Sub: \n", data-ones)
print("Mul: \n", data*ones)
print("Div: \n", data/ones)
```

```
data:
 [[45 69 31]
 [79 24 97]
 [34 30 88]]
ones:
 [[1. 1. 1.]
 [1. 1. 1.]
 [1. 1. 1.]]
Add:
 [[46. 70. 32.]
 [80. 25. 98.]
 [35. 31. 89.]]
Sub:
 [[44. 68. 30.]
 [78. 23. 96.]
 [33. 29. 87.]]
Mul:
 [[45. 69. 31.]
 [79. 24. 97.]
 [34. 30. 88.]]
Div:
 [[45. 69. 31.]
 [79. 24. 97.]
 [34. 30. 88.]]
```

```
print("broadcasting: \n", data*1.5)
print("max: \n",data.max())
print("min: \n",data.min())
print("sum: \n",data.sum())
print("mean: \n",data.mean())
print("std: \n",data.std())
```

```
broadcasting:
 [[ 67.5 103.5  46.5]
 [118.5  36.  145.5]
 [ 51.   45.  132. ]]
max:
 97
min:
 24
sum:
 497
mean:
 55.22222222222222
std:
 26.515311206830326
```

- **Matrices**

  Membuat matriks dengan numpy sama seperti dengan create array asalkan dimensi yang dimasukkan mendeskripsikan matriks. Operasi aritmatika (+,-,*,/)pada matriks dapat dilakukan jika kedua matriks berukuran sama. Untuk operasi dot product di matriks dapat dilakukan dengan menggunakan metode dot(). Untuk mengakses elemen di matriks dapat menggunakan indexing dan slicing.

- **Transposing**

  Untuk mentranspose matriks / mengubah row menjadi kolom dan sebaliknya dapat menggunakan np.transpose atau pada numpy array terdapat properti T yang fungsinya sama.

- **Flatten N-Dimensional Array**

  Untuk meratakan array dapat menggunakan ndarray.flatten() atau ndarray.ravel(). Perbedaan keduanya adalah ravel merupakan referensi ke array induk sehingga perubahan yang terjadi akan mempengaruhi array induknya.