Numpy

Numpy

- List에서 계산을 수행하는 방법
- Numeric python
- Numpy Array : alternative to python list and also can calculate without error
- Contain only one type (같은 유형으로 바꿔서 배열)

Numpy subsetting

- ' : '를 이용해서 범위 정하기
- 끝 숫자는 항상 +1

2D Numpy arrays

- 행과 열이 있는 numpy arrays
- 단일 유형만 포함

▼ Row & Column

- Row : 행, 가로
- Column : 열, 세로
- [행, 열]
- [행, 열]
- ' : ' 전체 범위를 뜻함

np_2d.shape

- 행과 열의 개수를 나타냄

np.mean

- 평균 계산

np.median

- 중앙값

np.corrcoef

- 값을 두개씩 묶어줌
- ' , '로 묶어줄 arrays 구별

np.std

- 표준편차

> 💡 단일 데이터 유형을 적용하기때문에 파이썬 list보다 속도가 매우 빠름

---

np.arange

- 순차적인 array 생성 (범위를 넣으면 범위 속 숫자들이 순차적으로 정렬되어 나옴)
- (시작 숫자, 끝숫자) : 시작 숫자 ~ 끝숫자 -1

```
np.arange(1,11)
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

- (start = 시작, stop = 끝) : 시작 ~ 끝 -1

```
np.arange(start=1, stop=11)
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

- (start = 시작, stop = 끝, step = 차이) : 시작부터 끝까지 step 숫자만큼 등차적으로 배열

```
np.arange(start=1, stop=11, step=2)
```

```
array([1, 3, 5, 7, 9])
```

✔️ 끝숫자를 앞에 적더라도 순차적으로 배열됨

```
np.arange(stop=11, start=1)
```

```
array([ 1,  2,  3,  4,  5,  6,  7,  8,  9, 10])
```

np.sort

- 행렬의 정렬
- 오름차순 (default), 내림차순 ('[::-1]' 붙이면 됨)

```
np.sort(arr)[::-1]
```

```
array([10,  9,  8,  8,  7,  6,  5,  4,  3,  2,  1])
```

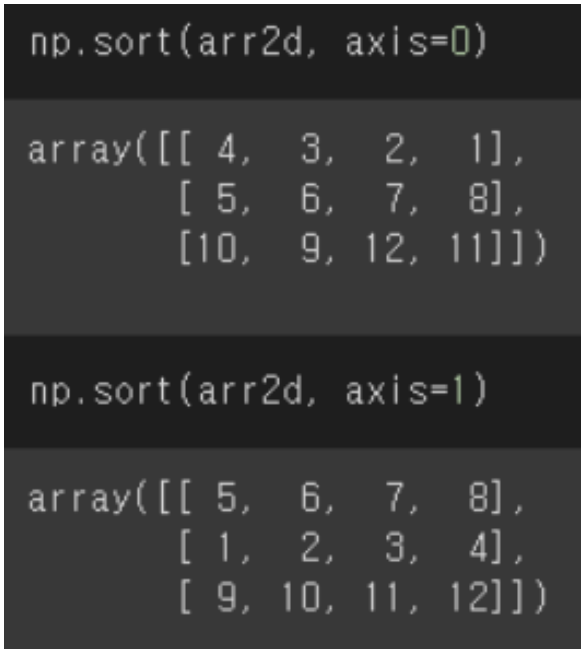
▼ 2D sort 함수

- 같은 열 안에서만 오름차순으로 정렬 (default)

```
np.sort(arr2d)
```

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

- axis = 0 : 행을 의미하며, 같은 열끼리 작업 (같은 열 안에서 오름차순으로 정렬)
- axis = 1 : 열을 의미하는데 같은 행끼리 작업 (같은 행 안에서 오름차순으로 정렬)



✔️ 2차원 배열 : (행, 열) - index
axis = 0 : 행
axis = 1 : 열

np.argsort

- 정렬 시 index를 반환하는 모듈 (오름차순으로 index 매겨짐)
- axis = 0 : 행으로 따질때 index
- axis = 0 : 열로 따질때 index

```
arr2d = np.array([[5, 6, 7, 8],
                  [4, 3, 2, 1],
                  [10, 9, 12, 11]])
```

```
np.argsort(arr2d,axis=0)
```

```
array([[1, 1, 1, 1],
       [0, 0, 0, 0],
       [2, 2, 2, 2]])
```

```
np.argsort(arr2d, axis=1)
```

```
array([[0, 1, 2, 3],
       [3, 2, 1, 0],
       [1, 0, 3, 2]])
```

❓ 행으로 따질때 만약 [1,2,3,7], [5,6,8,9]라면 가장 작은 숫자(1, 5)로 순서를 따지고 행이 0인지 1인지 매겨지는 것?

np.sum

- 전체를 더한 값
- axis = 0 : 같은 열끼리 더한 값
- axis = 1 : 같은 행끼리 더한값

```
a = np.array([[1, 2, 3],
              [2, 3, 4]])
```

```
np.sum(a)
```

```
15
```

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

```
array([3, 5, 7])
```

```
np.sum(a, axis=1)
```

```
array([6, 9])
```

- 행렬간 사칙연산, 일괄적인 값으로 사칙연산 모두 가능하다

> ✔️ 행렬간 계산을 할때는 열과 행의 숫자가 같아야한다

np.dot

- 행렬간 곱셈

```
a = np.array([[1, 2, 3],
              [2, 3, 4]])
```

```
b = np.array([[3, 4, 5],
              [1, 2, 3]])
```

```
a * b
```

```
array([[ 3,  8, 15],
       [ 2,  6, 12]])
```

```
c = np.array([[1, 2],
              [3, 4],
              [5, 6]])
```

```
np.dot(a,c)
```

```
array([[22, 28],
       [31, 40]])
```

❓ 행렬 간 열과 행의 숫자가 맞아야된다고 알고 있는데 달라도 계산 가능한 이유와 곱셈이 어떻게 해서 저런 값이 나오는 지