11. N차원 배열 다루기 #src: 8_Python/ch11_N차원배열 참고

1) NumPy 패키지; 파이썬을 사용한 과학 컴퓨팅의 기본 패키지 #https://numpy.org/

(1) 넘파이 주요 함수

용도	넘파이 주요 함수
배열 생성	arange, array, copy, empty, empty_like, eye fromfile, fromfunction, identity, linspace, logspace, mgrid, ogrid, ones, ones_like, r, zeros, zeros_like
모양 변경	ndarray.astype, atleast_1d, atleast_2d, atleast_3d, mat
배열 조작	array_split, column_stack, concatenate, diagonal, dsplit, dstack, hsplit, hstack, ndarray.item, mewaxis, ravel, repeat, reshape, resize, squeeze, swapaxes, take, transpose, vsplit, vstack
찾기	all, any, nonzero, where
정렬	argmax, argmin, argsort, max, min, ptp, seachsorted, sort
배열 운영	choose, compress, cumprod, cumsum, inner, ndarray.fill, imag, prod, put, putmask, real, sum
기초 통계	cov, mean, std, var
선형 대수	cross, dot, outer, linalg.svd, vdot

① ndarray.ndim; 배열의 축수(차원) ndarray.shape; 각 차원 배열 크기: (n, m)
ndarray.size; 배열 요소의 총수(n*m) ndarray.dtype; 배열 내의 요소 타입 (형변환X)
ndarray.itemsize; 배열 각 요소의 바이트 단위 사이즈

② dtype; 배열 생성시 dtype을 지정(생략시 자동 셋팅). dtype변경은 astype()을 이용

2) 넘파이 배열; array, arange, ones, zeros 등 #메뉴얼: https://numpy.org/doc/stable/index.html

(1) numpy.array(object, dtype=None, copy=True)

(2) 기본값이 있는 배열 생성 #dtype=float64(기본값), 타입 지정 가능

① zeros(); 0으로 채워진 배열 ② ones(); 1로 구성된 배열

③ empty(); 초기내용이 임의이고 메모리의 상태에 따라 달라지는 배열

(3) 연속된 값을 갖는 배열 만들기 ┌#start 생략시 0, step 생략시 1

① numpy.arange([start,]stop, [step,]dtype=None); start부터 stop앞까지 step씩 증가하는 값

② numpy.linspace(start, stop, num); start부터 stop(포함)까지 num(생략시 50)개 목록을 생성

(4) 배열의 차원 변경: resize(); 배열 자체를 수정 reshape(); 차원 수정된 배열 반환

(5) 배열 인쇄: numpy.set_printoptions(threshold=None) #배열 크면 모서리만 인쇄되는 것 조정可

(6) 기본 조작; 산술 연산자(+, -, *, >, < 등)는 요소별로 적용

① 행렬의 곱: dot(), @ ② 복합 대입 연산자(+=, *= 등) 사용 – 배열 수정

③ 배열 요소의 집계: sum(), max() 등 ④ 축(면:2, 행:1, 열:0)별 집계: sum(axis=n) 등

(7) 범용 함수: 수학 함수, 삼각 함수, 비트 함수, 비교 함수, 부동 함수 등(배열의 요소마다 적용)

3) 배열 합치기/분할하기

(1) 인덱싱과 슬라이싱 #첫 인덱스:0, 맨 뒤부터: -1

np_array_obj[start : stop : step]; start부터 stop앞 까지 step씩 증가 (stop 포함X)

#다차원 배열일 경우 , 로 구분해서 각 차원별로 start, stop 인덱스를 지정

(2) 두 배열을 쌓아 합치기: hstack(); 옆에, vstack(); 아래에 dstack(); 3번째 축을 쌓아 합침

① column_stack(); 1차원 배열을 열 단위로 배열하여 2차원 배열을 만듦

② newaxis: 1차원 배열이 2차원 구조가 되도록 만듦

③ stack((A, B), axis=n); 축 속성 axis 값에 따라 배열을 합침 #0: 첫 차원, -1: 마지막 차원

(3) r_[]; 행 단위로 데이터 쌓음, c_[]; 열 단위로 데이터 쌓음 ┌#3차원 이상에서만 사용可

(4) 하나의 배열을 분할: hsplit(); 가로 축 vsplit(); 세로 축 dsplit(); 세번째 축을 따라 배열 나눔

① split(A, num, axis=n); axis가 0이면 vsplit(), 1이면 hsplit(), 2는 dsplit()와 동일함

② array_split(); split()과 다른점은 나누어 떨어지지 않는 정수도 사용 가능 (배열//n+1 반환)

4) 복사와 뷰

(1) 할당 =; 이름만 다른 동일한 객체(같은 주소, 하나를 변경하면 다른 하나도 변경 됨)

(2) 얕은 복사 view(); 메모리를 직접 소유X, shape만 가지고 있음 #슬라이싱하면 view 생성 됨

(3) 복사 copy(); 메모리를 직접 소유, 다른 주소(하나를 변경해도 다른 하나에 영향X)

5) 고급 인덱싱; 배열의 인덱싱을 단일 숫자가 아닌 넘파이 배열을 이용

6) 선형 대수학

(1) 배열 조작

① 행렬 곱: @ or dot()

② 역행렬: np.linalg.inv(x)

③ 단위 행렬: np.eye(n)

④ 대각합: np.trace(x)

⑤ 연립방정식 해 풀기: np.linalg.solve(a, b)

⑥ 대각행렬: np.diag(x)

⑦ 내적: np.dot(a, b)

(2) 선형 연립 방정식

① 최소 제곱법(Least Square Method; LSM)

② 최소 자승법(Least Mean Square; LMS)

7) 유용한 정보 및 Tip

(1) shape 자동 지정: shape 값이 -1이면 자동 지정됨

(2) 히스토그램: matplotlib의 hist()함수는 히스토그램을 자동으로 그림