Week2-1

1. 학습정리
    1. numpy
        1. 배열을 c 형식 배열로 제공
        2. reshape
            1. 나머지 파라미터에 맞춰 알아서 지정하고 싶을 때 -1 넣음
        3. diag
            1. 행렬에서 대각 값 추출
        4. 랜덤샘플링

- 데이터 분포에 따른 sampling으로 array를 생성

```
np.random.uniform(0,1,10).reshape(2,5)
```
균등분포

```
array([[ 0.67406593,  0.71072857,  0.06963986,  0.09194939,  0.47293574],
       [ 0.13840676,  0.97410297,  0.60703044,  0.04002073,  0.08057727]])
```

```
np.random.normal(0,1,10).reshape(2,5)
```
정규분포

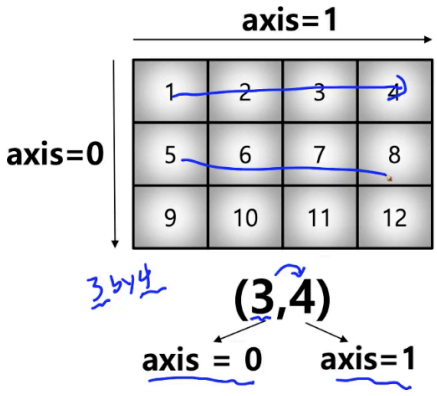
```
array([[ 1.02694847,  0.39354215,  0.63411928, -1.03639086, -1.76669162],
       [ 0.50628853, -1.42496802,  1.23288754,  1.26424168,  0.53718751]])
```

5. 축

axis

creation function

- 모든 operation function을 실행할 때 기준이 되는 dimension 축




```
test_array = np.arange(1,13).reshape(3,4)
test_array
```

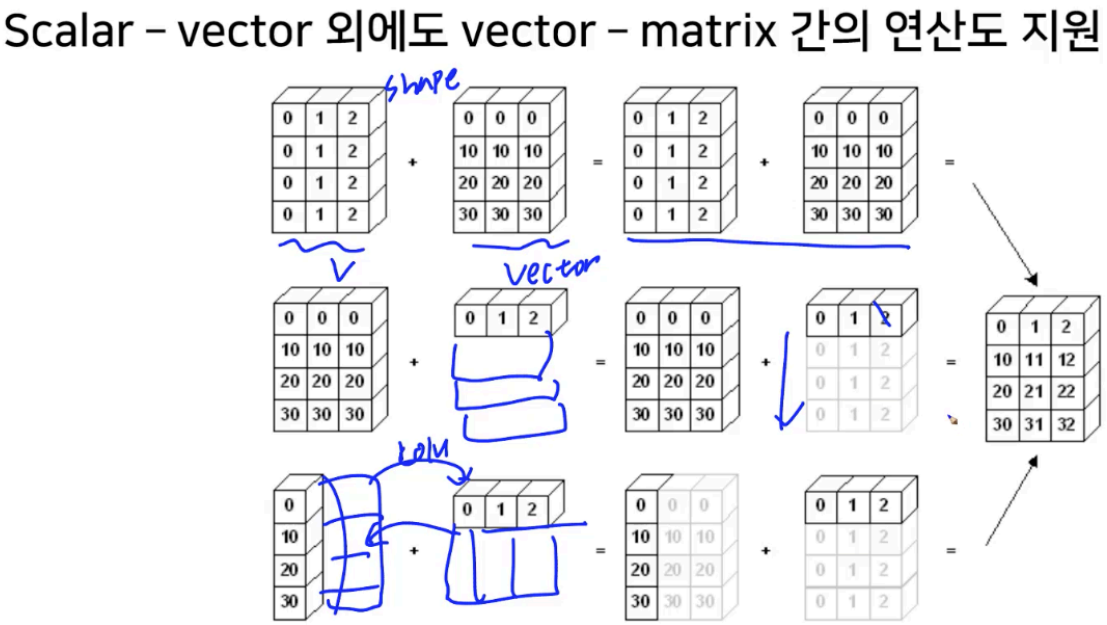
```
array([[ 1,  2,  3,  4],
       [ 5,  6,  7,  8],
       [ 9, 10, 11, 12]])
```

```
test_array.sum(axis=1), test_array.sum(axis=0)
```

```
(array([10, 26, 42]), array([15, 18, 21, 24]))
```

6. 브로드캐스팅

broadcasting



7. 비교

```
a = np.arange(10)
a
```

```
array([0, 1, 2, 3, 4, 5, 6, 7, 8, 9])
```

```
a < 4
```

```
array([ True,  True,  True,  True, False, False, False, False, False,
       False])
```

8. where

np.where

```
np.where(a > 0, 3, 2) # where(condition, TRUE, FALSE)
```

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

```
a = np.arange(10)
np.where(a>5)
```

Index 값 반환

```
(array([6, 7, 8, 9]),)
```

```
a = np.array([1, np.NaN, np.Inf], float)
np.isnan(a)
```

Not a Number

```
array([False,  True, False], dtype=bool)
```

```
np.isfinite(a)
```

is finite number

```
array([ True, False, False], dtype=bool)
```

9. boolean index

```
test_array[test_array > 3]
```

조건이 True인 index의 element만 추출

```
array([ 4.,  8.,  9.,  7.])
```

10. fancy index

fancy index

boolean & fancy inde

- numpy는 array를 index value로 사용해서 값 추출

```
a = np.array([2, 4, 6, 8], float)
b = np.array([0, 0, 1, 3, 2, 1], int) # 반드시 integer로 선언
a[b] #bracket index, b 배열의 값을 index로 하여 a의 값들을 추출함
```

```
array([ 2.,  2.,  4.,  8.,  6.,  4.])
```

```
a.take(b) #take 함수: bracket index와 같은 효과
```

```
array([ 2.,  2.,  4.,  8.,  6.,  4.])
```

0	1	2	3
2	4	6	8

2. 벡터
    1. norm
        1. 원점에서부터의 거리
        2. L1 노름 : 각 성분의 변화량의 절대값의 합

$$\|\mathbf{x}\|_1 = \sum_{i=1}^{d} |x_i|$$

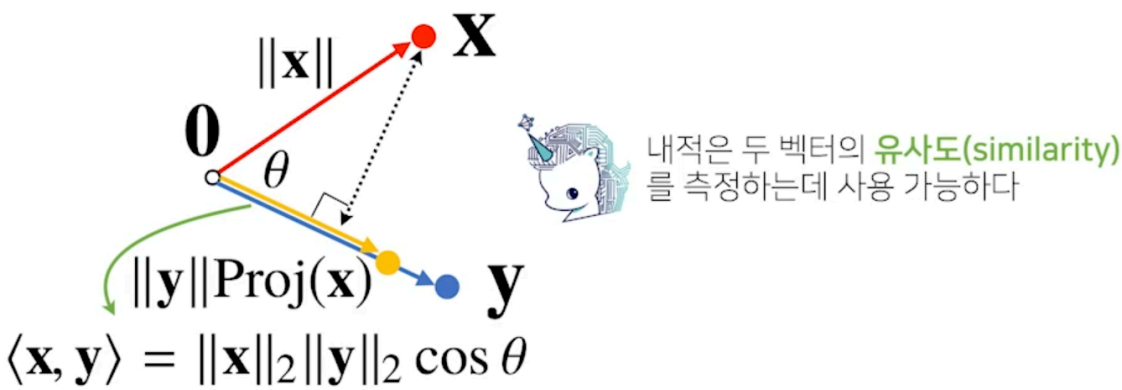
3. L2 노름 : 피타고라스 정리를 이용해 유클리드 거리 계산

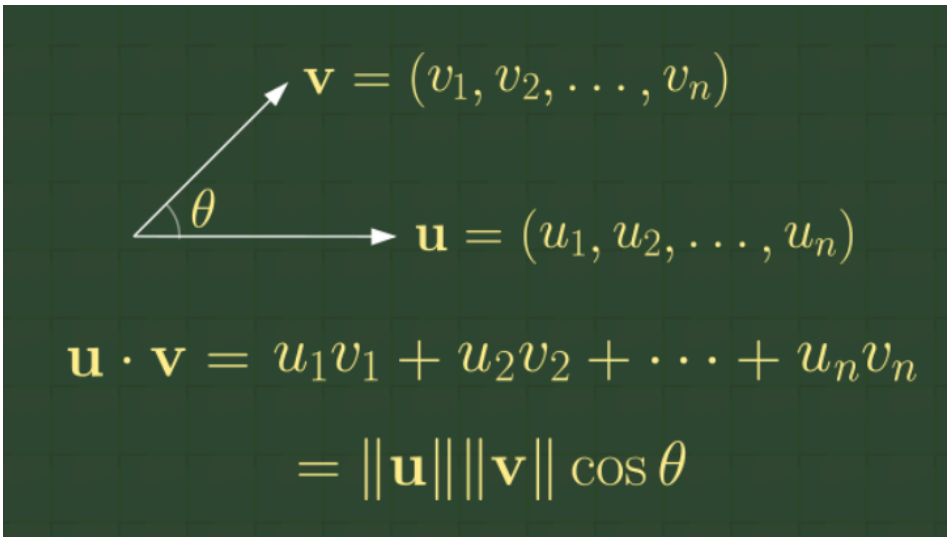
$$\|\mathbf{x}\|_2 = \sqrt{\sum_{i=1}^{d} |x_i|^2}$$

2. 두 벡터 사이의 거리 계산
    1. 벡터의 뺄셈 이용해서 계산 가능
3. 벡터 사이의 각도
    1. L2 노름으로만 계산 가능
    2. 내적과 L2 노름

$$\cos\theta = \frac{\cancel{2}\langle \mathbf{x}, \mathbf{y}\rangle}{\cancel{2}\|\mathbf{x}\|_2\|\mathbf{y}\|_2}$$

4. 내적
    1. 정사영 된 벡터의 길이를 y의 길이 만큼 조정한 값
    2. 두 벡터의 유사도 측정에 유용
    3. 대응되는 원소들을 각각 곱해서 다 더한 것





    3. 행렬
        1. 행렬 내적
            1. 두 행렬의 각 행벡터 끼리 내적한 행렬이 생성됨
        2. 역행렬
            1. 역행렬
                1. np.linalg.inv(x)
            2. 슈도역행렬
                1. 크기 상관없이 역행렬 구할 수 있음
                2. np.linalg.pinv(x)
2. 피어세션
    1. 강의 리뷰
    2. 알고리즘 문제 라이브 코딩 & 풀이 발표
        1. https://leetcode.com/problems/add-two-numbers/submissions/
    3. 코드 리뷰
    4. 질답