데이터과학과 AI를 위한 파이썬

06강. 행렬 변환

세종사이버대학교

김명배 교수

학습내용

- 벡터의 덧셈/뺄셈, 내적/외적
- 벡터의 유사도
- 역함수
- 역행렬과 전치행렬

학습목표

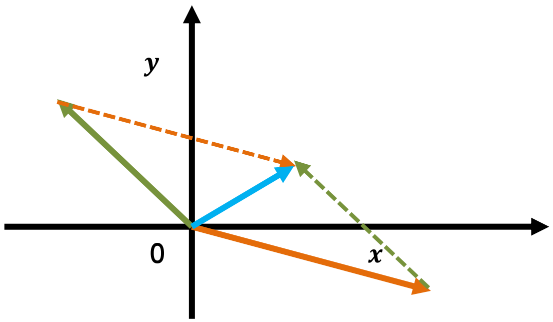
- 벡터의 덧셈과 뺄셈, 내적과 외적을 설명할 수 있고 파이썬으로 실행할 수 있다.
- 벡터의 거리와 유사도를 설명할 수 있고, 유클리드 거리, 맨해튼 거리, 코사인 유사도를 설명할 수 있다.
- 역함수를 설명할 수 있고 파이썬으로 실행할 수 있다.
- 역행렬과 전치행렬, 행렬식을 설명할 수 있다.

1. 벡터의 내적과 외적

1) 벡터의 덧셈과 뺄셈

가) 벡터의 덧셈

$$\vec{v} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \vec{w} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \text{ 일 때, } \vec{v} + \vec{w} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{bmatrix}$$



① 평행사변형 법칙
② 삼각형 법칙

1. 벡터의 내적과 외적

1) 벡터의 덧셈과 뺄셈

나) 벡터의 덧셈의 성질

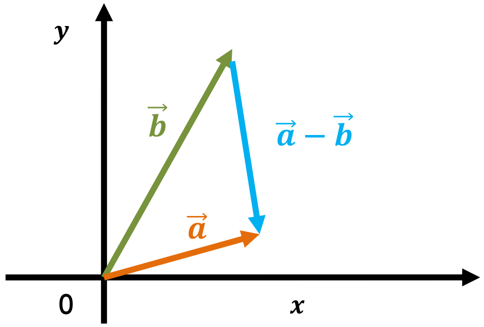
- 교환 법칙 : $A + B = B + A$
- 결합 법칙 : $(A + B) + C = A + (B + C)$
- 항등원 존재 : $A + 0 = A$
- 역원 존재 : $A + (-A) = 0$

1. 벡터의 내적과 외적

1) 벡터의 덧셈과 뺄셈

나) 벡터의 뺄셈

- 두 점 사이의 거리를 구할 때 사용하는 계산
- $\vec{a} - \vec{b} = \vec{a} + (-\vec{b})$



1. 벡터의 내적과 외적

2) 벡터의 곱셈

- 크게 **내적**과 **외적**으로 구분

가) 벡터의 내적(dot product, inner product)

- 벡터의 각 원소를 곱해서 더해주는 것(=스칼라)
- 두 벡터가 차원이 같아야 함
- 앞의 벡터가 행 벡터이고 뒤의 벡터가 열벡터 이어야 함

$$\vec{x} \cdot \vec{y} = X^T Y = \underbrace{[x_1\ x_2\ \cdots x_n]}_{1\text{행 } n\text{열}} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n = \text{스칼라}$$

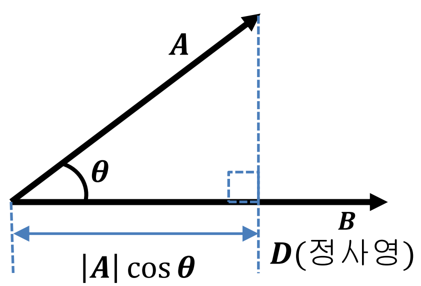
= n행 1열

1. 벡터의 내적과 외적

2) 벡터의 곱셈

가) 벡터의 내적(dot product, inner product)

- $\vec{a} \cdot \vec{b} = |\vec{a}|\,|\vec{b}|\cos\theta$



1. 벡터의 내적과 외적

2) 벡터의 곱셈

가) 벡터의 내적(dot product, inner product)

[내적의 성질]

- 교환 법칙 : $\vec{a} \cdot \vec{b} = \vec{b} \cdot \vec{a}$
- 분배 법칙 : $\vec{a} \cdot (\vec{b} + \vec{c}) = \vec{a} \cdot \vec{b} + \vec{a} \cdot \vec{c}$
- $(k\vec{a}) \cdot \vec{b} = \vec{a} \cdot (k\vec{b}) = k(\vec{a} \cdot \vec{b})$

※ 인공지능에서 개체간 유사도나 비유사도를 측정하는데 활용함

[파이썬] numpy.dot() 함수 사용

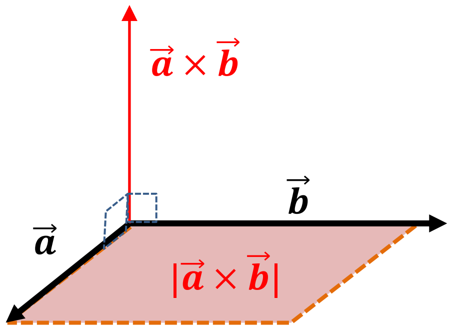
1. 벡터의 내적과 외적

2) 벡터의 곱셈

나) 벡터의 외적(outer product, cross product)

- 3차원 공간에 있는 벡터 간 연산(연산결과가 벡터임)
- 방향 : 두 벡터 $\vec{a}$와 $\vec{b}$에 동시에 수직

  크기 : $\vec{a}$와 $\vec{b}$ 크기를 변으로 하는 평행사변형의 넓이



1. 벡터의 내적과 외적

2) 벡터의 곱셈

나) 벡터의 외적(outer product , cross product)

[외적의 성질]

- 반대칭성 : $\vec{a} \times \vec{b} = -\vec{b} \times \vec{a}$
- 분배 법칙 : $\vec{a} \times (\vec{b} + \vec{c}) = \vec{a} \times \vec{b} + \vec{a} \times \vec{c}$
- $\vec{a} \cdot (\vec{b} \times \vec{c}) = (\vec{a} \times \vec{b}) \cdot \vec{c}$

1. 벡터의 내적과 외적

2) 벡터의 곱셈

나) 벡터의 외적(outer product , cross product)

- 외적 구하는 법

$\vec{a} = (a_1, a_2, a_3)$, $\vec{b} = (b_1, b_2, b_3)$ 일 때,

$\vec{a} \times \vec{b} = (a_2 b_3 - a_3 b_2,\ a_3 b_1 - a_1 b_2,\ a_1 b_2 - a_2 b_1)$

[파이썬] numpy.cross() 함수를 사용

`

1. 벡터의 내적과 외적

3) 벡터의 거리와 유사도

- 벡터의 거리는 두 데이터의 유사도(similarity) 개념
  거리가 멀다 ➔ 유사도가 낮다
  거리가 가깝다 ➔ 유사도가 높다
- 거리의 종류
  가) 유클리드 거리(Euclidean distance)
  나) 맨해튼 거리(Manhattan distance)
  다) 코사인 거리(Cosine distance)

※ 추천 시스템에서 아이템이나 사용자 간의 유사도를 측정할 때 사용.
텍스트 분석 분야에서는 문서(문장, 키워드) 간의 유사도를 측정할 때 사용

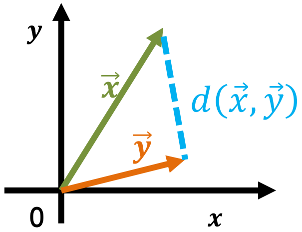
1. 벡터의 내적과 외적

3) 벡터의 거리와 유사도

가) 유클리디안 거리

- 두 벡터 간 직선거리

$$d(\vec{x}, \vec{y}) = \sqrt{(x_1 - y_1)^2 + (x_2 - y_2)^2}$$



[파이썬] SciPy의 euclidean()함수를 사용

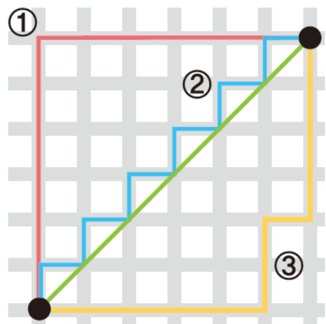
1. 벡터의 내적과 외적

3) 벡터의 거리와 유사도

나) 맨해튼 거리

- 사각형 격자로 된 경로에서 도착점까지의 최단 거리

$d(\vec{x}, \vec{y}) = |x_1 - x_2| + |y_1 - y_2|$



[파이썬] math 함수의 수학 연산으로 계산

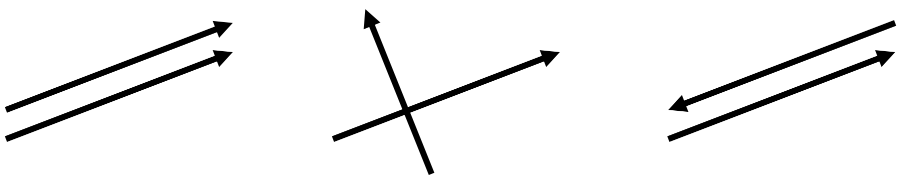
1. 벡터의 내적과 외적

3) 벡터의 거리와 유사도

다) 코사인 유사도

- 두 벡터가 이루는 각이 작을 수록 유사도가 높음

$$\text{코사인유사도} = \cos\theta = \frac{A \cdot B}{|A||B|}$$



a) 코사인 유사도 = 1　a) 코사인 유사도 = 0　a) 코사인 유사도 = -1

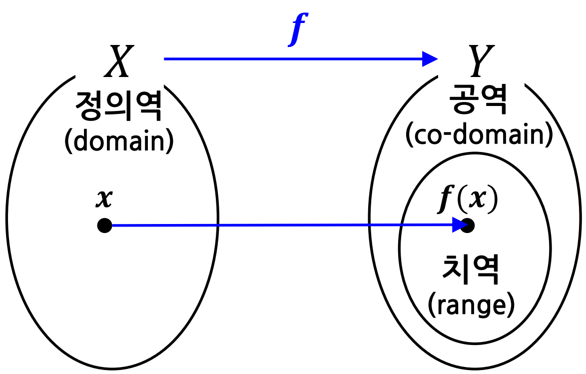
[파이썬] nympy의 norm()함수를 이용하여 계산

2. 행렬 변환

1) 함수와 벡터의 변환

가) 함수

- 함수는 두 집합 사이의 대응관계

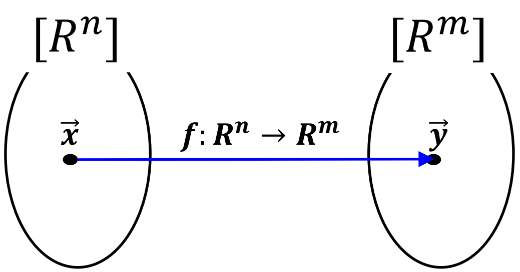


2. 행렬 변환

1) 함수와 벡터의 변환

나) 벡터의 변환

- 집합의 원소를 벡터라고 볼 수 있음



- 벡터의 변환은 비디오 게임 등 3D 공간에서 벡터의 회전 및 이동시키는데 사용됨

2. 행렬 변환

2) 선형 변환

- 선형 결합을 보존하는 두 벡터 공간 사이의 함수

  즉, 하나의 벡터공간을 다른 벡터공간으로 변환시키는 함수($R^m \rightarrow R^n$)

- 선형결합의 두 조건

a) 합의 법칙 : $T(\vec{a}+\vec{b}) = T(\vec{a}) + T(\vec{b})$

b) 스칼라 곱의 법칙 : $T(c\vec{a}) = cT(\vec{a})$

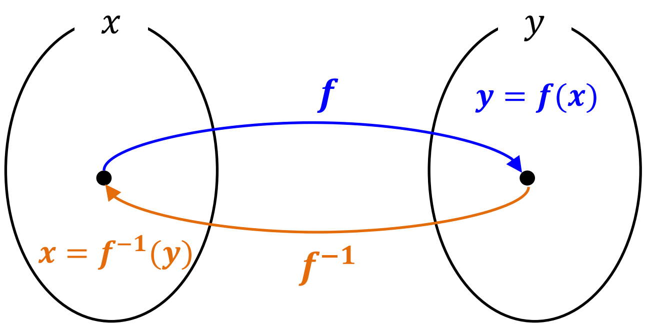
[예시]

$$A = \begin{bmatrix} 1 & 2 & 3 \\ 3 & 4 & 5 \end{bmatrix}, \underbrace{B = \begin{bmatrix} 5 \\ 6 \\ 7 \end{bmatrix}}_{R^3} \quad \Rightarrow \quad \underbrace{AB = \begin{bmatrix} 38 \\ 74 \end{bmatrix}}_{R^2}$$

2. 행렬 변환

3) 역함수

- 변수(x)와 함수 값(y)을 서로 뒤바꾸어 얻는 함수
- 역함수를 갖는 함수를 가역함수라고 함
- $(f^{-1})^{-1} = f$



2. 행렬 변환

3) 역함수

가) 역함수의 성질

- $(f^{-1})^{-1} = f$
- $(f^{-1} \circ f)(\vec{x}) = \vec{x}(\vec{x} \in X)$
  $(f^{-1} \circ f)$: X에서의 항등함수
- $(f \circ f^{-1})(\vec{y}) = \vec{y}(\vec{y} \in Y)$
  $(f \circ f^{-1})$: Y에서의 항등함수

[참고]항등함수와 상수함수

항등함수 : $I_x = f(x) = x$

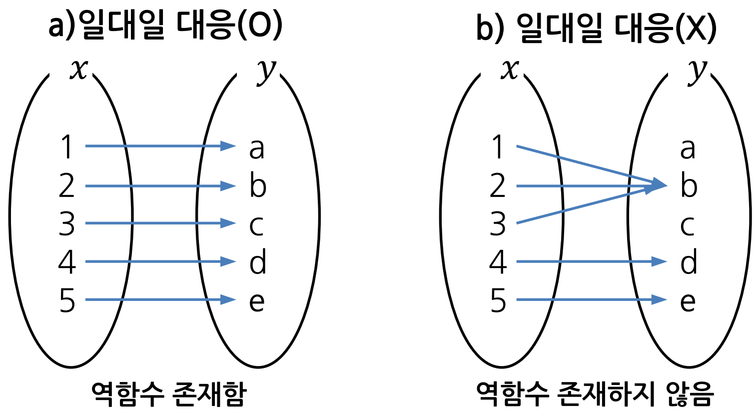
상수함수 : $f(x) = c$, c는 상수

2. 행렬 변환

3) 역함수

나) 역함수 구하는 방법

- $y = f(x)$를 x에대해 풀고 x와 y, 정의역과 치역을 바꿈
  (∴ 원래 함수는 꼭 일대일 대응이어야 한다.)



2. 행렬 변환

4) 역행렬(inverse matrix)

- 행렬 A와 곱셈하였을 때 단위행렬 I가 나오는 행렬(B)

$AB = BA = I, \quad B = A^{-1}$

가) 역행렬 공식

2차 정사각행렬 $A = \begin{bmatrix} a & b \\ c & d \end{bmatrix}$에 대해

- $ad - bc \neq 0$ 이면 $A^{-1} = \frac{1}{ad-bc}\begin{bmatrix} d & -b \\ -c & ad \end{bmatrix}$
- $ad - bc = 0$ 이면 A의 역행렬은 없음

※ 여기서 $ad - bc$를 행렬식이라고 함

2. 행렬 변환

4) 역행렬(inverse matrix)

나) 역행렬 성질

a) $(A^{-1})^{-1} = A$

b) $(kA)^{-1} = \frac{1}{k}A^{-1}$

c) $(AB)^{-1} = B^{-1}A^{-1}$

2. 행렬 변환

4) 역행렬(inverse matrix)

다) 역행렬의 활용

- 선형방정식의 해(X)를 구할 때 유용함($AX = B$ ☒$X = A^{-1}B$)

$$\begin{gathered} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = b_2 \\ \vdots \\ a_{n1}x_1 + a_{n2}x_2 + \cdots + a_{nn}x_n = n \end{gathered} \quad \square \begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix} = \begin{bmatrix} b_1 \\ \vdots \\ b_n \end{bmatrix}$$

[파이썬] numpy의 linalg.inv() 함수 활용

2. 행렬 변환

5) 전치행렬

가) 전치행렬(transposed matrix)

- 행렬의 열과 행을 바꾼 행렬($m \times n \ \rightarrow n \times m$)

$$A_{m\times n} = A^T_{n\times m}$$

$$A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix} \qquad A^T = \begin{bmatrix} a_{11} & a_{21} & \cdots & a_{m1} \\ a_{12} & a_{22} & \cdots & a_{m2} \\ \vdots & \vdots & \ddots & \vdots \\ a_{1n} & a_{2n} & \cdots & a_{nm} \end{bmatrix}$$

2. 행렬 변환

5) 전치행렬

나) 전치행렬의 성질

a) $\left(A^T\right)^T = A$

b) $(A+B)^T = A^T + B^T$

c) $(AB)^T = B^T A^T$

d) $(kA)^T = kA^T$ (k는 임의의 실수)

[파이썬] numpy의 a.T, transpose(a), swapaxes(a, 0, 1) 사용

2. 행렬 변환

5) 전치행렬

다) 전치행렬의 행렬식(diterminant)

- 행 개수와 열의 개수가 같은 행렬(정사각행렬)에 수를 대응시키는 함수
- $\det A$로 표시함

a) 2 × 2의 행렬식 : $\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$

b) 3 × 3의 행렬식 :

$$\det \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = (aei + bfg + cdh) - (ceg + bdi + afh)$$

2. 행렬 변환

5) 전치행렬

라) 전치행렬의 역

- 어떤 행렬이 가역성이면 전치행렬도 가역성을 가짐

$$(A^T)^{-1} = (A^{-1})^T$$

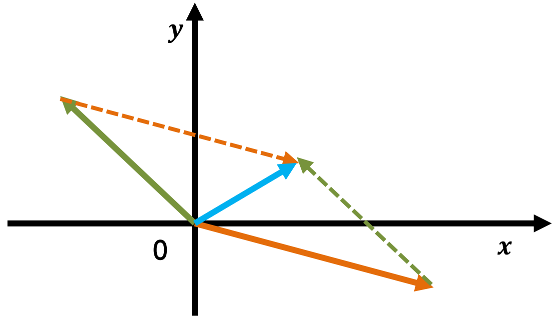
실습 동영상

정리하기

1. 벡터의 덧셈

$$\vec{v} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \vec{w} = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} \text{ 일 때, } \vec{v} + \vec{w} = \begin{bmatrix} x_1 + y_1 \\ x_2 + y_2 \\ \vdots \\ x_n + y_n \end{bmatrix}$$

- 평행사변형 법칙, 삼각형 법칙

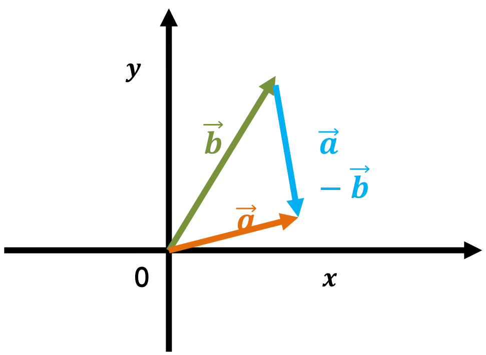


정리하기

2. 벡터의 뺄셈

- 두 점 사이의 거리를 구할 때 사용하는 계산
- 덧셈 방식으로 계산

$\vec{a} - \vec{b} = \vec{a} + (-\vec{b})$



정리하기

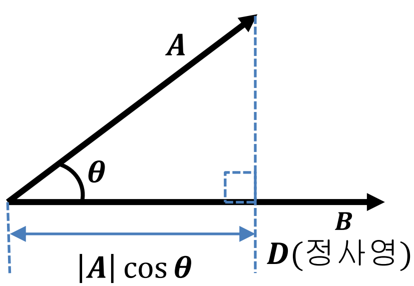
3. 벡터의 곱셈 - 내적

- 벡터의 각 원소를 곱해서 더해주는 것(=스칼라)

$$\vec{x} \cdot \vec{y} = X^T Y = \underbrace{[x_1 \ x_2 \ \cdots x_n]}_{1\text{행 } n\text{열}} \underbrace{\begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix}}_{n\text{행 } 1\text{열}} = x_1 y_1 + x_2 y_2 + \cdots + x_n y_n = \text{스칼라}$$

1행 n열 = n행 1열

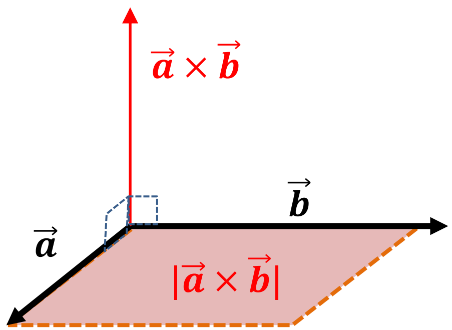
- $\vec{a} \cdot \vec{b} = |\vec{a}| \, |\vec{b}| \cos \theta$



정리하기

3. 벡터의 곱셈 - 외적

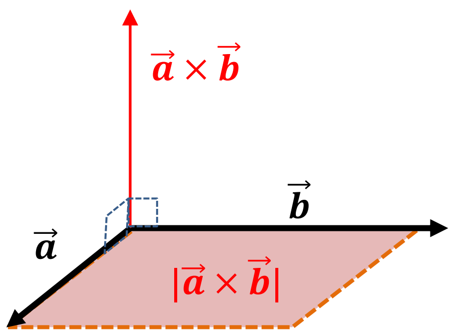
- 3차원 공간에 있는 벡터 간 연산(연산결과가 벡터임)
- 방향 : 두 벡터 $\vec{a}$와 $\vec{b}$에 동시에 수직
- 크기 : $\vec{a}$와 $\vec{b}$ 크기를 변으로 하는 평행사변형의 넓이
- [파이썬] numpy.dot() 함수 사용



정리하기

4. 벡터의 곱셈 - 외적

- 3차원 공간에 있는 벡터 간 연산(연산결과가 벡터임)
- 방향 : 두 벡터 $\vec{a}$와 $\vec{b}$에 동시에 수직
- 크기 : $\vec{a}$와 $\vec{b}$ 크기를 변으로 하는 평행사변형의 넓이
- [파이썬] numpy.cross() 함수 사용



$\vec{a} = (a_1, a_2, a_3)$, $\vec{b} = (b_1, b_2, b_3)$ 일 때,

$$\vec{a} \times \vec{b} = (a_2 b_3 - a_3 b_2,\ a_3 b_1 - a_1 b_2,\ a_1 b_2 - a_2 b_1)$$

정리하기

5. 벡터의 거리와 유사도

- 벡터의 거리 ↔ 유사도
- 거리의 종류

가) 유클리드 거리 : 두 벡터 간 직선거리(scipy.euclidean 함수)

나) 맨해튼 거리 : 사각형 격자로 된 경로에서 최단 거리(math 함수)

다) 코사인 유사도 : 두 벡터가 이루는 각을 이용한 유사도(numpy.norm 함수)

6. 행렬 변환

- 벡터의 변환 : 집합의 원소를 벡터라고 본 함수
- 선형 변환 : 선형 결합을 보존하는 두 벡터 공간 사이의 함수($R^m \rightarrow R^n$)
- 역함수 : 변수(x)와 함수 값(y)을 <u>서로 뒤바꾸어</u> 얻는 함수(($f^{-1} \circ f$) : X에서의 항등함수)
- 역행렬 : 행렬 A와 곱셈하였을 때 <u>단위행렬</u> I가 나오는 행렬(B)($AB = BA = I$)
  선형방정식의 해(X)를 구할 때 유용함($AX = B$ ☒$X = A^{-1}B$)
  (**[파이썬]** numpy의 linalg.inv() 함수 활용)
- 전치행렬 : 행렬의 행과 열을 바꾼 행렬($m \times n \rightarrow n \times m$)
- 행렬식 : 행 개수와 열의 개수가 같은 행렬(정사각행렬)에 수를 대응시키는 함수($\det A$)