Week 1. Basic MathMatics (Linear Algebra)

Kisoo Kim

Kwangwoon University, Information Convergence

Kisooofficial@naver.com

목차

- 스칼라
- 벡터
  - 여러 가지 벡터의 연산
  - 벡터의 내적
- 행렬
  - 행렬의 연산
  - 역행렬
- 선형 시스템
- 고윳값 분해 및 특이값 분해

스칼라

- 스칼라(Scala)
  - 크기만으로 나타낼 수 있는 물리량
    - 길이, 넓이, 온도, 무게
    - $s \in \mathbb{R}$
  - 스칼라는 덧셈, 뺄셈, 곱셈, 나눗셈 모두 가능

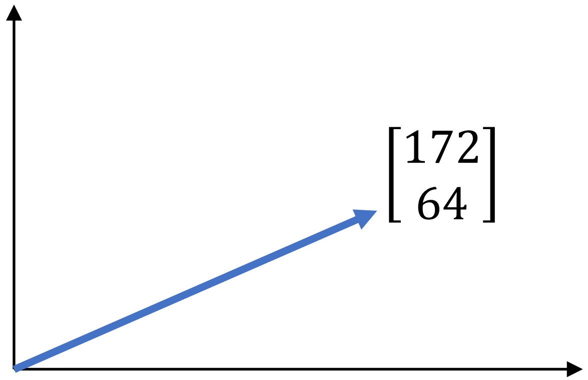
Index	나이	키	몸무게
1	25	172	64
2	24	180	92
3	25	177	75
4	22	169	71
5	28	188	80

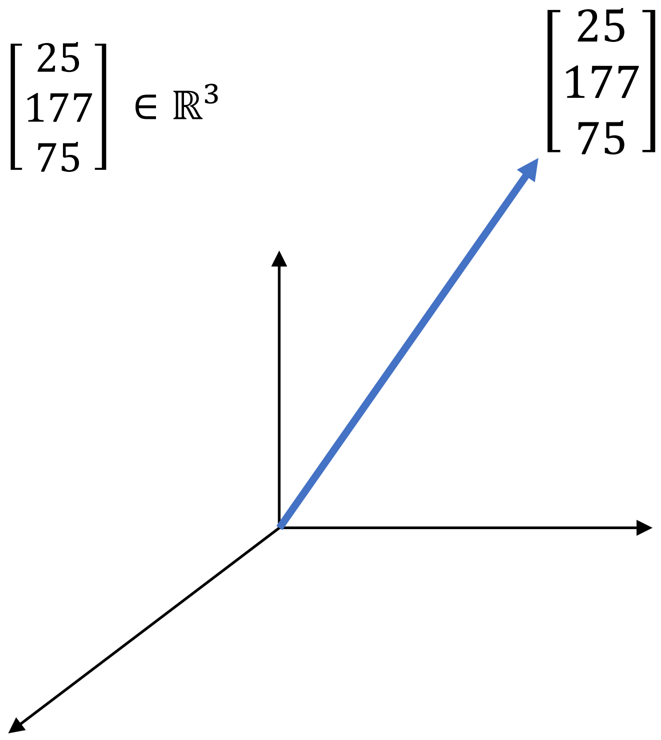
벡터(Vector)

- 벡터(Vector)
  - 스칼라의 집합, 행렬(Matrix)를 구성하는 기본 단위
  - 크기와 방향을 모두 나타내는 표현
    - 크기와 방향을 갖는 유향선분

$$a = \begin{bmatrix} 172 \\ 64 \end{bmatrix} \in \mathbb{R}^2, \qquad b = \begin{bmatrix} 25 \\ 177 \\ 75 \end{bmatrix} \in \mathbb{R}^3$$

  - 벡터의 기하학적인 의미





벡터(Vector)

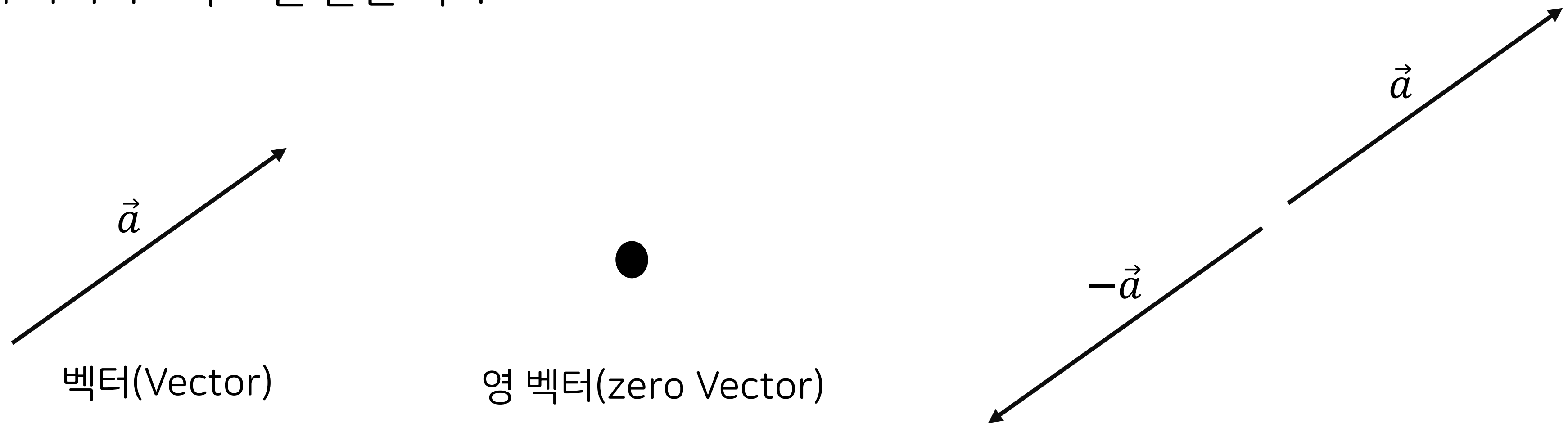
- 벡터(Vector)
  - 열 벡터(column vector) : 열 방향으로 나열한 벡터(기본값)
    - 같은 데이터의 다른 사람들 (ex. 여러 사람들의 몸무게 데이터)
  - 행 벡터(row vector) : 행 방향으로 나열한 벡터, 열 벡터의 transpose
    - 같은 사람의 여러 항목에 대한 데이터 (ex. Index 3의 나이, 키, 몸무게 데이터)

Index	나이	키	몸무게
1	25	172	64
2	24	180	92
3	25	177	75
4	22	169	71
5	28	188	80

벡터(Vector)

- 영 벡터(zero vector)
  - 벡터의 시작 지점과 종료 지점이 동일한 벡터
  - 크기가 0인 벡터

- 크기는 같지만 방향이 서로 반대인 벡터
  - 원래 벡터에 마이너스 부호를 붙인 벡터



벡터 : 연산

- 두 벡터 간의 덧셈, 뺄셈, 스칼라 곱이 모두 가능함
  - 전제 조건 : 두 벡터의 크기가 동일할 때만 연산이 가능하다.
    - 수식을 통한 이해

$$a = \begin{bmatrix} 8 \\ 3 \\ 5 \end{bmatrix} \in \mathbb{R}^3, \qquad b = \begin{bmatrix} 2 \\ 5 \\ 10 \end{bmatrix} \in \mathbb{R}^3;$$

$$a + b = \begin{bmatrix} 8 \\ 3 \\ 5 \end{bmatrix} + \begin{bmatrix} 2 \\ 5 \\ 10 \end{bmatrix} = \begin{bmatrix} 8+2 \\ 3+5 \\ 5+10 \end{bmatrix} = \begin{bmatrix} 10 \\ 8 \\ 15 \end{bmatrix}; \qquad 2a - 3b = 2\begin{bmatrix} 8 \\ 3 \\ 5 \end{bmatrix} - 3\begin{bmatrix} 2 \\ 5 \\ 10 \end{bmatrix} = \begin{bmatrix} 16 \\ 6 \\ 10 \end{bmatrix} - \begin{bmatrix} 6 \\ 15 \\ 30 \end{bmatrix} = \begin{bmatrix} 16-6 \\ 6-15 \\ 10-30 \end{bmatrix} = \begin{bmatrix} 10 \\ -9 \\ -20 \end{bmatrix}$$

  - 벡터는 2, 3차원이 아닌 n차원으로 확장 가능

- 모든 n차원 벡터 전체의 집합을 n**-공간** (n차원 공간) $\mathbb{R}^n$으로 나타낸다. 즉

$$\mathbb{R}^n = \{(x_1, x_2, \ldots, x_n) \mid x_i \in \mathbb{R}, i = 1, 2, \ldots, n\}$$

벡터의 연산법칙

- 벡터의 연산 법칙
  1. u + v = v + u
  2. (u + v) + w = u + (v + w)
  3. u + 0 = 0 + u = u
  4. u + (-u) = 0
  5. k(u+v) = ku + kv (단, k는 임의의 실수)
  6. k(lu) = l(ku)
  7. (k+l)u = ku + lu

벡터의 내적

- 노름(norm)
  - $\mathbb{R}^n$의 벡터 $X = (x_1,\ x_2,\ x_3, \ldots, x_n)$에 대하여

$$\| \mathbf{x} \| = \sqrt{x_1^2 + x_2^2 + \cdots + x_n^2}$$

를 벡터 x의 노름(norm)이라고 함. 즉, 원점에서 벡터 X까지의 거리가 된다.

  - 두 벡터 X, Y의 차에 대한 노름의 의미는?

벡터의 내적

- 벡터의 내적
  - 두 벡터 간의 유사도를 구할 때 많이 쓰임
    - EX. 자연어 처리에서 두 문서 간의 유사도를 어떻게 구해야 할까?
  - 두 벡터의 내적

정의. **[내적**(Euclidean inner product, dot product)**]** $\mathbb{R}^n$의 벡터 $\mathbf{x} = (x_1, x_2, \ldots, x_n)$, $\mathbf{y} = (y_1, y_2, \ldots, y_n)$에 대하여 실수

$$x_1y_1 + x_2y_2 + \cdots + x_ny_n$$

을 $\mathbf{x}$와 $\mathbf{y}$의 **내적**(Euclidean inner product, dot product)이라 하고 $\mathbf{x} \cdot \mathbf{y}$로 나타낸다. 즉

$$\mathbf{x} \cdot \mathbf{y} = x_1y_1 + x_2y_2 + \cdots + x_ny_n = < \mathbf{x}, \mathbf{y} > = \mathbf{y}^T\mathbf{x}$$

벡터의 내적

- 유사도를 구하는 데 있어 내적을 하는 이유가 무엇일까?
  - 거리 기반의 유사도의 한계점
    - 데이터 분석 시, 데이터의 패턴에 관심이 있는 경우라면?
  - 코사인 유사도의 등장
    - 거리 기반이 아닌, 각도 기반으로 유사도를 구함. 두 벡터 간의 각이 작으면 유사도가 높다고 판단

정의. $\mathbb{R}^n$의 벡터 $\mathbf{x}$, $\mathbf{y}$에 대하여

$$\mathbf{x} \cdot \mathbf{y} = \|\mathbf{x}\| \ \|\mathbf{y}\| \cos\theta, \quad (0 \le \theta \le \pi)$$

인 θ를 $\mathbf{x}$와 $\mathbf{y}$가 이루는 각(angle, 사잇각)이라 한다.

벡터의 내적

- $\mathbb{R}^4$의 벡터 x = (2, -1, 3, 2), y = (3, 2, 1, -4)에 대하여 ||x-y||와 x · y, $cos\theta$를 구하시오.

행렬(Matrix)

- 행렬(Matrix)
  - 사각형 형태로 숫자를 나열 ($A \in \mathbb{R}^{m \times n}$)
  - 행과 열로 구성된 형태
    - 행 : 가로 방향; 열 : 세로 방향
    - m은 행 벡터의 개수, n은 열 벡터의 개수

$$A = \begin{bmatrix} 3 & 2 \\ 4 & 1 \\ 5 & 4 \end{bmatrix} \in \mathbb{R}^{3 \times 2}$$

행렬(Matrix)

- 행렬(Matrix)
  - 행렬을 구성하는 스칼라 값 : 원소(element)

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \\ a_{31} & a_{32} \end{bmatrix} \in \mathbb{R}^{3\times2}, \qquad a_{ij}(i : row\ number, j : column\ number)$$

  - 데이터 분석 시 데이터를 구성하는 어떤 숫자들의 집합

Index	나이	키	몸무게
1	25	172	64
2	24	180	92
3	25	177	75
4	22	169	71
5	28	188	80

$$\longrightarrow \begin{bmatrix} 1 & 25 & 172 & 64 \\ 2 & 24 & 180 & 92 \\ 3 & 25 & 177 & 75 \\ 4 & 22 & 169 & 71 \\ 5 & 28 & 188 & 80 \end{bmatrix} \in \mathbb{R}^{5\times4}$$

행렬 : 연산

- 두 행렬의 덧셈과 뺄셈
  - 벡터와 동일하게, 두 행렬의 크기가 같은 경우에만 가능하며, 같은 원소끼리 더함

$$C = A \pm B \left(c_{ij} = a_{ij} + b_{ij}\right)$$

$$A = \begin{bmatrix} 4 & 1 \\ -1 & -3 \end{bmatrix}, \qquad B = \begin{bmatrix} -2 & 0 \\ 2 & -1 \end{bmatrix},$$

$$A + B = \begin{bmatrix} 4-2 & 1+0 \\ -1+2 & -3-1 \end{bmatrix} = \begin{bmatrix} 2 & 1 \\ 1 & -4 \end{bmatrix},$$

$$A - B = \begin{bmatrix} 4-(-2) & 1-0 \\ -1-2 & -3-(-1) \end{bmatrix} = \begin{bmatrix} 6 & 1 \\ -3 & -2 \end{bmatrix}$$

행렬 : 연산

- 행렬의 스칼라 곱

$$D = k \times A + B \left(d_{ij} = k \times a_{ij} + b_{ij}\right)$$

$$A = \begin{bmatrix} 4 & 1 \\ -1 & -3 \end{bmatrix}, \quad B = \begin{bmatrix} -2 & 0 \\ 2 & -1 \end{bmatrix}, \quad 2A = \begin{bmatrix} 8 & 2 \\ -2 & -6 \end{bmatrix}, \quad 3B = \begin{bmatrix} -6 & 0 \\ 6 & -3 \end{bmatrix}$$

$$2A + B = \begin{bmatrix} 8 & 2 \\ -2 & -6 \end{bmatrix} + \begin{bmatrix} -2 & 0 \\ 2 & -1 \end{bmatrix} = \begin{bmatrix} 8-2 & 2+0 \\ -2+2 & -6-1 \end{bmatrix} = \begin{bmatrix} 6 & 2 \\ 0 & -7 \end{bmatrix},$$

$$2A - 3B = \begin{bmatrix} 8 & 2 \\ -2 & -6 \end{bmatrix} - \begin{bmatrix} -6 & 0 \\ 6 & -3 \end{bmatrix} = \begin{bmatrix} 8-(-6) & 2-0 \\ -2-6 & -6-(-3) \end{bmatrix} = \begin{bmatrix} 14 & 2 \\ -8 & -3 \end{bmatrix}$$

BOAZ

행렬 : 연산

- 두 행렬의 원소 곱
  - 두 행렬의 크기가 동일할 때만 가능하고, 두 동일한 위치의 원소를 곱함

$$C = A \odot B \ \left(c_{ij} = a_{ij} b_{ij}\right)$$

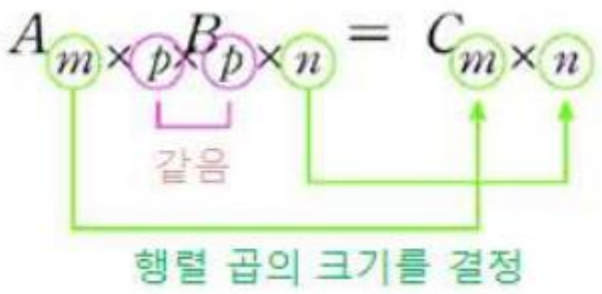
$$A = \begin{bmatrix} 4 & 1 \\ -1 & -3 \end{bmatrix}, \qquad B = \begin{bmatrix} -2 & 0 \\ 2 & -1 \end{bmatrix},$$

$$A \odot B = \begin{bmatrix} 4 \times (-2) & 1 \times 0 \\ -1 \times 2 & -3 \times (-1) \end{bmatrix} = \begin{bmatrix} -8 & 0 \\ -2 & 3 \end{bmatrix},$$

BOAZ

행렬 : 연산

- 행렬의 곱(matrix multiplication)
  - 교환법칙이 성립하지 않는 연산
  - 곱하는 행렬의 열 크기와 곱해지는 행렬의 행 크기가 같아야 함.
    - Ex. 행렬 AB에서 A가 3x2 행렬이라면, B의 행 크기는 2xn의 형태가 되어야 함. (행렬 곱 후에 행렬의 크기는 3xn)



행렬 : 연산

- 행렬의 곱(matrix multiplication)
  - 곱하는 행렬의 열 크기와 곱해지는 행렬의 행 크기가 같아야 함.
  - 곱하는 행렬과 곱해지는 행렬의 대상을 고르고, 서로 같은 위치에 있는 요소끼리 곱함

$$C = AB(C_{ij} = a_{i1}b_{1j} + a_{i2}b_{2j} + a_{i3}b_{3j} + \cdots + a_{ip}b_{pj} = \sum_{k=1}^{p} a_{ik}b_{kj}$$

$$C = AB = \begin{bmatrix} A_{(1)} \\ A_{(2)} \\ \vdots \\ A_{(m)} \end{bmatrix} \begin{bmatrix} B^{(1)} & B^{(2)} & \cdots & B^{(n)} \end{bmatrix} = \begin{bmatrix} A_{(1)}B^{(1)} & A_{(1)}B^{(2)} & \cdots & A_{(1)}B^{(n)} \\ \vdots & & & \vdots \\ A_{(m)}B^{(1)} & & \cdots & A_{(m)}B^{(n)} \end{bmatrix}_{m \times n}$$

$$AB = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1p} \\ \vdots & \vdots & & \vdots \\ \boxed{a_{i1} \quad a_{i2} \quad \cdots \quad a_{ip}} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mp} \end{bmatrix} \begin{bmatrix} b_{11} & \cdots & b_{1j} & \cdots & b_{1n} \\ b_{21} & \cdots & b_{2j} & \cdots & b_{2n} \\ \vdots & & \vdots & & \vdots \\ b_{p1} & \cdots & b_{pj} & \cdots & b_{pn} \end{bmatrix}$$

행렬 : 연산

- 행렬 $A = \begin{bmatrix} 2 & 3 & 5 \\ 1 & 4 & 2 \end{bmatrix}$, $B = \begin{bmatrix} -1 & 2 \\ -3 & 1 \\ 0 & 3 \end{bmatrix}$에 대하여 두 행렬의 곱 AB, BA를 각각 구하시오.

BOAZ

여러 가지 행렬

- 전치 행렬
  - 기존의 행과 열을 바꾼 바꾼 행렬
  - A의 전치 행렬을 A^T로 표현함. 행렬 A의 크기가 2 x 3이라면, A의 전치 행렬의 크기는 3 x 2가 됨

$$A^T = [a_{ij}{}']_{n \times m},\ a_{ij}{}' = a_{ji} \quad (1 \le i \le n, 1 \le j \le m)$$

$$A = \begin{bmatrix} 1 & 2 & 4 \\ 3 & 1 & 5 \end{bmatrix}, \qquad A^T = \begin{bmatrix} 1 & 3 \\ 2 & 1 \\ 4 & 5 \end{bmatrix}$$

- 대칭 행렬
  - 기존 행렬과 전치 행렬이 같은 정사각행렬, $A = A^T$
    - 정사각행렬 : 행과 열의 크기가 같은 행렬, 예를 들면, 행렬의 크기가 2 x 2, 3 x 3, 4 x 4인 경우가 해당

여러 가지 행렬

- 대각 행렬(diagonal matrix)
  - 정사각행렬에서 대각선의 원소를 제외한 나머지 원소가 모두 0인 행렬
    - 특히, B와 같이 대각선의 원소가 모두 1인 대각 행렬을 단위 행렬(identity matrix)라고 함

$$A = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 2 & 0 \\ 0 & 0 & 3 \end{bmatrix}, \qquad B = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

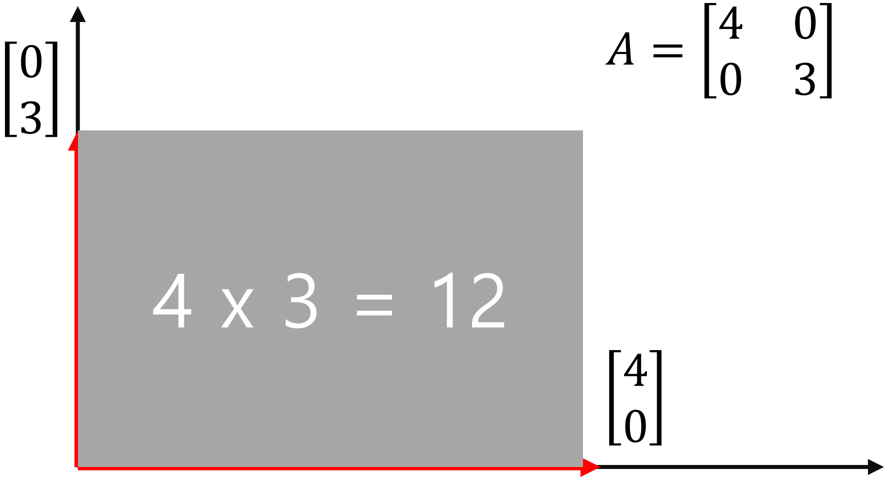
  - 단위 행렬의 경우에는 차원 축소(PCA) 혹은 Neural Network에서 weight parameter를 초기화 하는 데 쓰임

- 영행렬(zero matrix)
  - 모든 행렬 구성 원소가 0인 행렬

행렬식

- 행렬식(determinant, det(A) or |A|)
  - 정사각행렬의 특성을 하나의 스칼라로 표현
  - 행렬이 단위 공간을 얼마나 늘렸는 지 혹은 줄였는 지를 나타냄
    - 행렬식 = 1 : 해당 행렬이 단위 공간의 부피와 같음
    - 행렬식 = 0 : 해당 행렬이 나타내는 부피가 0
    - 행렬식 = 10 : 해당 행렬이 단위 공간 부피의 10배에 해당함

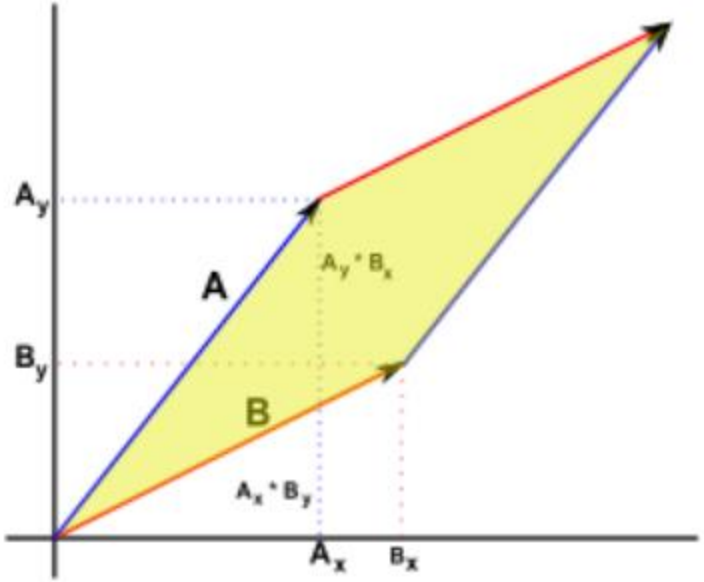
$$A = \begin{bmatrix} 4 & 0 \\ 0 & 3 \end{bmatrix}$$



행렬식

- 행렬식(determinant, det(A) or |A|)
  - 2 x 2 행렬식

$$\det(A) = |A| = \begin{vmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{vmatrix} = a_{11}a_{22} - a_{12}a_{21}$$



역행렬(Inverse Matrix)

- 역행렬(Inverse Matrix)
  - 행렬 A의 역행렬이란 $AB = I$를 만족하는 행렬 B
  - 행렬 A의 역행렬을 A^{-1}처럼 표현함.
  - 데이터 분석에서 고윳값 분해, 특이값 분해, 컴퓨터 그래픽스(object의 위치, 회전 조절 등), 최적화 문제에서 활용

$$AB = I = AA^{-1} = A^{-1}A$$

- 가역행렬(Invertible Matrix)
  - 역행렬이 존재하는 행렬
    - 역행렬이 존재하지 않는 행렬을 비가역행렬(noninvertible matrix)
  - 역행렬이 존재하기 위한 조건은 행렬 A의 행렬식이 0이다. $(\det(A) = 0)$

BOAZ

역행렬

- 2x2 행렬의 역행렬 연산
  - 행렬 A의 행렬식을 계산해서 0인지 확인
  - 행렬 A의 행렬식이 0이 아니라면, 아래 식을 이용하여 역행렬 계산

$$A = \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \qquad A^{-1} = \frac{1}{\det(A)} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix} = \frac{1}{a_{11}a_{22} - a_{12}a_{21}} \begin{bmatrix} a_{22} & -a_{12} \\ -a_{21} & a_{11} \end{bmatrix}$$

- 역행렬을 제대로 구했는 지 확인하는 방법
  - $AA^{-1} = I$를 만족하는 지 확인

역행렬

- 아래 행렬의 역행렬을 구하고, 역행렬이 맞는 지 확인하시오.

$$A = \begin{bmatrix} 4 & 2 \\ 5 & 3 \end{bmatrix}$$

선형방정식(Linear equation)

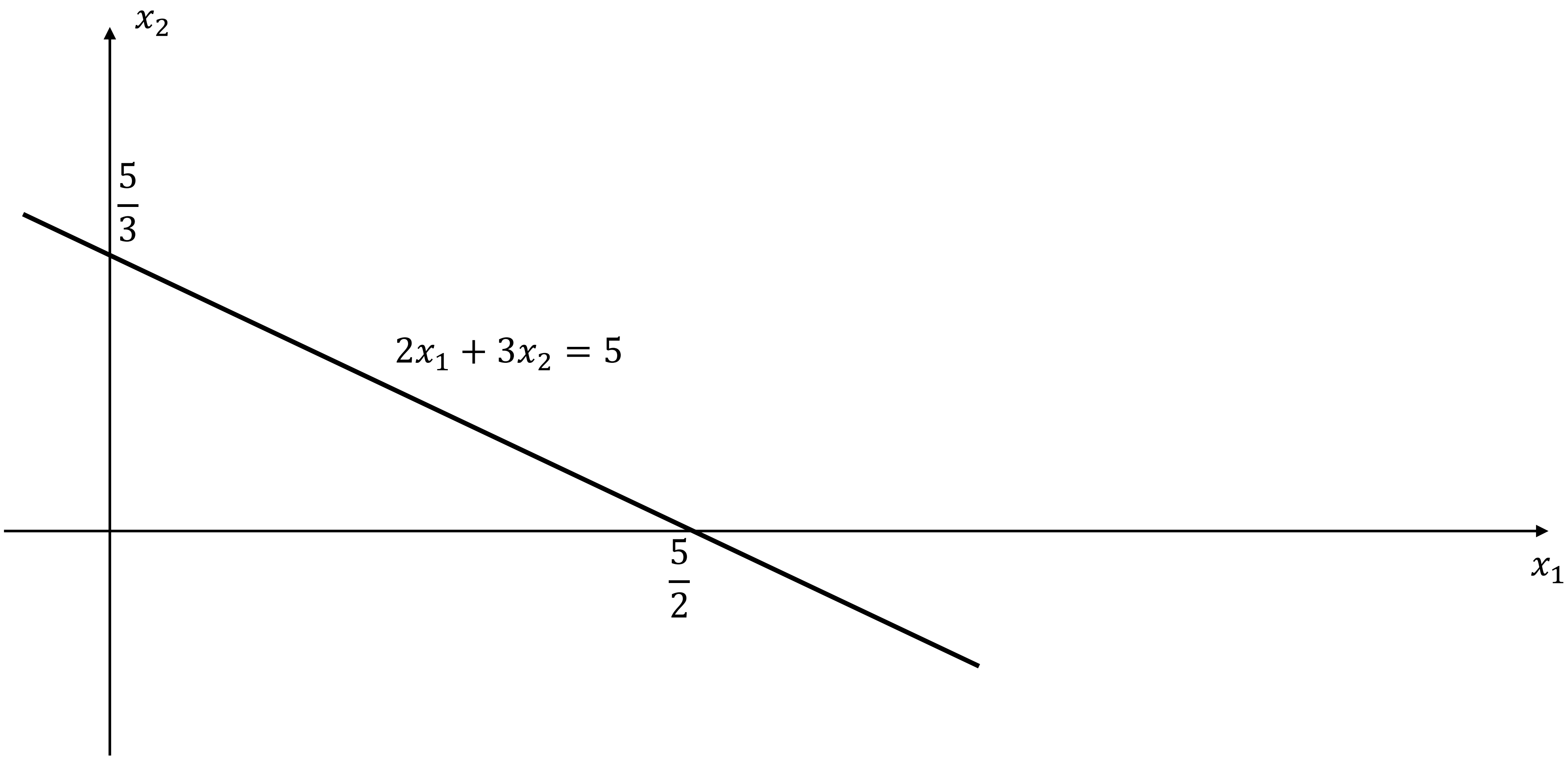
- 선형방정식(linear equation)의 정의
  - 변수($x_1, x_2, x_3, \cdots, x_n$)에 대한 1차 방정식
  - 최고차항의 차수가 1임
    - Ex. $2x_1 + 3x_2 = 5, \quad x_1 + 2x_2 + 3x_3 + 4x_4 = 10$
    - $x_1 + x_2 + x_3x_4 + x_5^2 = 10$은 선형방정식이 아니다.
- 선형방정식의 형태

$$a_1x_1 + a_2x_2 + a_3x_3 + \cdots + a_nx_n = b$$

  - $x_1, x_2, x_3, \cdots, x_n$ 은 변수이고, $a_1, a_2, a_3, \cdots, a_n$ 는 각 변수 앞에 붙은 계수이다.
- 선형방정식의 해
  - $x_1, x_2, x_3, \cdots, x_n$ 의 값을 찾는 것, $2x_1 + 3x_2 = 5$에서 x_1의 값이 1이고 x_2의 값이 1인 것을 찾는 것

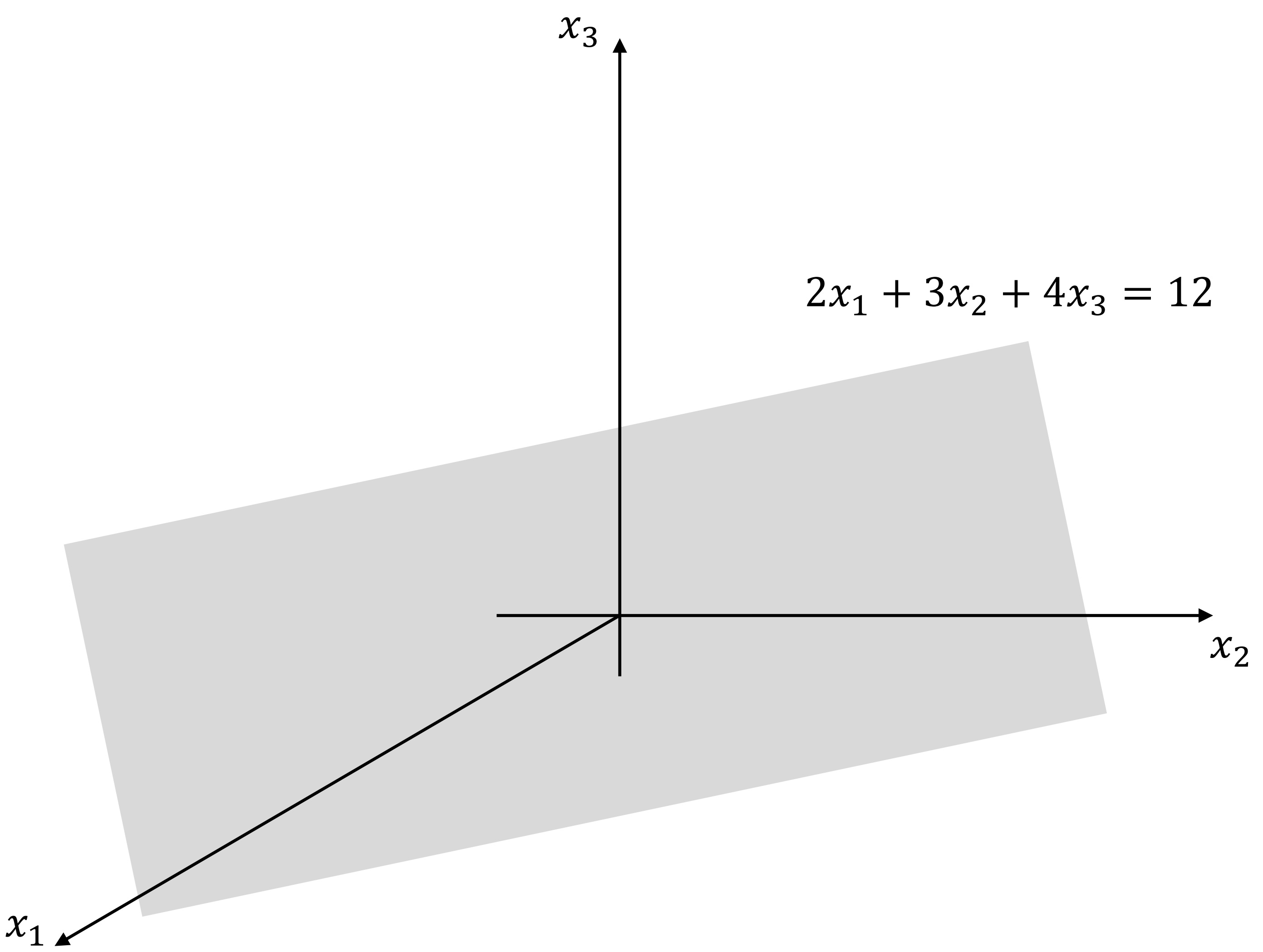
선형방정식

- 선형방정식의 예



선형방정식

- 선형방정식의 예



선형시스템(Linear System)

- 선형 시스템(Linear System)의 정의
  - 선형방정식의 집합
  - 연립1차방정식(system of linear equation)
  - 데이터 분석에서 최적화, 신호처리 등에 활용될 수 있음
- 변수가 2개로 이루어진 선형시스템의 해

● (미지수가 2개인 선형연립방정식의) 해집합의 다양한 모습

① $$\begin{aligned} x_1 + x_2 &= 3 \\ x_1 - x_2 &= 1 \end{aligned}$$

② $$\begin{aligned} 2x_1 - x_2 &= -2 \\ -2x_1 + x_2 &= 2 \end{aligned}$$

③ $$\begin{aligned} 2x_1 - x_2 &= -2 \\ -2x_1 + x_2 &= 4 \end{aligned}$$

  - ①은 한 점에서 만나고, ②는 해가 무수히 많고, ③은 해가 없다.

선형시스템(Linear System)

- 변수가 n개인 선형 시스템으로 일반화
  - N개의 변수를 갖는 m개의 방정식

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned} \longrightarrow A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

  - 첨가행렬(Augmented Matrix)

$$[A \ \vdots \ \mathbf{b}\ ] = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} & \vdots & b_1 \\ a_{21} & a_{22} & \cdots & a_{2n} & \vdots & b_2 \\ \vdots & \vdots & \ddots & \vdots & \vdots & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} & \vdots & b_m \end{bmatrix}$$

선형시스템(Linear System)

- 다음 선형연립방정식을 행렬의 곱을 이용하여 나타내고, 첨가행렬을 구하시오.

$$x_1 + 2x_2 - x_3 = 2$$

$$-2x_1 + x_2 = 1$$

$$x_1 - 3x_2 + 5x_3 = 10$$

동차 선형시스템(homogeneous linear system)

- 동차 선형시스템
  - 우변이 0인 선형시스템

$$\begin{cases} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n = 0 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n = 0 \\ \vdots \qquad \vdots \qquad \qquad \vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n = 0 \end{cases}$$

  - 선형 시스템의 해
    - 오직 하나의 해(x1, x2, x3, ---, xn이 모두 0인 경우)
    - 무한 개의 해

벡터공간(Vector space)

- 벡터공간(Vector space)
  - 벡터의 덧셈과 스칼라 곱이 정의된 공간
  - 선형공간(Linear Space)
  - $\mathbb{R}^n$: n차원, 성분이 n개인 열벡터로 구성
- 차원의 개념
  - 1차원 : 숫자 하나로 표현
  - 2차원 : 각 벡터의 원소가 2개로 구성
  - 3차원 : 각 벡터의 원소가 3개로 구성

벡터공간(Vector Space)

- 유닛벡터(unit vector)
  - 어떤 공간의 좌표 축의 기본 벡터
    - 예시, 3차원 공간($\mathbb{R}^3$)

$$i = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix}, \mathrm{j} = \begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix}, \mathrm{k} = \begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix}$$

  - N차원 벡터는 유닛벡터의 선형결합으로 나타낼 수 있음
    - 예시, 3차원 벡터 a에 대한 유닛벡터의 선형결합

$$\mathrm{a} = \begin{bmatrix} 1 \\ 3 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 3 \\ 0 \end{bmatrix} + \begin{bmatrix} 0 \\ 0 \\ 2 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \\ 0 \end{bmatrix} + 3\begin{bmatrix} 0 \\ 1 \\ 0 \end{bmatrix} + 2\begin{bmatrix} 0 \\ 0 \\ 1 \end{bmatrix} = \mathrm{i} + 3\mathrm{j} + 2\mathrm{k}$$

선형결합(Linear Combinations)

- 선형결합(Linear Combinations)
  - 벡터 $\mathrm{v}_1, \mathrm{v}_2, \cdots, \mathrm{v}_n \in \mathbb{R}^n$ 및 스칼라 $\mathrm{c}_1, \mathrm{c}_2, \cdots, \mathrm{c}_n \in \mathbb{R}^1$가 주어졌을 때, 아래를 $\mathrm{v}_1, \mathrm{v}_2, \cdots, \mathrm{v}_n$의 선형결합이라고 함

$$\mathrm{c}_1\mathrm{v}_1 + \mathrm{c}_2\mathrm{v}_2 + \mathrm{c}_3\mathrm{v}_3 + \cdots + \mathrm{c}_n\mathrm{v}_n$$

  - 여기서 스칼라 값 $\mathrm{c}_1, \mathrm{c}_2, \cdots, \mathrm{c}_n$는 0을 포함한 모든 실수가 될 수 있음
  - 뉴럴 네트워크에서 비슷하게 많이 사용함

선형결합(Linear Combinations)

- 선형시스템에서의 선형결합
  - 데이터 분석에서는 A를 해당 데이터의 레코드, x를 해당 컬럼에 대한 가중치로 해석할 수 있음

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \cdots + a_{1n}x_n &= b_1 \\ a_{21}x_1 + a_{22}x_2 + \cdots + a_{2n}x_n &= b_2 \\ &\vdots \\ a_{m1}x_1 + a_{m2}x_2 + \cdots + a_{mn}x_n &= b_m \end{aligned} \longrightarrow A = \begin{bmatrix} a_{11} & a_{12} & \cdots & a_{1n} \\ a_{21} & a_{22} & \cdots & a_{2n} \\ \vdots & \vdots & & \vdots \\ a_{m1} & a_{m2} & \cdots & a_{mn} \end{bmatrix}, \quad \mathbf{x} = \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{b} = \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_m \end{bmatrix}$$

선형결합(Linear Combinations)

- 아래의 선형시스템을 선형결합의 형태로 변형하시오.

$$\begin{aligned} x_1 + 2x_2 + 3x_3 &= 1 \\ 2x_1 + 5x_2 + 3x_3 &= 3 \\ x_1 \qquad\quad + 8x_3 &= -1 \end{aligned}$$

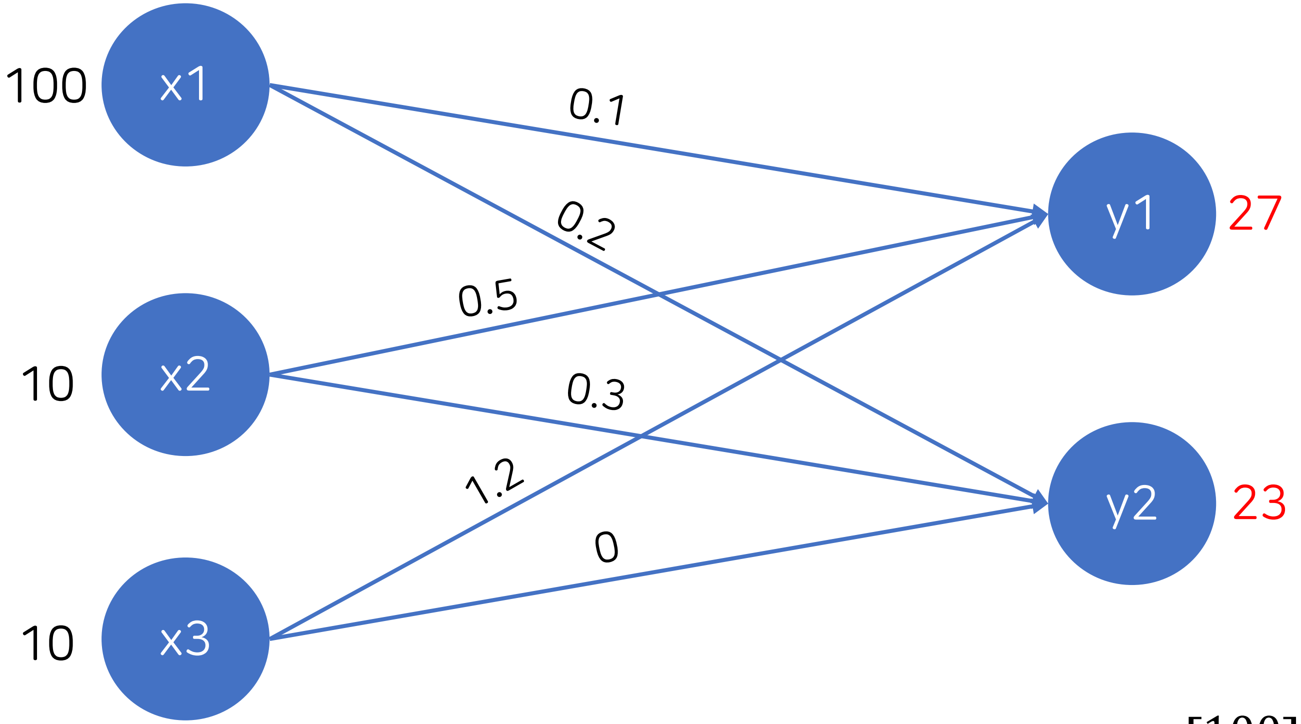
Neural Network와 선형변환

- Neural Network와 선형변환
  - 사람의 성격에 따른 중간고사와 기말고사 점수를 예측한다면?

Index	계획형(x1)	감성형(x2)	외향성(x3)
1	55	77	90
2	35	66	25
3	40	100	65
4	100	10	10
5	70	70	70

Neural Network와 선형변환

- Neural Network로 나타내기
  - Ax = b의 형태로 나타내기 (A는 보통 가중치, x는 데이터 레코드, b는 신경망 통과 후 결과)

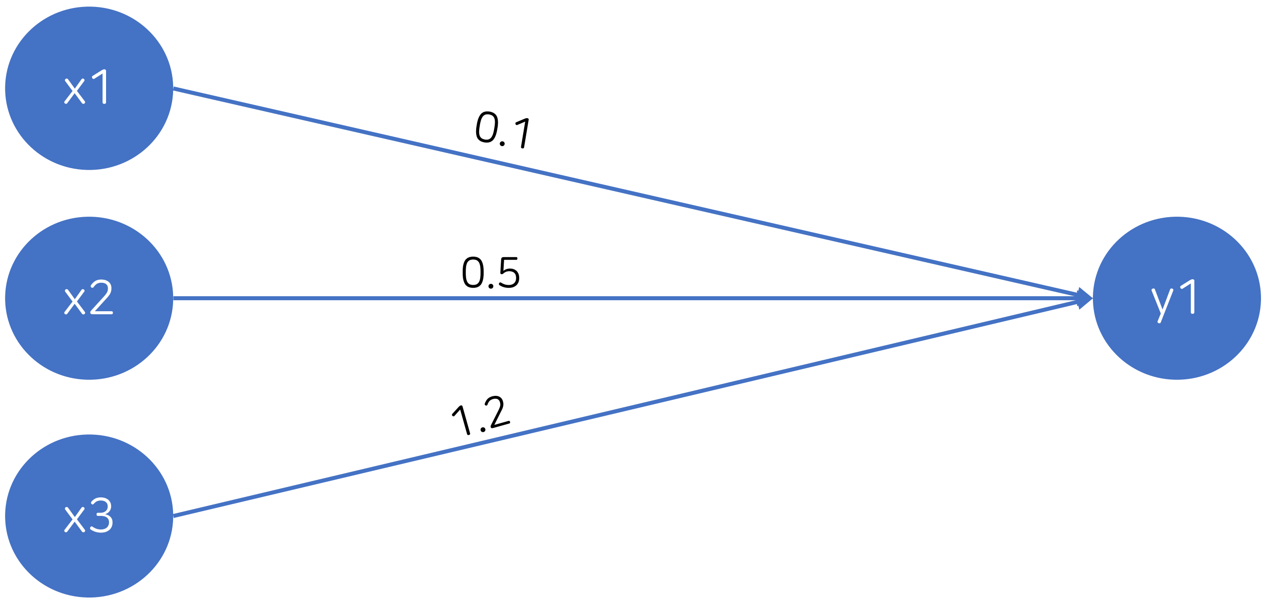


$$\begin{bmatrix} 0.1 & 0.5 & 1.2 \\ 0.2 & 0.3 & 0 \end{bmatrix} \begin{bmatrix} 100 \\ 10 \\ 10 \end{bmatrix} = \begin{bmatrix} 27 \\ 23 \end{bmatrix}$$

Neural Network와 선형변환

- 만약, 앞에서 한 명의 데이터가 아닌, 여러 명의 데이터를 모두 만족시키는 가중치를 찾으려면?

Index	계획형(x1)	감성형(x2)	외향성(x3)	시험 점수
1	55	77	90	75
2	35	66	25	80
3	40	100	65	72
4	100	10	10	77
5	70	70	70	70



Neural Network와 선형변환

- 데이터의 개수가 가중치의 개수보다 많은 경우
  - 명확한 가중치를 찾기 어려움

- Ax = b에서 최대한 Ax와 b의 차이가 없도록 하는 x를 찾아줘야 함!
  - 즉, b-Ax가 최소가 되도록 해줘야 함
    - 여기서 b-Ax를 loss function에 비유할 수 있음
  - b-Ax가 가장 최소가 되도록 하는 x를 최소제곱해라고 함

BOAZ

Neural Network와 선형변환

- 아래 데이터에서 $x = \begin{bmatrix}1\\1\\1\end{bmatrix}$ 인 경우와, $x = \begin{bmatrix}1\\0\\1\end{bmatrix}$ 인 경우로 나누어서 어느 경우가 loss가 최소가 되는 지 비교

Index	계획형(x1)	감성형(x2)	외향성(x3)	시험 점수
1	55	77	90	75
2	35	66	25	80
3	40	100	65	72
4	100	10	10	77
5	70	70	70	70

BOAZ

고윳값 분해(eigen-decomposition)

- 고윳값 분해(eigen-decomposition)
  - 정사각행렬을 고유벡터와 고윳값으로 분해하는 방법
  - 정사각행렬 A의 고유벡터(eigenvector)는 하나의 0이 아닌 벡터 v이며, A와 곱해도 v의 축적(scale)만 변한다는 조건 만족

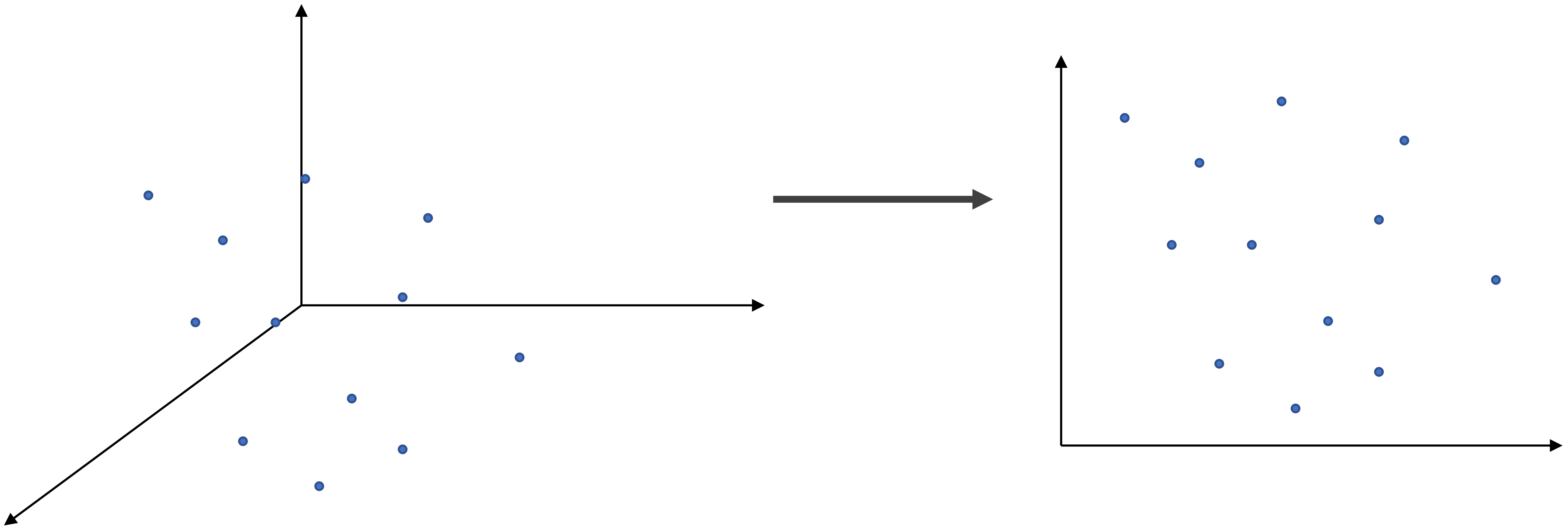
$$\mathrm{Av} = \lambda \mathrm{v}$$

    - 이때, v는 고유벡터, λ는 고유값(eigenvalue)
  - 행렬 A에 선형독립인 n개의 고유벡터 행렬 V와 고윳값 행렬 λ에 대해
    - V = $\left[\mathrm{v}^{(1)}, \mathrm{v}^{(2)}, \cdots, \mathrm{v}^{(n)}\right]$, $\lambda = [\lambda_1, \lambda_2, \cdots, \lambda_\mathrm{n}]^\mathrm{T}$,
    - 데이터 분석에서 고유값이 의미하는 것은 분산을 의미함

$$\mathrm{A} = \mathrm{Vdiag}(\lambda)\mathrm{V}^{-1}$$

특이값 분해(Singular value decomposition)

- 특이값 분해
  - 고윳값 분해와 유사한 개념이며, 차이점은 고윳값 분해는 정방행렬에서만 가능함
  - 행렬의 차원 축소을 위한 도구
  - 주어진 행렬 A의 차원(m)보다 더 낮은 차원(d)의 공간을 찾는 것



특이값 분해(Singular value decomposition)

- 특이값 분해
  - 사각행렬 $A \in \mathbb{R}^{m \times n}$에 대한 특이값 분해는 아래와 같음
  - 모든 정보(Full SVD)보다는 일부 정보(특정 k개만 추출)만 이용하는 특이값 분해를 주로 이용

$$A = U\Sigma V^T$$

$$A = U\Sigma V^T = \begin{bmatrix} \mathbf{u}_1 \mathbf{u}_2 \cdots \mathbf{u}_k \, \mathbf{u}_{k+1} \cdots \mathbf{u}_m \end{bmatrix} \left[\begin{array}{cccc|ccc} \sigma_1 & & & 0 & 0 & \cdots & 0 \\ & \sigma_2 & & & 0 & \cdots & 0 \\ & & \ddots & & \vdots & & \vdots \\ 0 & & & \sigma_k & 0 & \cdots & 0 \\ \hline 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \\ \vdots & \vdots & & \vdots & \vdots & & \vdots \\ 0 & 0 & \cdots & 0 & 0 & \cdots & 0 \end{array}\right] \begin{bmatrix} \mathbf{v}_1^T \\ \mathbf{v}_2^T \\ \vdots \\ \mathbf{v}_n^T \end{bmatrix}$$

Next lecture

- 기초 수학 – 다변수 함수의 미분 + Pandas

Assignments

- Assignments
  - 기초 선형대수학 및 Numpy와 관련된 과제
  - 총 100점 만점, 60점 이상이어야 Pass

- 제출 형식 : ipynb 파일, pdf 파일
- Due Date : 3월 19일 23시 59분까지
  - 지각 제출 : 전체 점수의 80%만 반영